Kerangka Kerja

# AWS Kerangka Well-Architected

# AWS Kerangka Well-Architected: Kerangka Kerja

# Table of Contents

# AWS Kerangka Well-Architected

Tanggal publikasi: 27 Juni 2024 (Revisi dokumen)

The AWS Well-Architected Framework membantu Anda memahami pro dan kontra dari keputusan yang Anda buat saat membangun sistem. AWS Dengan menggunakan Kerangka Kerja ini, Anda akan mengetahui praktik-praktik terbaik berkaitan dengan arsitektur untuk mendesain dan mengoperasikan sistem yang andal, aman, efisien, hemat biaya, dan ramah lingkungan di cloud.

## Pengantar

The AWS Well-Architected Framework membantu Anda memahami pro dan kontra dari keputusan yang Anda buat saat membangun sistem. AWS Kerangka Kerja ini akan membantu Anda mempelajari praktik-praktik terbaik berkaitan dengan arsitektur untuk mendesain dan mengoperasikan beban kerja yang aman, andal, efisien, hemat biaya, dan ramah lingkungan di AWS Cloud. Layanan ini menyediakan cara yang bisa Anda lakukan untuk menilai arsitektur Anda secara terus menerus berdasarkan praktik terbaik dan mengidentifikasi area yang perlu diperbaiki. Proses peninjauan arsitektur adalah percakapan konstruktif tentang keputusan arsitektur, dan tidak dilakukan melalui mekanisme audit. Kami percaya bahwa memiliki sistem yang didesain dengan baik akan meningkatkan peluang keberhasilan bisnis.

AWS Solutions Architects memiliki pengalaman bertahun-tahun dalam merancang solusi di berbagai vertikal bisnis dan kasus penggunaan. Kami telah membantu merancang dan meninjau ribuan arsitektur pelanggan di AWS. Dari pengalaman ini, kami mengidentifikasi praktik terbaik dan strategi inti dalam merancang sistem di cloud.

AWS Well-Architected Framework mendokumentasikan serangkaian pertanyaan mendasar yang membantu Anda memahami apakah arsitektur tertentu selaras dengan praktik terbaik cloud. Kerangka kerja ini memberikan pendekatan yang konsisten untuk mengevaluasi sistem berdasarkan kualitas sistem berbasis cloud yang Anda harapkan, serta perbaikan yang diperlukan untuk mencapai kualitas tersebut. Karena AWS terus berkembang, dan kami terus belajar lebih banyak dari bekerja dengan pelanggan kami, kami akan terus menyempurnakan definisi yang dirancang dengan baik.

Kerangka kerja ini ditujukan bagi mereka yang berperan dalam teknologi, seperti chief technology officer (CTOs), arsitek, pengembang, dan anggota tim operasi. Ini menjelaskan praktik dan strategi AWS terbaik untuk digunakan saat merancang dan mengoperasikan beban kerja cloud, dan menyediakan tautan ke detail implementasi lebih lanjut dan pola arsitektur. Untuk informasi lebih lanjut, lihat beranda AWS Well-Architected.

AWS juga menyediakan layanan untuk meninjau beban kerja Anda tanpa biaya. AWS Well-Architected Tool AWS (WA Tool) adalah layanan di cloud yang menyediakan proses yang konsisten bagi Anda untuk meninjau dan mengukur arsitektur Anda menggunakan Well-Architected Framework. AWS Alat AWS WA memberikan rekomendasi untuk membuat beban kerja Anda lebih andal, aman, efisien, dan hemat biaya.

Untuk membantu Anda dalam menerapkan praktik terbaik, kami telah membuat Lab AWS Well-Architected, yang menyediakan untuk Anda repositori kode dan dokumentasi untuk memberi Anda pengalaman praktik langsung dalam menerapkan praktik terbaik. Kami juga telah bekerja sama dengan AWS Partner Network (APN) Partners terpilih, yang merupakan anggota program AWS Well-Architected Partner. AWS Mitra ini memiliki AWS pengetahuan yang mendalam, dan dapat membantu Anda meninjau dan meningkatkan beban kerja Anda.

# Ketentuan

Setiap hari, para ahli AWS membantu pelanggan dalam merancang sistem untuk memanfaatkan praktik terbaik di cloud. Kami bekerja sama dengan Anda untuk membuat perubahan arsitektur seiring perkembangan desain Anda. Saat Anda melakukan deployment sistem ini ke lingkungan penggunaannya, kami mempelajari seberapa baik kinerja sistem ini dan konsekuensi dari perubahan tersebut.

Berdasarkan apa yang telah kami pelajari, kami telah menciptakan Kerangka AWS Well-Architected, yang menyediakan serangkaian praktik terbaik yang konsisten bagi pelanggan dan mitra untuk mengevaluasi arsitektur, dan memberikan serangkaian pertanyaan yang dapat Anda gunakan untuk mengevaluasi seberapa baik arsitektur selaras dengan praktik terbaik. AWS

AWS Well-Architected Framework didasarkan pada enam pilar — keunggulan operasional, keamanan, keandalan, efisiensi kinerja, optimalisasi biaya, dan keberlanjutan.

Tabel 1. Pilar Kerangka Kerja AWS Well-Architected

| Nama | Deskripsi |
| --- | --- |
| Keunggulan operasional | Kemampuan untuk mendukung pengembangan dan menjalankan beban kerja dengan efektif, mendapatkan wawasan tentang operasi mereka, serta untuk meningkatkan proses dan |

| Nama | Deskripsi |
| --- | --- |
| | prosedur pendukung secara terus menerus untuk memberikan nilai bisnis. |
| Keamanan | Pilar keamanan menjelaskan cara memanfaat kan teknologi cloud untuk melindungi data, sistem, dan aset guna meningkatkan postur keamanan Anda. |
| Keandalan | Pilar keandalan berkenaan dengan kemampuan beban kerja untuk menjalank an fungsinya dengan benar dan konsisten sesuai dengan yang diharapkan. Ini termasuk kemampuan untuk mengoperasikan dan menguji beban kerja di seluruh siklus hidupnya. Paper ini memberikan panduan praktik terbaik yang mendalam untuk menerapkan beban kerja yang andal. AWS |
| Efisiensi kinerja | Kemampuan untuk menggunakan sumber daya komputasi secara efisien untuk memenuhi persyaratan sistem, dan untuk memelihara efisiensi tersebut seiring dengan perubahan permintaan dan perkembangan teknologi. |
| Optimasi biaya | Kemampuan untuk menjalankan sistem guna mendapatkan nilai bisnis dengan harga yang paling rendah. |
| Keberlanjutan | Kemampuan untuk terus meningkatkan dampak pada keberlanjutan dengan mengurangi konsumsi energi dan meningkat kan efisiensi di semua komponen beban kerja dengan memaksimalkan manfaat dari sumber daya yang disediakan dan meminimalkan total sumber daya yang dibutuhkan. |

Dalam Kerangka AWS Well-Architected, kami menggunakan istilah-istilah ini:

- Komponen adalah kode, konfigurasi, dan AWS Sumber Daya yang bersama-sama memenuhi persyaratan. Komponen biasanya berupa unit kepemilikan teknis, dan terpisah dari komponen lainnya.
- Istilah beban kerja digunakan untuk mengidentifikasi serangkaian komponen yang dikombinasikan untuk memberikan nilai bisnis. Beban kerja biasanya merupakan tingkat detail yang dibicarakan oleh pimpinan bisnis dan teknologi.
- Kami berpikir tentang arsitektur sebagai bagaimana komponen bekerja bersama dalam sebuah beban kerja. Cara komunikasi dan interaksi antarkomponen sering menjadi fokus diagram arsitektur.
- Milestones menandai perubahan kunci di arsitektur Anda seiring dengan perkembangannya di siklus hidup produksi (desain, implementasi, pengujian, peluncuran, dan proses produksi).
- Dalam sebuah organisasi, portofolio teknologi adalah sekumpulan beban kerja yang diperlukan agar bisnis dapat beroperasi.
- Tingkat upaya mengategorikan banyaknya waktu, upaya, dan kesulitan untuk mengimplementasikan suatu tugas. Setiap organisasi harus mempertimbangkan ukuran dan keahlian tim serta kompleksitas beban kerja untuk konteks tambahan agar dapat mengategorikan tingkat usaha organisasi dengan tepat.
  - Tinggi: Pekerjaan dapat berlangsung selama beberapa minggu atau bulan. Upaya ini dapat dibagi menjadi beberapa kisah, rilis, dan tugas.
  - Sedang: Pekerjaan dapat berlangsung selama beberapa hari atau minggu. Upaya ini dapat dibagi menjadi beberapa rilis, dan tugas.
  - Rendah: Pekerjaan dapat berlangsung selama beberapa jam atau hari. Upaya ini dapat dibagi menjadi beberapa tugas.

Saat merancang beban kerja, Anda memilah pilar sesuai dengan konteks bisnis Anda. Keputusan bisnis ini dapat mendorong prioritas rekayasa Anda. Anda dapat mengoptimalkan pengurangan dampak terhadap keberlanjutan dan memperkecil biaya dengan mengorbankan keandalan dalam lingkungan pengembangan, atau, untuk solusi yang sangat penting, Anda dapat mengoptimalkan keandalan dengan biaya dan dampak terhadap keberlanjutan yang lebih besar. Dalam solusi e-commerce, kinerja dapat memengaruhi pendapatan dan minat beli pelanggan. Keamanan dan keunggulan operasi umumnya menjadi pilar yang tidak dapat dikorbankan.

# Pada arsitektur

Di lingkungan on-premise, pelanggan sering kali memiliki tim pusat untuk arsitektur teknologi yang bertindak sebagai lapisan atas bagi tim produk lain untuk memastikan mereka mengikuti praktik terbaik. Tim arsitektur teknologi biasanya berisi serangkaian peran, seperti Arsitek Teknis (infrastruktur), Arsitek Solusi (perangkat lunak), Arsitek Data, Arsitek Jaringan, dan Arsitek Keamanan. Seringkali tim ini menggunakan TOGAFatau Zachman Framework sebagai bagian dari kemampuan arsitektur perusahaan.

Di AWS, kami lebih suka mendistribusikan kemampuan ke dalam tim daripada memiliki tim terpusat dengan kemampuan itu. Tentunya akan ada risiko ketika Anda memilih untuk mendistribusikan wewenang pengambilan keputusan, misalnya, memastikan semua tim memenuhi standar internal. Kami meminimalkan risiko ini melalui dua cara. Pertama, kami memiliki latihan (panduan, proses, standar, dan norma yang berlaku) yang berfokus untuk memberikan kemampuan tersebut kepada setiap tim, dan kami memiliki ahli yang memastikan bahwa tim kami telah memenuhi standar yang ditetapkan. Kedua, kami mengimplementasikan mekanisme yang menjalankan pemeriksaan otomatis untuk menjamin pemenuhan standar.

> “Niat baik saja tidak cukup, harus diikuti dengan mekanisme yang bagus untuk mewujudkan sesuatu” — Jeff Bezos.

Artinya, upaya terbaik manusia digantikan dengan mekanisme (seringnya otomatis) yang memeriksa kepatuhan pada aturan atau proses. Pendekatan terdistribusi ini didukung oleh prinsip-prinsip kepemimpinan Amazon, dan menetapkan sebuah budaya di semua peran yang bekerja kembali dari pelanggan. Penelusuran mundur adalah bagian mendasar dalam proses inovasi kami. Kami memulai dari pelanggan dan keinginan mereka, kemudian menggunakan informasi ini sebagai penentu dan pedoman upaya kami. Tim khusus pelanggan membuat produk sesuai kebutuhan pelanggan.

Untuk arsitektur, artinya kami mengharapkan setiap tim dapat membuat arsitektur dan mengikuti praktik terbaik. Untuk membantu tim baru mendapatkan kemampuan ini atau tim yang ada untuk meningkatkan standar mereka, kami mengaktifkan akses ke komunitas virtual insinyur utama yang dapat meninjau desain mereka dan membantu mereka memahami apa praktik AWS terbaik. Komunitas kepala rekayasa pengguna utama berupaya agar praktik terbaik dapat terlihat dan diterapkan. Salah satu contohnya melalui obrolan makan siang yang berfokus pada penerapan praktik terbaik dengan contoh nyata. Obrolan ini direkam dan dapat digunakan sebagai materi orientasi untuk anggota tim baru.

AWS Praktik terbaik muncul dari pengalaman kami menjalankan ribuan sistem pada skala internet. Kami cenderung menggunakan data untuk menentukan praktik terbaik, tetapi selain itu kami juga menggunakan ahli pokok bahasan, seperti kepala rekayasa pengguna utama. Saat kepala rekayasa pengguna utama membentuk praktik terbaik baru, mereka bekerja sebagai komunitas untuk memastikan tim mengikuti mereka. Nantinya, praktik terbaik ini diformalkan ke dalam proses peninjauan internal kami, serta ke dalam mekanisme yang menegakkan kepatuhan. Kerangka Kerja Well-Architected adalah implementasi proses tinjauan internal kami yang digunakan oleh pelanggan, yang telah kami kodifikasi dengan pemikiran kepala rekayasa pengguna utama di berbagai peran, seperti Arsitektur Solusi dan tim rekayasa internal. Kerangka Kerja Well-Architected adalah mekanisme yang dapat diskalakan yang dapat Anda gunakan untuk memanfaatkan pembelajaran ini.

Dengan mengikuti pendekatan komunitas kepala rekayasa pengguna utama dengan kepemilikan arsitektur terdistribusi, kami percaya bahwa arsitektur korporasi Well-Architected dapat dibuat sesuai kebutuhan pelanggan. Pemimpin teknologi (seperti manajer CTOs atau pengembangan), melakukan tinjauan Well-Architected di semua beban kerja Anda akan memungkinkan Anda untuk lebih memahami risiko dalam portofolio teknologi Anda. Dengan pendekatan ini, Anda dapat mengidentifikasi tema di seluruh tim yang dapat ditangani oleh organisasi Anda melalui mekanisme, pelatihan, atau obrolan makan siang di mana kepala rekayasa pengguna utama dapat membagikan pemikirannya tentang area tertentu kepada banyak tim.

## Prinsip desain umum

Kerangka Kerja Well-Architected mengidentifikasi seperangkat prinsip desain umum untuk mendukung desain yang baik di cloud:

- Jangan lagi menebak-nebak kebutuhan kapasitas Anda: Jika Anda tidak menentukan kapasitas dengan baik selama deployment beban kerja, sumber daya Anda yang mahal mungkin tidak akan terpakai atau ada banyak kendala kinerja karena keterbatasan kapasitas. Dengan komputasi cloud, permasalahan ini akan sirna. Anda dapat menggunakan jumlah kapasitas sesuai kebutuhan, dan menaikkan atau menurunkan skalanya secara otomatis.
- Uji sistem dalam skala produksi: Di cloud, Anda dapat membuat lingkungan pengujian berskala produksi sesuai permintaan, menyelesaikan pengujian, kemudian menonaktifkan sumber dayanya. Karena Anda hanya membayar lingkungan pengujian saat sedang berjalan, Anda dapat menyimulasikan lingkungan langsung Anda dengan biaya yang lebih murah daripada pengujian on-premise.
- Lakukan Otomatisasi dengan mempertimbangkan percobaan arsitektural: Dengan otomatisasi, Anda dapat membuat dan mereplikasi beban kerja dengan biaya rendah dan menghindari biaya

untuk upaya manual. Anda dapat melacak perubahan pada otomatisasi, mengaudit dampaknya, dan mengembalikan ke parameter sebelumnya saat dibutuhkan.

- Pertimbangkan arsitektur revolusioner: Di lingkungan tradisional, keputusan arsitektur sering diimplementasikan sebagai peristiwa statis sekali tempo, dengan beberapa versi utama sistem selama masa pakainya. Seiring dengan perkembangan bisnis dan konteksnya, keputusan awal ini dapat menghambat kemampuan sistem untuk memenuhi kebutuhan bisnis yang terus berubah. Di cloud, kemampuan untuk mengotomatiskan dan menguji sesuai permintaan menurunkan risiko dampak perubahan desain. Hal ini memungkinkan sistem berkembang seiring waktu sehingga bisnis dapat memanfaatkan inovasi sebagai praktik standar.
- Dorong arsitektur menggunakan data: Di cloud, Anda dapat mengumpulkan data tentang pengaruh pilihan arsitektur terhadap perilaku beban kerja Anda. Dengan demikian, Anda dapat membuat keputusan sesuai fakta terkait cara memperbaiki beban kerja. Infrastruktur cloud Anda berupa kode, sehingga Anda dapat menggunakan data tersebut untuk menginformasikan pilihan dan peningkatan arsitektur Anda dari waktu ke waktu.
- Tingkatkan melalui game day: Uji kinerja proses dan arsitektur Anda dengan rutin mengadakan game day untuk menyimulasikan peristiwa di produksi. Hal ini akan membantu Anda memahami sisi mana yang perlu ditingkatkan dan membantu mengembangkan pengalaman organisasi dalam menangani peristiwa.

# Pilar kerangka kerja

Membuat sistem perangkat lunak sangat mirip dengan membangun sebuah bangunan. Jika fondasinya tidak kokoh, masalah struktur dapat mengganggu kesatuan dan fungsi bangunan. Saat merancang solusi teknologi, jika Anda mengabaikan enam pilar keunggulan operasional, keamanan, keandalan, efisiensi kinerja, optimisasi biaya, dan keberlanjutan, membangun sistem yang sesuai dengan ekspektasi dan kebutuhan Anda bisa jadi sulit dilakukan. Dengan memasukkan pilar-pilar ini ke dalam arsitektur, Anda lebih mudah dalam memproduksi sistem yang stabil dan efisien. Hal ini akan memungkinkan Anda berfokus pada aspek-aspek desain lain, seperti kebutuhan fungsional.

Pilar

- Keunggulan operasional
- Keamanan
- Keandalan
- Efisiensi kinerja
- Optimalisasi Biaya
- Keberlanjutan

# Keunggulan operasional

Pilar Keunggulan Operasional mencakup kemampuan untuk mendukung pengembangan dan menjalankan beban kerja dengan efektif, mendapatkan wawasan tentang operasi mereka, serta untuk meningkatkan proses dan prosedur pendukung secara terus menerus untuk memberikan nilai bisnis.

Pilar keunggulan operasional menyediakan gambaran umum tentang prinsip, praktik terbaik, dan pertanyaan desain. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keunggulan Operasional.

Topik

- Prinsip desain
- Definisi
- Praktik terbaik
- Sumber daya

# Prinsip desain

Berikut ini adalah prinsip-prinsip desain untuk meraih keunggulan operasional di cloud:

- Membentuk tim-tim berdasarkan tujuan bisnis yang ingin dicapai: Kemampuan sebuah tim untuk mencapai tujuan bisnis berasal dari visi kepemimpinan, operasi yang efektif, dan model operasional yang selaras dengan bisnis. Kepemimpinan harus sepenuhnya diinvestasikan dan berkomitmen untuk CloudOps transformasi dengan model operasi cloud yang sesuai yang memberi insentif kepada tim untuk beroperasi dengan cara yang paling efisien dan memenuhi hasil bisnis. Model operasi yang tepat menggunakan kemampuan personel, proses, dan teknologi untuk menskalakan, mengoptimalkan produktivitas, serta membedakan melalui ketangkasan, responsivitas, dan adaptasi. Visi jangka panjang organisasi diwujudkan menjadi tujuan yang dikomunikasikan di seluruh korporasi kepada pemangku kepentingan dan konsumen layanan cloud Anda. Tujuan dan operasional KPIs diselaraskan di semua tingkatan. Praktik ini menopang nilai jangka panjang yang diperoleh dari penerapan prinsip-prinsip desain berikut.
- Implementasikan observabilitas untuk wawasan yang dapat ditindaklanjuti: Dapatkan pemahaman komprehensif tentang perilaku beban kerja, performa, keandalan, biaya, dan kesehatan. Tetapkan indikator kinerja utama (KPIs) dan manfaatkan telemetri observabilitas untuk membuat keputusan berdasarkan informasi dan mengambil tindakan segera ketika hasil bisnis berisiko. Tingkatkan performa, keandalan, dan biaya secara proaktif berdasarkan data observabilitas yang dapat ditindaklanjuti.
- Lakukan otomatisasi yang aman jika memungkinkan: Di cloud, Anda bisa menerapkan disiplin teknik yang sama yang Anda gunakan untuk kode aplikasi pada seluruh lingkungan Anda. Anda dapat menentukan seluruh beban kerja dan operasinya (aplikasi, infrastruktur, konfigurasi, dan prosedur) sebagai kode, dan memperbaruinya. Anda kemudian dapat mengotomatiskan operasi beban kerja Anda dengan memulainya sebagai respons terhadap peristiwa. Di cloud, Anda dapat menggunakan keamanan otomatisasi dengan mengonfigurasi pagar pembatas, termasuk pengontrolan tingkat, ambang batas kesalahan, dan persetujuan. Melalui otomatisasi yang efektif, Anda dapat mencapai respons yang konsisten terhadap peristiwa, membatasi kesalahan manusia, dan mengurangi kerja keras operator.
- Buat perubahan yang sering, kecil, dan dapat dibatalkan: Rancang beban kerja dapat disesuaikan skalanya dan perijinan komponen-komponennya tidak terlalu bergantung satu sama lain (loosely coupled) untuk dapat diperbarui secara rutin. Teknik deployment otomatis bersama dengan perubahan yang lebih kecil dan bertahap mengurangi radius ledakan dan memungkinkan pembalikan lebih cepat ketika terjadi kegagalan. Hal ini meningkatkan kepercayaan diri untuk

memberikan perubahan yang menguntungkan pada beban kerja Anda sekaligus mempertahankan kualitas dan beradaptasi dengan cepat terhadap perubahan kondisi pasar.

- Perbaiki prosedur operasi sesering mungkin: Saat beban kerja Anda berkembang, prosedur operasional yang mendukungnya juga harus menyesuaikan. Saat Anda menggunakan prosedur operasi, carilah peluang untuk meningkatkannya. Lakukan peninjauan rutin dan pastikan bahwa semua prosedur sudah berlaku efektif dan dipahami dengan baik oleh tim. Jika kesenjangan diidentifikasi, perbarui prosedur yang sesuai. Komunikasikan pembaruan prosedural kepada semua pemangku kepentingan dan tim. Ciptakan mekanisme yang menyenangkan dalam operasi Anda untuk berbagi praktik terbaik dan mengedukasi tim.
- Mengantisipasi kegagalan: Maksimalkan keberhasilan operasional dengan menciptakan skenario kegagalan untuk memahami profil risiko beban kerja dan dampaknya terhadap hasil bisnis Anda. Uji keefektifan prosedur dan respons tim Anda terhadap simulasi kegagalan ini. Ambil keputusan yang bijaksana untuk mengelola risiko terbuka yang diidentifikasi dalam pengujian Anda.
- Pelajari dari semua peristiwa dan metrik operasional: Ciptakan peningkatan melalui pelajaran yang dipetik dari semua peristiwa dan kegagalan operasional. Bagikan materi yang telah dipelajari kepada seluruh tim dan organisasi. Pembelajaran harus menyoroti data dan anekdot tentang bagaimana operasi berkontribusi pada hasil bisnis.
- Gunakan layanan terkelola: Mengurangi beban operasional dengan menggunakan layanan AWS terkelola jika memungkinkan. Bangun prosedur operasional seputar interaksi dengan layanan tersebut.

## Definisi

Ada empat area praktik terbaik untuk keunggulan operasional di cloud:

- Organisasi
- Persiapkan
- Jalankan
- Kembangkan

Kepemimpinan organisasi Anda menentukan tujuan bisnis. Organisasi Anda harus memahami kebutuhan dan prioritas serta menggunakannya untuk mengatur dan melakukan pekerjaan guna mendukung pencapaian hasil bisnis. Beban kerja Anda harus memberikan informasi yang diperlukan untuk mendukungnya. Mengimplementasikan layanan untuk mencapai integrasi, deployment, dan

penyediaan beban kerja Anda akan menciptakan peningkatan alur perubahan yang menguntungkan menuju produksi dengan mengotomatiskan proses yang berulang.

Operasi beban kerja Anda dapat memiliki risiko tersendiri. Anda harus memahami risiko tersebut dan mengambil keputusan yang matang untuk beralih ke produksi. Tim Anda harus mampu mendukung beban kerja Anda. Dengan metrik bisnis dan operasional yang diperoleh dari hasil bisnis yang diinginkan, Anda dapat memahami kondisi beban kerja, aktivitas operasi, serta respons Anda terhadap insiden. Prioritas Anda akan berubah sesuai dengan kebutuhan bisnis Anda dan perubahan lingkungan bisnis. Gunakan ini sebagai loop umpan balik untuk mendorong peningkatan secara berkelanjutan bagi organisasi Anda dan operasi beban kerja Anda.

## Praktik terbaik

> **Note**
>
> Semua pertanyaan keunggulan operasional memiliki OPS awalan sebagai singkatan untuk pilar.

Topik



### Organisasi

Tim Anda harus memiliki pemahaman bersama tentang seluruh beban kerja Anda, peran mereka di dalamnya, dan sasaran bisnis bersama untuk menetapkan prioritas yang akan mendukung kesuksesan bisnis. Prioritas yang terdefinisi dengan baik akan memaksimalkan manfaat dari upaya Anda. Evaluasi kebutuhan internal dan eksternal pelanggan dengan melibatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk menentukan arah fokus upaya. Mengevaluasi kebutuhan pelanggan akan memastikan Anda memiliki pemahaman menyeluruh mengenai dukungan yang dibutuhkan untuk mencapai hasil bisnis. Pastikan Anda mengetahui pedoman atau kewajiban yang ditetapkan oleh tata kelola organisasi Anda serta faktor eksternal, seperti persyaratan kepatuhan peraturan dan standar industri, yang mungkin mewajibkan atau menekankan fokus tertentu. Validasikan bahwa Anda memiliki mekanisme untuk

mengidentifikasi perubahan pada tata kelola internal dan persyaratan kepatuhan eksternal. Jika persyaratan ini belum teridentifikasi, pastikan Anda telah menerapkan uji kelayakan untuk penetapan tersebut. Tinjau prioritas Anda secara berkala agar dapat diperbarui sesuai perubahan kebutuhan.

Evaluasi ancaman pada bisnis (misalnya, risiko dan kewajiban hukum bisnis, serta ancaman keamanan informasi) dan pelihara informasi ini pada registri risiko. Evaluasi dampak risiko, serta kompromi di antara kepentingan yang bertentangan atau pendekatan alternatif. Misalnya, meningkatkan kecepatan fitur baru ke pasar dapat lebih diprioritaskan daripada optimalisasi biaya, atau Anda dapat memilih basis data relasional untuk data non-relasional guna menyederhanakan upaya migrasi sistem tanpa pemfaktoran ulang. Kelola manfaat dan risiko untuk membuat keputusan yang tepat ketika menentukan arah fokus upaya. Risiko atau pilihan tertentu mungkin dapat diterima sesaat, risiko terkait mungkin dapat dimitigasi, atau membiarkan risiko tetap ada mungkin menjadi tidak dapat diterima. Jika demikian, Anda akan mengambil tindakan untuk mengatasi risiko tersebut.

Tim Anda harus memahami peran mereka dalam mencapai hasil bisnis. Tim wajib memahami peran mereka dalam kesuksesan tim lain, peran tim lain dalam kesuksesan mereka, dan memiliki sasaran bersama. Memahami tanggung jawab, kepemilikan, bagaimana keputusan diambil, dan siapa yang memiliki otoritas untuk mengambil keputusan akan membantu memfokuskan upaya dan memaksimalkan manfaat dari tim Anda. Kebutuhan sebuah tim akan dibentuk oleh pelanggan yang mereka dukung, organisasi mereka, formasi tim, dan karakteristik beban kerja mereka. Tidak realistis berharap pada satu model operasional untuk mampu mendukung semua tim dan beban kerja mereka di organisasi Anda.

Pastikan ada pemilik yang teridentifikasi untuk setiap aplikasi, beban kerja, platform, dan komponen infrastruktur, serta ada pemilik yang teridentifikasi untuk setiap proses dan prosedur yang bertanggung jawab atas definisinya, dan pemilik yang bertanggung jawab atas kinerja mereka.

Memahami nilai bisnis setiap komponen, proses, dan prosedur, tentang alasan penyediaan sumber daya atau alasan dilakukannya aktivitas, serta alasan adanya kepemilikan tersebut akan menjadi dasar tindakan anggota tim Anda. Tentukan dengan jelas tanggung jawab anggota tim sehingga mereka bisa bertindak dengan benar dan memiliki mekanisme untuk mengidentifikasi tanggung jawab dan kepemilikan. Miliki mekanisme untuk meminta penambahan, perubahan, dan pengecualian sehingga Anda tidak membatasi inovasi. Tetapkan perjanjian antar-tim yang menjelaskan tentang bagaimana mereka bekerja sama untuk mendukung satu sama lain dan mendukung hasil bisnis Anda.

Sediakan dukungan untuk anggota tim Anda agar mereka bisa menjadi lebih efektif dalam mengambil tindakan dan mendukung hasil bisnis Anda. Kepemimpinan senior yang terlibat harus menetapkan ekspektasi dan mengukur kesuksesan. Pimpinan senior harus menjadi pendukung, penasihat, dan

pendorong untuk mengadopsi praktik terbaik maupun perkembangan organisasi. Biarkan anggota tim mengambil tindakan ketika terdapat risiko pada hasil agar dampak dapat diminimalkan, serta dorong mereka untuk melakukan eskalasi kepada pengambil keputusan dan pemangku kepentingan ketika mereka yakin terdapat risiko agar dapat segera ditangani dan insiden dapat dicegah. Sediakan komunikasi yang tepat waktu, jelas, dan dapat ditindaklanjuti tentang risiko yang diketahui serta peristiwa yang direncanakan agar anggota tim dapat mengambil tindakan yang tepat waktu dan sesuai.

Dorong pelaksanaan eksperimen untuk meningkatkan proses pembelajaran dan menjaga minat serta keterlibatan anggota tim. Tim harus mengembangkan set keterampilan mereka untuk mengadopsi perkembangan teknologi, serta untuk mengimbangi perubahan permintaan dan tanggung jawab. Dukung dan dorong hal ini dengan menyediakan waktu terstruktur khusus untuk pembelajaran. Pastikan anggota tim Anda memiliki sumber daya, berupa alat bantu dan anggota tim, untuk meraih keberhasilan serta skala untuk mendukung hasil bisnis Anda. Manfaatkan keragaman lintas organisasi untuk mencari berbagai perspektif unik. Gunakan perspektif ini untuk meningkatkan inovasi, menantang asumsi Anda, dan mengurangi risiko bias konfirmasi. Kembangkan inklusi, keragaman, dan kemudahan akses dalam tim Anda untuk mendapatkan perspektif yang bermanfaat.

Jika ada peraturan eksternal atau persyaratan kepatuhan yang berlaku untuk organisasi Anda, Anda harus menggunakan sumber daya yang disediakan oleh Kepatuhan Cloud AWS untuk membantu Anda memberikan edukasi bagi tim Anda agar mereka dapat menentukan dampak pada prioritas Anda. Kerangka Kerja Well-Architected menekankan pembelajaran, pengukuran, dan peningkatan. Ini memberikan pendekatan yang konsisten bagi Anda untuk mengevaluasi arsitektur, dan menerapkan desain yang akan skala dari waktu ke waktu. AWS menyediakan AWS Well-Architected Tool untuk membantu Anda meninjau pendekatan Anda sebelum pengembangan, keadaan beban kerja Anda sebelum produksi, dan keadaan beban kerja Anda dalam produksi. Anda dapat membandingkan beban kerja dengan praktik terbaik AWS arsitektur terbaru, memantau statusnya secara keseluruhan, dan mendapatkan wawasan tentang potensi risiko. AWS Trusted Advisor adalah alat yang menyediakan akses ke serangkaian pemeriksaan inti yang merekomendasikan pengoptimalan yang dapat membantu membentuk prioritas Anda. Pelanggan Dukungan Bisnis dan Korporasi menerima akses ke pemeriksaan tambahan yang berfokus pada keamanan, keandalan, kinerja, optimalisasi biaya, dan keberlanjutan yang dapat membantu membentuk prioritas mereka lebih lanjut.

AWS dapat membantu Anda mendidik tim Anda tentang AWS dan layanannya untuk meningkatkan pemahaman mereka tentang bagaimana pilihan mereka dapat berdampak pada beban kerja Anda. Gunakan sumber daya yang disediakan oleh AWS Support (Pusat AWS Pengetahuan, Forum AWS Diskusi, dan AWS Support Pusat) dan AWS Dokumentasi untuk mendidik tim Anda. Hubungi AWS

Support melalui AWS Support Pusat untuk bantuan dengan AWS pertanyaan Anda. AWS juga membagikan praktik dan pola terbaik yang telah kami pelajari melalui pengoperasian AWS di Amazon Builders' Library. Berbagai macam informasi berguna lainnya tersedia melalui AWS Blog dan The Official AWS Podcast. AWS Pelatihan dan Sertifikasi memberikan beberapa pelatihan melalui kursus digital mandiri tentang AWS dasar-dasar. Anda juga dapat mendaftar untuk pelatihan yang dipimpin instruktur untuk lebih mendukung pengembangan keterampilan tim AWS Anda.

Gunakan alat atau layanan yang memungkinkan Anda mengatur lingkungan Anda secara terpusat di seluruh akun, seperti AWS Organizations, untuk membantu mengelola model operasi Anda. Layanan seperti AWS Control Tower memperluas kemampuan manajemen ini dengan memungkinkan Anda menentukan cetak biru (mendukung model operasi Anda) untuk penyiapan akun, menerapkan penggunaan tata kelola berkelanjutan AWS Organizations, dan mengotomatiskan penyediaan akun baru. Penyedia Managed Services seperti AWS Managed Services, AWS Managed Services Partner, atau Managed Services Provider di Jaringan AWS Partner, menyediakan keahlian dalam menerapkan lingkungan cloud, serta mendukung persyaratan keamanan dan kepatuhan serta tujuan bisnis Anda. Menambahkan Layanan Terkelola ke model operasi Anda dapat menghemat waktu dan sumber daya Anda, serta memungkinkan Anda menjaga tim internal Anda untuk bisa tetap belajar dan fokus pada hasil strategis yang akan membedakan bisnis Anda, dan bukan mengembangkan keterampilan dan kemampuan baru.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keunggulan operasional. (Untuk melihat daftar pertanyaan dan praktik terbaik keunggulan operasional, buka Lampiran.)

### OPS1: Bagaimana Anda menentukan apa prioritas Anda?

Setiap orang harus memahami peran mereka dalam mencapai kesuksesan bisnis. Miliki sasaran bersama guna menetapkan prioritas untuk sumber daya. Ini akan memaksimalkan manfaat dari upaya Anda.

### OPS2: Bagaimana Anda menyusun organisasi Anda untuk mendukung hasil bisnis Anda?

Tim Anda harus memahami peran mereka dalam mencapai hasil bisnis. Tim wajib memahami peran mereka dalam kesuksesan tim lain, peran tim lain dalam kesuksesan mereka, dan memiliki sasaran bersama. Memahami tanggung jawab, kepemilikan, bagaimana keputusan diambil, dan siapa yang memiliki otoritas untuk mengambil keputusan akan membantu memfokuskan upaya dan memaksimalkan manfaat dari tim Anda.

OPS3: Bagaimana budaya organisasi Anda mendukung hasil bisnis Anda?

Berikan dukungan kepada anggota tim Anda sehingga mereka dapat menjadi lebih efektif dalam mengambil tindakan dan mendukung hasil bisnis Anda.

Anda mungkin mendapati bahwa Anda ingin mengutamakan sebagian kecil dari prioritas Anda pada titik waktu tertentu. Gunakan pendekatan yang seimbang dalam jangka panjang untuk memastikan pengembangan kemampuan yang diperlukan dan manajemen risiko. Tinjau prioritas Anda secara rutin dan perbarui sesuai perubahan kebutuhan. Ketika tanggung jawab dan kepemilikan tidak ditetapkan atau tidak diketahui, Anda menanggung risiko karena tidak melakukan tindakan yang diperlukan secara tepat waktu serta risiko munculnya upaya yang berulang dan berpotensi bertentangan untuk menangani kebutuhan-kebutuhan tersebut. Budaya organisasi berdampak langsung pada retensi dan kepuasan kerja anggota tim. Dukung keterlibatan dan kemampuan anggota tim Anda untuk mencapai keberhasilan bisnis Anda. Eksperimen diperlukan untuk menciptakan inovasi dan mewujudkan ide. Ketahui bahwa hasil yang tidak sesuai harapan merupakan eksperimen yang berhasil, karena dengan begitu jalur yang tidak mengarahkan kepada keberhasilan dapat diidentifikasi.

## Persiapkan

Untuk menyiapkan keunggulan operasional, Anda harus memahami beban kerja Anda serta perkiraan perilakunya. Dengan begitu Anda akan mampu merancangnya agar dapat menyediakan wawasan tentang statusnya dan membangun prosedur untuk mendukungnya.

Desain beban kerja Anda sedemikian rupa sehingga memberikan informasi yang Anda perlukan untuk memahami status internalnya (seperti metrik, log, dan jejak) di semua komponen untuk mendukung observabilitas dan investigasi masalah. Observabilitas lebih dari sekadar pemantauan sederhana, yang memberikan pemahaman yang komprehensif tentang cara kerja internal sistem berdasarkan output eksternalnya. Berakar pada metrik, log, dan jejak, observabilitas menawarkan wawasan mendalam tentang perilaku dan dinamika sistem. Dengan observabilitas yang efektif, tim dapat membedakan pola, anomali, dan tren, memungkinkan mereka untuk secara proaktif mengatasi masalah potensial dan menjaga kesehatan sistem yang optimal. Mengidentifikasi indikator kinerja utama (KPIs) sangat penting untuk memastikan keselarasan antara kegiatan pemantauan dan tujuan bisnis. Penyelarasan ini memastikan bahwa tim membuat keputusan berbasis data menggunakan metrik yang benar-benar penting, mengoptimalkan kinerja sistem dan hasil bisnis. Selain itu, observabilitas memberdayakan bisnis untuk menjadi proaktif, bukan reaktif. Tim dapat memahami cause-and-effect hubungan dalam sistem mereka, memprediksi dan mencegah masalah

daripada hanya bereaksi terhadapnya. Seiring berkembangnya beban kerja, penting untuk meninjau kembali dan menyempurnakan strategi observabilitas, guna memastikannya tetap relevan dan efektif.

Adopsi pendekatan yang meningkatkan aliran perubahan ke dalam produksi dan yang mencapai pemfaktoran ulang, umpan balik cepat atas kualitas, dan perbaikan bug. Hal-hal ini mempercepat perubahan positif yang memasuki tahap produksi, membatasi masalah yang diterapkan, dan memungkinkan identifikasi serta perbaikan yang cepat terhadap masalah yang muncul dari aktivitas deployment atau yang ditemukan di lingkungan Anda.

Adopsi pendekatan yang memberikan umpan balik cepat atas kualitas dan mencapai pemulihan cepat dari perubahan yang tidak memiliki hasil yang tidak diinginkan. Menggunakan praktik tersebut akan memitigasi dampak masalah akibat deployment perubahan. Antisipasikan perubahan yang tidak berhasil sehingga Anda mampu merespons lebih cepat jika dibutuhkan serta menguji dan memvalidasi perubahan yang Anda buat. Perhatikan aktivitas terencana di lingkungan Anda sehingga Anda dapat mengelola risiko perubahan yang mempengaruhi aktivitas terencana. Prioritaskan perubahan yang sering, kecil, dan dapat dikembalikan untuk membatasi cakupan perubahan. Hal ini menghasilkan pemecahan masalah dan perbaikan yang lebih cepat dengan opsi membatalkan perubahan. Dengan begitu Anda juga dapat memperoleh manfaat dari perubahan yang berharga secara lebih sering.

Evaluasi kesiapan operasional beban kerja, proses, prosedur, dan personel Anda untuk memahami risiko operasional terkait beban kerja Anda. Gunakan proses yang konsisten (termasuk daftar periksa manual dan otomatis) untuk mengetahui saat Anda siap untuk mengoperasikan beban kerja Anda atau untuk melakukan perubahan. Hal ini juga akan membantu Anda menemukan area mana pun yang harus Anda buatkan rencana untuk ditangani. Miliki runbook yang mendokumentasikan aktivitas rutin serta buku pedoman yang memandu proses penyelesaian masalah Anda. Pahami manfaat dan risiko untuk membuat keputusan yang tepat agar perubahan dapat diterapkan dalam produksi.

AWS memungkinkan Anda untuk melihat seluruh beban kerja Anda (aplikasi, infrastruktur, kebijakan, tata kelola, dan operasi) sebagai kode. Hal ini berarti Anda dapat menerapkan disiplin rekayasa yang sama yang Anda gunakan untuk kode aplikasi ke setiap elemen tumpukan Anda dan membagikan semuanya ke seluruh tim atau organisasi untuk memperbesar manfaat upaya pengembangan. Gunakan operasi sebagai kode di cloud dan kemampuan untuk bereksperimen dengan aman guna mengembangkan beban kerja Anda, prosedur operasi Anda, serta antisipasi kegagalan. Menggunakan AWS CloudFormation memungkinkan Anda untuk memiliki lingkungan pengembangan, pengujian, dan produksi kotak pasir yang konsisten, templat, dengan peningkatan tingkat kontrol operasi.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keunggulan operasional.

### OPS4: Bagaimana Anda menerapkan observabilitas dalam beban kerja Anda?

Terapkan observabilitas dalam beban kerja Anda sehingga Anda dapat memahami statusnya dan membuat keputusan berbasis data berdasarkan persyaratan bisnis.

### OPS5: Bagaimana Anda mengurangi cacat, memudahkan remediasi, dan meningkatkan aliran ke produksi?

Adopsi pendekatan yang meningkatkan aliran perubahan ke dalam produksi, yang mencapai pemfaktoran ulang, umpan balik cepat atas kualitas, dan perbaikan bug. Ini mempercepat perubahan yang bermanfaat memasuki produksi, membatasi masalah yang di-deploy, dan mencapai identifikasi cepat serta perbaikan masalah akibat aktivitas deployment.

### OPS6: Bagaimana Anda mengurangi risiko penerapan?

Adopsi pendekatan yang memberikan umpan balik cepat atas kualitas dan mencapai pemulihan cepat dari perubahan yang tidak memiliki hasil yang tidak diinginkan. Menggunakan praktik tersebut akan memitigasi dampak masalah akibat deployment perubahan.

### OPS7: Bagaimana Anda tahu bahwa Anda siap untuk mendukung beban kerja?

Evaluasi kesiapan operasional beban kerja, proses, dan prosedur, serta personel Anda untuk memahami risiko operasional terkait beban kerja Anda.

Berinvestasi dalam implementasi aktivitas operasi sebagai kode untuk memaksimalkan produktivitas personel operasi, meminimalkan tingkat kesalahan, dan mencapai respons otomatis. Gunakan “pre-mortem” untuk mengantisipasi kegagalan dan membuat prosedur ketika diperlukan. Terapkan metadata menggunakan Tag Sumber Daya dan AWS Resource Groups ikuti strategi penandaan yang konsisten untuk mencapai identifikasi sumber daya Anda. Tandai sumber daya Anda untuk pengaturan, akuntansi biaya, kontrol akses, dan penargetan pelaksanaan aktivitas operasi otomatis. Adopsi praktik deployment yang memanfaatkan elastisitas cloud untuk memfasilitasi aktivitas pengembangan, dan pra-deployment sistem untuk implementasi yang lebih cepat. Ketika Anda

membuat perubahan pada daftar periksa yang Anda gunakan untuk mengevaluasi beban kerja Anda, rencanakan apa yang akan Anda lakukan dengan sistem langsung yang tidak lagi patuh.

## Jalankan

Observabilitas memungkinkan Anda fokus pada data yang bermakna serta memahami interaksi dan output beban kerja Anda. Dengan berkonsentrasi pada wawasan penting dan menghilangkan data yang tidak perlu, Anda mempertahankan pendekatan langsung untuk memahami kinerja beban kerja. Hal ini sangat penting tidak hanya untuk mengumpulkan data tetapi juga untuk menafsirkannya dengan benar. Menentukan garis acuan yang jelas, menetapkan ambang batas peringatan yang sesuai, dan memantau secara aktif setiap penyimpangan. Pergeseran metrik kunci, terutama ketika berkorelasi dengan data lain, dapat menunjukkan dengan tepat area masalah tertentu. Dengan observabilitas, Anda lebih siap untuk memperkirakan dan mengatasi tantangan potensial, memastikan bahwa beban kerja Anda beroperasi dengan lancar dan memenuhi kebutuhan bisnis.

Keberhasilan operasi beban kerja diukur dengan pencapaian hasil bisnis dan pelanggan. Tetapkan hasil yang diharapkan, tentukan bagaimana keberhasilan akan diukur, dan identifikasi metrik yang akan digunakan pada perhitungan tersebut untuk menentukan apakah beban kerja dan operasi Anda berhasil. Kondisi operasional meliputi kondisi beban kerja serta kondisi dan keberhasilan aktivitas operasi yang dilakukan dalam dukungan beban kerja (misalnya, deployment dan respons insiden). Tetapkan baris acuan metrik untuk peningkatan, investigasi, serta intervensi, kumpulkan dan analisis metrik Anda, kemudian validasi pemahaman Anda tentang keberhasilan operasi dan bagaimana hal tersebut berubah seiring waktu. Gunakan metrik yang dikumpulkan untuk menentukan apakah Anda memenuhi kebutuhan pelanggan dan bisnis, serta mengidentifikasi area yang perlu ditingkatkan.

Manajemen peristiwa operasional yang efektif dan efisien diperlukan untuk mencapai keunggulan operasional. Hal ini berlaku untuk peristiwa operasional baik yang terencana maupun tidak terencana. Gunakan runbook yang telah dibuat untuk peristiwa yang dipahami dengan baik, dan gunakan buku panduan untuk membantu investigasi dan resolusi masalah. Prioritaskan respons terhadap peristiwa berdasarkan dampaknya pada bisnis dan pelanggan. Pastikan bahwa jika muncul peringatan sebagai respons terhadap suatu peristiwa, ada proses terkait untuk dijalankan, dengan pemilik yang diidentifikasi secara spesifik. Tentukan terlebih dulu personel yang dibutuhkan untuk menyelesaikan suatu peristiwa dan sertakan proses eskalasi agar dapat melibatkan personel tambahan, jika diperlukan, berdasarkan urgensi dan dampaknya. Identifikasi dan libatkan individu yang memiliki wewenang untuk membuat keputusan mengenai tindakan yang akan menimbulkan dampak bisnis dari respons peristiwa yang belum ditangani sebelumnya.

Komunikasikan status operasional beban kerja melalui dasbor dan pemberitahuan yang disesuaikan dengan audiens target (misalnya, pelanggan, bisnis, pengembang, operasi) sehingga mereka bisa

mengambil tindakan yang sesuai, ekspektasi mereka terkelola, serta mereka mendapatkan informasi ketika operasi kembali normal.

Di AWS, Anda dapat menghasilkan tampilan dasbor dari metrik yang dikumpulkan dari beban kerja dan secara native. AWS Anda dapat memanfaatkan CloudWatch atau aplikasi pihak ketiga untuk menggabungkan dan menyajikan tampilan tingkat bisnis, beban kerja, dan operasi dari aktivitas operasi. AWS memberikan wawasan beban kerja melalui kemampuan logging termasuk AWS X-Ray,, CloudWatch CloudTrail, dan VPC Flow Logs untuk mengidentifikasi masalah beban kerja dalam mendukung analisis akar penyebab dan remediasi.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keunggulan operasional.

OPS8: Bagaimana Anda memanfaatkan observabilitas beban kerja di organisasi Anda?

Memastikan kondisi beban kerja yang optimal dengan memanfaatkan observabilitas. Memanfaat kan metrik, log, dan jejak yang relevan untuk mendapatkan pandangan komprehensif tentang kinerja beban kerja Anda dan mengatasi masalah secara efisien.

OPS9: Bagaimana Anda memahami kesehatan operasi Anda?

Tetapkan, catat, dan analisis metrik operasi untuk mendapatkan visibilitas peristiwa operasi sehingga Anda dapat mengambil tindakan yang tepat.

OPS10: Bagaimana Anda mengelola beban kerja dan peristiwa operasi?

Siapkan dan validasikan prosedur untuk merespons peristiwa guna meminimalkan gangguannya pada beban kerja Anda.

Semua metrik yang Anda kumpulkan harus selaras dengan kebutuhan bisnis dan hasil yang didukung. Kembangkan respons dalam skrip untuk memahami peristiwa dengan baik dan otomatiskan respons tersebut saat ada peristiwa yang dikenali.

## Kembangkan

Belajar, berbagi, dan terus melakukan peningkatan untuk mempertahankan keunggulan operasional. Dedikasikan siklus kerja untuk melakukan perbaikan bertahap yang hampir terus-menerus. Lakukan

analisis pasca-insiden tentang semua peristiwa yang memengaruhi pelanggan. Identifikasi faktor kontribusi dan tindakan preventif untuk meminimalkan atau mencegah kemungkinan terjadi lagi. Komunikasikan faktor penyebab kepada komunitas yang terpengaruh jika perlu. Evaluasi secara teratur dan prioritaskan peluang untuk peningkatan (misalnya, permintaan fitur, perbaikan masalah, serta persyaratan kepatuhan), termasuk beban kerja dan prosedur operasi.

Sertakan loop umpan balik dalam prosedur Anda untuk mengidentifikasi dengan cepat area yang perlu ditingkatkan serta menangkap pembelajaran dari operasi yang berjalan.

Bagikan pelajaran yang didapatkan kepada seluruh tim untuk membagikan manfaat dari pelajaran tersebut. Analisis tren dalam pelajaran yang didapatkan dan lakukan analisis metrik operasi secara retrospektif lintas tim untuk mengidentifikasi peluang dan metode untuk peningkatan. Implementasikan perubahan yang ditujukan untuk menghadirkan peningkatan dan evaluasi hasil untuk menentukan keberhasilan.

Pada AWS, Anda dapat mengekspor data log Anda ke Amazon S3 atau mengirim log langsung ke Amazon S3 untuk penyimpanan jangka panjang. Dengan menggunakan AWS Glue, Anda dapat menemukan dan menyiapkan data log Anda di Amazon S3 untuk analitik, dan menyimpan metadata terkait di. AWS Glue Data Catalog Amazon Athena, melalui integrasi aslinya dengan AWS Glue, kemudian dapat digunakan untuk menganalisis data log Anda, menanyakannya menggunakan standar. SQL Menggunakan alat intelijen bisnis seperti Amazon QuickSight, Anda dapat memvisualisasikan, menjelajahi, dan menganalisis data Anda. Menemukan tren dan peristiwa ketertarikan yang bisa mendorong peningkatan.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keunggulan operasional.

### OPS11: Bagaimana Anda mengembangkan operasi?

Luangkan waktu dan sumber daya khusus untuk peningkatan bertahap yang hampir berkelanjutan untuk meningkatkan dan efisiensi operasi Anda.

Keberhasilan pengembangan operasi terbentuk dari: peningkatan kecil yang sering, penyediaan lingkungan yang aman dan waktu untuk bereksperimen, mengembangkan, serta menguji peningkatan; dan lingkungan yang mendukung untuk belajar dari kegagalan. Dukungan operasi untuk lingkungan sandbox, pengembangan, pengujian, dan produksi, dengan meningkatkan tingkat kontrol operasi, memfasilitasi pengembangan dan meningkatkan prediktabilitas hasil yang sukses dari perubahan yang diterapkan ke produksi.

## Sumber daya

Buka sumber daya berikut untuk mempelajari selengkapnya tentang praktik terbaik kami untuk Keunggulan Operasional.

### Dokumentasi

- DevOps dan AWS

### Laporan resmi

- Pilar Keunggulan Operasional

### Video

- DevOps di Amazon

# Keamanan

Pilar Keamanan berkenaan dengan kemampuan untuk melindungi data, sistem, dan aset untuk memanfaatkan teknologi cloud guna meningkatkan keamanan Anda.

Pilar keamanan memberikan gambaran umum tentang prinsip, praktik terbaik, dan pertanyaan desain. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keamanan.

Topik

- Prinsip desain
- Definisi
- Praktik terbaik
- Sumber daya

## Prinsip desain

Ada sejumlah prinsip di cloud yang dapat membantu Anda memperkuat keamanan beban kerja Anda:

- Menerapkan fondasi identitas yang kuat: Menerapkan prinsip hak istimewa paling sedikit dan menegakkan pemisahan tugas dengan otorisasi yang sesuai untuk setiap interaksi dengan sumber daya Anda. AWS Pusatkan manajemen identitas, dan targetkan untuk tidak bergantung pada kredensial statis jangka panjang.
- Menjaga keterlacakan: Pantau, munculkan peringatan, dan audit tindakan serta perubahan dalam lingkungan Anda secara waktu nyata. Integrasikan pengumpulan log dan metrik dengan sistem agar dapat bertindak berdasarkan investigasi yang berjalan otomatis.
- Terapkan keamanan di semua lapisan: Terapkan pertahanan secara mendalam dengan banyak kontrol keamanan. Terapkan ke semua lapisan (misalnya, tepi jaringanVPC, penyeimbangan beban, setiap instance dan layanan komputasi, sistem operasi, aplikasi, dan kode).
- Lakukan otomatisasi praktik terbaik keamanan: Mekanisme keamanan berbasis perangkat lunak otomatis meningkatkan kemampuan Anda untuk meningkatkan skala dengan lebih cepat, hemat biaya, dan aman. Ciptakan arsitektur yang aman, termasuk implementasi kontrol yang ditentukan dan dikelola sebagai kode dalam templat yang dikontrol versi.
- Lindungi data bergerak dan data diam: Klasifikasikan data sesuai tingkatan sensitivitasnya dan mekanisme pengunaannya, seperti enkripsi, tokenisasi dan kontrol akses jika sesuai.
- Jauhkan keterlibatan manusia dalam penanganan data: Gunakan mekanisme dan alat untuk mengurangi atau menghilangkan akses langsung atau pemrosesan data secara manual. Ini akan mengurangi risiko kekeliruan atau perubahan dan kesalahan manusia dalam penanganan data sensitif.
- Bersiap untuk peristiwa keamanan: Bersiaplah menghadapi insiden dengan membentuk manajemen insiden serta proses dan kebijakan investigasi yang selaras dengan kebutuhan organisasi Anda. Jalankan simulasi tanggap-insiden dan gunakan alat dengan otomatisasi untuk mempercepat deteksi, investigasi, dan pemulihan.

## Definisi

Terdapat tujuh area praktik terbaik untuk keamanan di cloud:

- Fondasi keamanan
- Pengelolaan identitas dan akses
- Deteksi
- Perlindungan infrastruktur
- Perlindungan data

- Respons insiden
- Keamanan aplikasi

Sebelum merancang beban kerja, Anda harus menerapkan praktik yang berpengaruh terhadap keamanan. Tentukan peran untuk setiap orang. Selain itu, Anda perlu mengidentifikasi insiden keamanan, melindungi sistem dan layanan Anda, serta menjaga kerahasiaan dan integritas data melalui perlindungan data. Anda harus memiliki proses yang terdefinisi dengan baik dan dapat dipraktikkan untuk merespons insiden keamanan. Alat dan teknik ini penting karena dapat mendukung berbagai sasaran, seperti mencegah kerugian finansial atau mematuhi peraturan yang berlaku.

Model Tanggung Jawab AWS Bersama membantu organisasi yang mengadopsi cloud untuk mencapai tujuan keamanan dan kepatuhan mereka. Karena AWS secara fisik mengamankan infrastruktur yang mendukung layanan cloud kami, sebagai AWS pelanggan Anda dapat fokus menggunakan layanan untuk mencapai tujuan Anda. AWS Cloud juga menyediakan akses yang lebih besar ke data keamanan dan pendekatan otomatis untuk menanggapi peristiwa keamanan.

## Praktik terbaik

Topik



### Keamanan

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keamanan ini. (Untuk melihat daftar pertanyaan dan praktik terbaik keamanan, buka Lampiran.).

SEC1: Bagaimana Anda mengoperasikan beban kerja Anda dengan aman?

Untuk mengoperasikan beban kerja dengan aman, Anda harus menerapkan praktik terbaik yang menyeluruh ke setiap area keamanan. Pilih persyaratan dan proses yang telah Anda tetapkan dalam keunggulan operasional di tingkat organisasi dan beban kerja, lalu terapkan ke semua area.

Tetap up to date dengan rekomendasi dari AWS, sumber industri, dan intelijen ancaman membantu Anda mengembangkan model ancaman dan tujuan pengendalian Anda. Otomatisasi proses, pengujian, dan validasi keamanan memungkinkan Anda menskalakan operasi keamanan Anda.

Dalam AWS, memisahkan beban kerja yang berbeda berdasarkan akun, berdasarkan fungsi dan kepatuhan atau persyaratan sensitivitas data, adalah pendekatan yang disarankan.

## Pengelolaan identitas dan akses

Manajemen identitas dan akses adalah bagian penting dari program keamanan informasi, yang memastikan bahwa hanya pengguna dan komponen yang sah dan diautentikasi yang dapat mengakses sumber daya Anda, dan hanya melalui cara yang Anda izinkan. Misalnya, Anda harus menentukan prinsipal (yaitu, akun, pengguna, peran, dan layanan yang dapat melakukan tindakan di akun Anda), membuat kebijakan yang selaras dengan prinsipal ini, dan menerapkan manajemen kredensial yang kuat. Elemen manajemen hak istimewa ini membentuk inti autentikasi dan otorisasi.

Pada tahun AWS, manajemen hak istimewa terutama didukung oleh layanan AWS Identity and Access Management (IAM), yang memungkinkan Anda untuk mengontrol akses pengguna dan program ke AWS layanan dan sumber daya. Anda harus menerapkan kebijakan yang terperinci, yang memberikan izin kepada pengguna, grup, peran, atau sumber daya. Anda juga memiliki kemampuan untuk meminta praktik kata sandi yang kuat, seperti tingkat kompleksitas, menghindari penggunaan kembali, dan menegakkan otentikasi multi-faktor (). MFA Anda dapat menggunakan federasi dengan layanan direktori yang ada. Untuk beban kerja yang memerlukan sistem untuk memiliki akses AWS, IAM memungkinkan akses aman melalui peran, profil instans, federasi identitas, dan kredensi sementara.

Pertanyaan berikut ini berfokus pada pertimbangan untuk keamanan ini.

### SEC2: Bagaimana Anda mengelola identitas untuk orang dan mesin?

Ada dua jenis identitas yang perlu Anda kelola saat mendekati beban AWS kerja yang aman. Pemahaman tentang jenis identitas yang harus Anda kelola dan berikan akses akan membantu Anda memverifikasi identitas yang tepat memiliki akses ke sumber daya yang tepat dalam kondisi yang tepat.

Identitas Manusia: Administrator, pengembang, operator, dan pengguna akhir Anda memerluka n identitas untuk mengakses AWS lingkungan dan aplikasi Anda. Ini adalah anggota organisasi Anda, atau pengguna eksternal dengan siapa Anda berkolaborasi, dan yang berinteraksi dengan AWS sumber daya Anda melalui browser web, aplikasi klien, atau alat baris perintah interaktif.

Identitas Mesin: Aplikasi layanan, alat operasional, dan beban kerja Anda memerlukan identitas untuk membuat permintaan ke AWS layanan, misalnya, untuk membaca data. Identitas ini mencakup mesin yang berjalan di AWS lingkungan Anda seperti EC2 instans atau AWS Lambda fungsi Amazon. Anda juga dapat mengelola identitas mesin untuk pihak eksternal yang membutuhkan akses. Selain itu, Anda mungkin juga memiliki mesin di luar AWS yang membutuhk an akses ke AWS lingkungan Anda.

### SEC3: Bagaimana Anda mengelola izin untuk orang dan mesin?

Kelola izin untuk mengontrol akses ke orang dan identitas mesin yang memerlukan akses AWS dan beban kerja Anda. Izin akan mengontrol siapa yang dapat mengakses hal tertentu, beserta kondisinya.

Kredensial tidak boleh dibagikan ke pengguna atau sistem lain. Akses pengguna harus diberikan menggunakan pendekatan hak istimewa paling rendah dengan praktik terbaik termasuk persyaratan kata sandi dan ditegakkan. MFA Akses terprogram, termasuk API panggilan ke AWS layanan, harus dilakukan dengan menggunakan kredenal hak istimewa sementara dan terbatas, seperti yang dikeluarkan oleh. AWS Security Token Service

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

| Pengguna mana yang membutuhkan akses programatis? | Untuk | Oleh |
| --- | --- | --- |
| Identitas tenaga kerja<br><br>(Pengguna dikelola di Pusat IAM Identitas) | Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs | Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.<br><br>• Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna.<br>• Untuk AWS SDKs, alat, dan AWS APIs, lihat otentikas i Pusat IAM Identitas di Panduan Referensi Alat AWS SDKs dan Alat. |
| IAM | Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs | Mengikuti petunjuk dalam Menggunakan kredensil sementara dengan AWS sumber daya di IAMPanduan Pengguna. |
| IAM | (Tidak direkomendasikan) Gunakan kredensi jangka panjang untuk menandata ngani permintaan terprogra m ke AWS CLI,, AWS SDKs atau. AWS APIs | Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.<br><br>• Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensil IAM pengguna di Panduan Pengguna.AWS Command Line Interface |

| Pengguna mana yang membutuhkan akses programatis? | Untuk | Oleh |
| --- | --- | --- |
| | | • Untuk AWS SDKs dan alat, lihat Mengautentikasi menggunakan kredensil jangka panjang di Panduan Referensi Alat AWS SDKs dan Alat.<br>• Untuk AWS APIs, lihat Mengelola kunci akses untuk IAM pengguna di Panduan IAM Pengguna. |

AWS menyediakan sumber daya yang dapat membantu Anda dengan identitas dan manajemen akses. Untuk membantu mempelajari praktik terbaik, jelajahi lab langsung kami tentang mengelola kredensi & otentikasi, mengendalikan akses manusia, dan mengendalikan akses programatik.

## Deteksi

Anda dapat menggunakan kontrol-kontrol detektif untuk mengidentifikasi potensi ancaman atau insiden keamanan. Hal ini merupakan bagian yang sangat penting dalam kerangka kerja tata kelola dan dapat digunakan untuk mendukung proses yang berkualitas, kewajiban kepatuhan atau hukum, serta upaya identifikasi dan respons terhadap ancaman. Ada beberapa jenis kontrol detektif. Misalnya, melakukan inventarisasi aset dan detail atributnya mendorong pengambilan keputusan (dan kontrol siklus hidup) yang lebih efektif untuk membantu menetapkan dasar operasional. Anda juga dapat menggunakan audit internal, pemeriksaan kontrol yang terkait dengan sistem informasi, untuk memverifikasi bahwa praktik yang dijalankan telah memenuhi kebijakan serta persyaratan dan bahwa Anda telah menetapkan notifikasi peringatan otomatis sesuai kondisi yang ditentukan. Kontrol ini merupakan faktor reaktif penting yang dapat membantu organisasi Anda mengidentifikasi dan memahami cakupan aktivitas beranomali.

Di AWS, Anda dapat menerapkan kontrol detektif dengan memproses log, peristiwa, dan pemantauan yang memungkinkan audit, analisis otomatis, dan mengkhawatirkan. CloudTrail log, AWS API panggilan, dan CloudWatch menyediakan pemantauan metrik dengan mengkhawatirkan, dan AWS Config menyediakan riwayat konfigurasi. Amazon GuardDuty adalah layanan deteksi

ancaman terkelola yang terus memantau perilaku berbahaya atau tidak sah untuk membantu Anda melindungi AWS akun dan beban kerja Anda. Log tingkat layanan juga tersedia, misalnya, Anda dapat menggunakan Amazon Simple Storage Service (Amazon S3) untuk membuat log permintaan akses.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keamanan ini.

SEC4: Bagaimana Anda mendeteksi dan menyelidiki peristiwa keamanan?

Catat dan analisis peristiwa dari log dan metrik untuk mendapatkan visibilitas. Ambil tindakan atas peristiwa keamanan dan potensi ancaman untuk membantu mengamankan beban kerja Anda.

Manajemen log sangat penting dalam beban kerja Well-Architected untuk berbagai alasan, mulai dari keamanan atau forensik hingga persyaratan hukum atau regulasi. Anda harus menganalisis log dan meresponsnya agar Anda dapat mengidentifikasi potensi insiden keamanan. AWS menyediakan fungsionalitas yang memudahkan manajemen log dengan memberikan kemampuan untuk menentukan siklus hidup retensi data atau menentukan tempat penyimpanan, pengarsipan, dan penghapusan data. Hal ini mempermudah penanganan data yang dapat diprediksi, andal, dan lebih hemat biaya.

## Perlindungan infrastruktur

Perlindungan infrastruktur berkaitan dengan metodologi kontrol, seperti pertahanan mendalam, yang diperlukan untuk memenuhi praktik terbaik dan kewajiban organisasi atau peraturan. Penggunaan metodologi ini sangat krusial untuk keberhasilan dan keberlangsungan operasi, baik di cloud maupun on-premise.

Di AWS, Anda dapat menerapkan inspeksi paket stateful dan stateless, baik dengan menggunakan teknologi AWS-native atau dengan menggunakan produk dan layanan mitra yang tersedia melalui aplikasi. AWS Marketplace Anda harus menggunakan Amazon Virtual Private Cloud (AmazonVPC) untuk membuat lingkungan pribadi, aman, dan skalabel di mana Anda dapat menentukan topologi Anda—termasuk gateway, tabel perutean, dan subnet publik dan pribadi.

Pertanyaan berikut ini berfokus pada pertimbangan untuk keamanan ini.

SEC5: Bagaimana Anda melindungi sumber daya jaringan Anda?

Setiap beban kerja yang memiliki suatu bentuk konektivitas jaringan, baik internet atau jaringan privat, memerlukan beberapa lapisan pertahanan untuk membantu melindungi dari ancaman eksternal dan internal berbasis jaringan.

SEC6: Bagaimana Anda melindungi sumber daya komputasi Anda?

Sumber daya komputasi di beban kerja Anda memerlukan beberapa lapisan pertahanan untuk membantu melindungi dari ancaman eksternal dan internal. Sumber daya komputasi mencakup EC2 instance, kontainer, AWS Lambda fungsi, layanan basis data, perangkat IoT, dan banyak lagi.

Sebaiknya terapkan banyak lapisan keamanan di semua jenis lingkungan. Untuk perlindungan infrastruktur, banyak konsep dan metode yang dapat digunakan untuk model cloud dan on-premise. Menerapkan perlindungan batas, memantau titik masuk dan keluar, serta pencatatan log, pemantauan, dan peringatan yang komprehensif sangat penting dalam rencana keamanan informasi yang efektif.

AWS pelanggan dapat menyesuaikan, atau mengeraskan, konfigurasi Amazon Elastic Compute Cloud (Amazon)EC2, wadah Amazon Elastic Container Service (AmazonECS), atau AWS Elastic Beanstalk instance, dan mempertahankan konfigurasi ini ke Amazon Machine Image () yang tidak dapat diubah. AMI Kemudian, apakah diluncurkan oleh Auto Scaling atau diluncurkan secara manual, semua server virtual baru (instance) yang diluncurkan dengan ini AMI menerima konfigurasi yang diperkeras.

## Perlindungan data

Sebelum merancang sistem apa pun, praktik dasar yang memengaruhi keamanan harus diterapkan. Misalnya, klasifikasi data menjadi cara untuk mengategorikan data organisasi berdasarkan tingkat sensitivitas, dan enkripsi melindungi data dengan membuatnya tidak dapat dikenali oleh akses tidak sah. Alat dan teknik ini penting karena dapat mendukung berbagai sasaran, seperti mencegah kerugian finansial atau mematuhi peraturan yang berlaku.

Dalam AWS, praktik berikut memfasilitasi perlindungan data:

- Sebagai AWS pelanggan, Anda memiliki kendali penuh atas data Anda.

- AWS memudahkan Anda untuk mengenkripsi data dan mengelola kunci, termasuk rotasi kunci biasa, yang dapat dengan mudah diotomatisasi oleh AWS atau dikelola oleh Anda.
- Tersedia log mendetail yang berisi konten penting seperti perubahan dan akses file.
- AWS telah merancang sistem penyimpanan untuk ketahanan yang luar biasa. Misalnya, Amazon S3 Standard, S3 Standard–IA, S3 One Zone-IA, dan Amazon Glacier dirancang untuk memberikan 99,999999999% ketahanan objek dalam setahun. Tingkat ketahanan ini sesuai dengan perkiraan rata-rata penurunan tahunan pada 0,000000001% objek.
- Penentuan versi, yang dapat menjadi bagian dari proses pengelolaan siklus hidup data yang lebih besar, dapat melindungi dari penimpaan, penghapusan, dan kerusakan serupa yang tidak disengaja.
- AWS tidak pernah memulai pergerakan data antar Daerah. Konten yang ditempatkan pada suatu Wilayah tidak akan berpindah kecuali jika Anda menggunakan fitur atau menggunakan layanan untuk memindahkannya.

Pertanyaan berikut ini berfokus pada pertimbangan untuk keamanan ini.

### SEC7: Bagaimana Anda mengklasifikasikan data Anda?

Klasifikasi memberikan cara untuk mengategorikan data, berdasarkan tingkat kekritisan dan sensitivitas untuk membantu Anda menentukan kontrol retensi dan perlindungan yang sesuai.

### SEC8: Bagaimana Anda melindungi data Anda saat istirahat?

Lindungi data diam dengan mengimplementasikan beberapa kontrol, untuk mengurangi risiko akses tanpa otorisasi atau penanganan yang salah.

### SEC9: Bagaimana Anda melindungi data Anda dalam perjalanan?

Lindungi data bergerak dengan mengimplementasikan beberapa kontrol untuk mengurangi risiko akses tanpa otorisasi atau hilangnya data.

AWS menyediakan beberapa cara untuk mengenkripsi data saat istirahat dan dalam perjalanan. Kami menyertakan fitur-fitur ke layanan kami yang mempermudah enkripsi data. Misalnya, kami telah

menerapkan enkripsi sisi server (SSE) untuk Amazon S3 untuk memudahkan Anda menyimpan data dalam bentuk terenkripsi. Anda juga dapat mengatur seluruh proses HTTPS enkripsi dan dekripsi (umumnya dikenal sebagai SSL terminasi) untuk ditangani oleh Elastic Load Balancing (). ELB

## Respons insiden

Dengan kontrol-kontrol detektif dan preventif yang sangat matang sekalipun, organisasi Anda harus tetap menyiapkan proses untuk merespons dan memitigasi potensi dampak insiden keamanan. Arsitektur beban kerja sangat berpengaruh pada kemampuan tim Anda untuk beroperasi secara efektif selama insiden, untuk mengisolasi atau membatasi sistem, serta untuk memulihkan operasi ke kondisi yang baik. Menetapkan alat dan akses sebelum terjadi insiden keamanan, lalu secara rutin melatih respons insiden melalui game day, akan membantu Anda dalam memverifikasi bahwa arsitektur Anda dapat mendukung investigasi dan pemulihan secara tepat waktu.

Dalam AWS, praktik berikut memfasilitasi respons insiden yang efektif:

- Tersedia pencatatan log mendetail yang berisi konten penting seperti perubahan dan akses file.
- Acara dapat diproses secara otomatis dan meluncurkan alat yang mengotomatiskan respons melalui penggunaan. AWS APIs
- Anda dapat membuat “clean room” (kamar bersih) dan alat terlebih dahulu dengan menggunakan AWS CloudFormation. Dengan demikian, Anda dapat melakukan forensik di lingkungan yang aman dan terisolasi.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keamanan ini.

SEC10: Bagaimana Anda mengantisipasi, menanggapi, dan memulihkan diri dari insiden?

Persiapan sangat penting untuk penyelidikan, respons, dan pemulihan peristiwa keamanan secara tepat waktu dan efektif guna membantu meminimalkan gangguan terhadap organisasi Anda.

Lakukan verifikasi apakah Anda dapat dengan cepat memberikan akses untuk tim keamanan, dan mengotomatiskan isolasi instans serta pencatatan data dan status untuk forensik.

## Keamanan aplikasi

Keamanan aplikasi (AppSec) menjelaskan keseluruhan proses bagaimana Anda mendesain, membangun, dan menguji properti keamanan dari beban kerja yang Anda kembangkan. Anda harus

melatih orang di organisasi Anda dengan baik, memahami karakteristik keamanan dari build Anda dan merilis infrastruktur, serta menggunakan otomatisasi untuk mengidentifkasi masalah keamanan.

Mengadopsi pengujian keamanan aplikasi sebagai bagian reguler dari siklus hidup pengembangan perangkat lunak (SDLC) dan proses pasca rilis membantu memvalidasi bahwa Anda memiliki mekanisme terstruktur untuk mengidentifikasi, memperbaiki, dan mencegah masalah keamanan aplikasi memasuki lingkungan produksi Anda.

Metodologi pengembangan aplikasi Anda harus menyertakan kontrol keamanan saat Anda mendesain, membangun, men-deploy, dan mengoperasikan beban kerja Anda. Saat melakukannya, selaraskan proses demi penurunan kecacatan yang terus-menerus dan meminimalkan utang teknis. Misalnya, menggunakan pemodelan ancaman dalam fase desain membantu Anda menemukan kecacatan desain lebih dini, dan kecacatan ini lebih mudah diatasi serta tidak terlalu mahal dibandingkan menunggu dan memperbaikinya nanti.

Biaya dan kompleksitas untuk menyelesaikan cacat biasanya lebih rendah pada awal Anda berada diSDLC. Cara termudah untuk mengatasi masalah ini adalah mengupayakan agar tidak ada kecacatan dari awal. Oleh karena itu, memulai dengan model ancaman membantu Anda fokus mendapatkan hasil yang tepat dari fase desain. Saat AppSec program Anda matang, Anda dapat meningkatkan jumlah pengujian yang dilakukan menggunakan otomatisasi, meningkatkan kesetiaan umpan balik kepada pembangun, dan mengurangi waktu yang diperlukan untuk tinjauan keamanan. Semua tindakan ini meningkatkan kualitas perangkat lunak yang Anda bangun, dan meningkatkan kecepatan penyiapan fitur untuk masuk ke produksi.

Pedoman implementasi ini berfokus pada empat bidang: organisasi dan budaya, keamanan dari pipeline, keamanan dalam pipeline, dan manajemen dependensi. Setiap area menyediakan seperangkat prinsip yang dapat Anda terapkan dan memberikan end-to-end pandangan tentang bagaimana Anda merancang, mengembangkan, membangun, menyebarkan, dan mengoperasikan beban kerja.

Di AWS, ada sejumlah pendekatan yang dapat Anda gunakan saat menangani program keamanan aplikasi Anda. Beberapa pendekatan ini bergantung pada teknologi, sedangkan yang lain fokus pada orang dan aspek organisasi dari program keamanan aplikasi Anda.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keamanan aplikasi ini.

SEC11: Bagaimana Anda menggabungkan dan memvalidasi properti keamanan aplikasi di seluruh siklus hidup desain, pengembangan, dan penerapan?

Melatih karyawan, menguji menggunakan otomatisasi, memahami dependensi, dan memvalida si karakteristik keamanan alat dan aplikasi akan membantu mengurangi kemungkinan masalah keamanan dalam beban kerja produksi.

## Sumber daya

Lihat sumber daya berikut untuk mempelajari selengkapnya tentang praktik terbaik kami untuk Keamanan.

### Dokumentasi

- AWS Keamanan Cloud
- Kepatuhan AWS
- AWS Blog Keamanan
- Model Kematangan Keamanan AWS

### Laporan resmi

- Pilar Keamanan
- AWS Ikhtisar Keamanan
- AWS Risiko dan Kepatuhan

### Video

- AWS Keamanan Negara Serikat
- Ikhtisar Tanggung Jawab Bersama

# Keandalan

Pilar Keandalan berkenaan dengan kemampuan beban kerja untuk menjalankan fungsinya dengan benar dan konsisten sesuai dengan yang diharapkan. Ini termasuk kemampuan untuk

mengoperasikan dan menguji beban kerja di seluruh siklus hidupnya. Paper ini memberikan panduan praktik terbaik yang mendalam untuk menerapkan beban kerja yang andal. AWS

Pilar keandalan memberikan gambaran umum tentang prinsip, praktik terbaik, dan pertanyaan desain. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keandalan.

Topik



## Prinsip desain

Terdapat lima prinsip desain untuk meraih keandalan di cloud:

- Secara otomatis pulih dari kegagalan: Dengan memantau beban kerja untuk indikator kinerja utama (KPIs), Anda dapat memulai otomatisasi ketika ambang batas dilanggar. Ini KPIs harus menjadi ukuran nilai bisnis, bukan aspek teknis dari operasi layanan. Hal ini menyediakan notifikasi otomatis dan pelacakan kegagalan, serta proses pemulihan otomatis yang menangani atau memperbaiki kegagalan. Dengan otomatisasi yang lebih canggih, antisipasi dan perbaikan kegagalan dapat dilakukan sebelum kegagalan itu terjadi.
- Prosedur pemulihan pengujian: Dalam lingkungan on-premise, pengujian biasanya dijalankan untuk membuktikan bahwa beban kerja bekerja dalam skenario tertentu. Pengujian biasanya tidak digunakan untuk memvalidasi strategi pemulihan. Di cloud, Anda dapat melakukan pengujian tentang bagaimana beban kerja mengalami kegagalan, dan Anda juga dapat memvalidasi prosedur-prosedur pemulihan. Anda dapat menggunakan otomatisasi untuk memicu berbagai kegagalan atau untuk membuat ulang skenario yang mengarah pada kegagalan sebelumnya. Pendekatan ini akan memperlihatkan jalur kegagalan yang dapat diuji dan diperbaiki sebelum sebuah skenario kegagalan benar-benar terjadi, sehingga dapat mengurangi risiko.
- Tingkatkan (skalakan) kapasitas secara horizontal untuk meningkatkan ketersediaan keseluruhan beban kerja: Ganti satu sumber daya besar dengan beberapa sumber daya kecil untuk mengurangi dampak kegagalan tunggal terhadap beban kerja keseluruhan. Distribusikan permintaan ke beberapa sumber daya yang lebih kecil untuk memverifikasi bahwa tidak terdapat titik kegagalan yang sama.

- Berhenti mengira-ngira besar kapasitas: Penyebab umum kegagalan dalam beban kerja on-premise adalah saturasi sumber daya, yaitu ketika permintaan yang ditempatkan di beban kerja melebihi kapasitas beban kerjanya (ini sering kali menjadi sasaran dari serangan denial of service). Di dalam cloud, Anda dapat memantau permintaan dan pemanfaatan beban kerja, serta mengotomatiskan penambahan atau penghapusan sumber daya untuk mempertahankan tingkat efisiensi untuk memenuhi permintaan tanpa kekurangan atau kelebihan penyediaan. Batasan tentunya masih ada, tetapi sebagian kuota dapat dikontrol dan sebagian lainnya dapat dikelola (lihat Kelola Kuota Layanan (Service Quotas) dan Pembatasan).
- Kelola perubahan dengan otomatisasi: Perubahan-perubahan yang dibuat untuk infrastruktur Anda harus dibuat dengan menggunakan otomatisasi. Perubahan yang harus dikelola mencakup perubahan pada otomatisasi, yang kemudian dapat dilacak dan ditinjau.

## Definisi

Terdapat empat area praktik terbaik untuk keandalan di cloud:

- Fondasi
- Arsitektur beban kerja
- Manajemen perubahan
- Manajemen kegagalan

Untuk mencapai keandalan, Anda harus mengawalinya dengan fondasi — sebuah lingkungan dengan Service Quotas dan topologi jaringan yang mengakomodasi beban kerja. Arsitektur beban kerja sistem terdistribusi harus didesain untuk dapat mencegah dan memitigasi kegagalan. Beban kerja harus menangani perubahan-perubahan yang terjadi dalam permintaan dan persyaratan, serta harus didesain untuk dapat mendeteksi kegagalan dan melakukan pemulihan mandiri secara otomatis.

## Praktik terbaik

Topik

## Fondasi

Persyaratan mendasar memiliki cakupan yang lebih luas dari satu beban kerja atau proyek. Sebelum merancang sistem apa pun, persyaratan mendasar yang memengaruhi keandalan harus diterapkan. Misalnya, Anda harus memiliki bandwidth jaringan yang memadai untuk pusat data Anda.

Dengan AWS, sebagian besar persyaratan dasar ini sudah dimasukkan atau dapat ditangani sesuai kebutuhan. Cloud dirancang untuk hampir tidak terbatas, jadi itu adalah tanggung jawab AWS untuk memenuhi kebutuhan jaringan dan kapasitas komputasi yang memadai, memungkinkan Anda untuk mengubah ukuran sumber daya dan alokasi sesuai permintaan.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keandalan. (Untuk melihat daftar pertanyaan keandalan dan praktik terbaik, buka Lampiran.).

### REL1: Bagaimana Anda mengelola Service Quotas dan kendala?

Untuk arsitektur beban kerja berbasis cloud, ada Service Quotas (kuota layanan) (yang juga disebut sebagai batas layanan). Kuota ini ada untuk mencegah penyediaan sumber daya lebih dari yang Anda butuhkan secara tidak sengaja dan untuk membatasi tarif permintaan pada API operasi sehingga dapat melindungi layanan dari penyalahgunaan. Ada juga kendala-k endala sumber daya, misalnya, laju Anda dapat mendorong bit di kabel serat optik, atau jumlah penyimpanan di diska secara fisik.

### REL2: Bagaimana Anda merencanakan topologi jaringan Anda?

Sering kali beban kerja ada di beberapa lingkungan. Ini termasuk beberapa lingkungan cloud (baik yang dapat diakses publik maupun privat) dan kemungkinan infrastruktur pusat data Anda yang ada sekarang. Rencana harus mencakup pertimbangan jaringan seperti konektivitas di dalam dan antarsistem, pengelolaan alamat IP publik, pengelolaan alamat IP privat, dan resolusi nama domain.

## Arsitektur beban kerja

Beban kerja yang andal dimulai dengan desain perangkat lunak dan infrastruktur yang diputuskan sejak awal. Pilihan arsitektur Anda akan memengaruhi perilaku beban kerja Anda di semua pilar Well-Architected. Untuk keandalan, terdapat beberapa pola tertentu yang harus diikuti.

Dengan AWS, pengembang beban kerja memiliki pilihan bahasa dan teknologi untuk digunakan. AWS SDKsmenghilangkan kompleksitas pengkodean dengan menyediakan layanan khusus bahasaAPIs. AWS IniSDKs, ditambah pilihan bahasa, memungkinkan pengembang untuk menerapkan praktik terbaik keandalan yang tercantum di sini. Para developer juga dapat membaca dan belajar dari cara Amazon membangun dan mengoperasikan perangkat lunak, di dalam Amazon Builders' Library.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keandalan.

### REL3: Bagaimana Anda mendesain arsitektur layanan beban kerja Anda?

Bangun beban kerja yang sangat skalabel dan andal menggunakan arsitektur berorientasi layanan (SOA) atau arsitektur layanan mikro. Arsitektur berorientasi layanan (SOA) adalah praktik membuat komponen perangkat lunak dapat digunakan kembali melalui antarmuka layanan. Arsitektur layanan mikro melangkah lebih jauh untuk membuat komponen menjadi lebih kecil dan lebih sederhana.

### REL4: Bagaimana Anda merancang interaksi dalam sistem terdistribusi untuk mencegah kegagalan?

Sistem terdistribusi mengandalkan jaringan komunikasi untuk membuat interkoneksi komponen, seperti server atau layanan. Beban kerja Anda harus beroperasi secara andal terlepas latensi atau hilangnya data yang terjadi di jaringan-jaringan ini. Komponen dari sistem terdistribusi harus beroperasi dengan cara yang tidak secara negatif memengaruhi beban kerja atau komponen-komponen lain. Praktik terbaik ini mencegah kegagalan dan meningkatkan waktu rata-rata antara kegagalan (MTBF).

### REL5: Bagaimana Anda merancang interaksi dalam sistem terdistribusi untuk mengurangi atau menahan kegagalan?

Sistem terdistribusi mengandalkan jaringan komunikasi untuk membuat interkoneksi komponen (seperti server atau layanan). Beban kerja Anda harus beroperasi secara andal terlepas latensi atau hilangnya data pada jaringan-jaringan ini. Komponen dari sistem terdistribusi harus beroperasi dengan cara yang tidak secara negatif memengaruhi beban kerja atau komponen-komponen lain. Berbagai praktik terbaik ini memungkinkan beban kerja bertahan dari stres atau kegagalan, lebih

REL5: Bagaimana Anda merancang interaksi dalam sistem terdistribusi untuk mengurangi atau menahan kegagalan?

cepat pulih darinya, dan memitigasi dampak gangguan tersebut. Hasilnya ditingkatkan waktu rata-rata untuk pemulihan (MTTR).

## Manajemen perubahan

Perubahan beban kerja atau lingkungannya harus diantisipasi dan diakomodasi guna mencapai operasi beban kerja yang andal. Perubahan ini mencakup semua hal yang terjadi pada beban kerja seperti lonjakan permintaan, dan juga perubahan-perubahan dari dalam, seperti deployment fitur dan patch keamanan.

Dengan menggunakan AWS, Anda dapat memantau perilaku beban kerja dan mengotomatiskan respons. KPIs Misalnya, beban kerja dapat memasukkan server tambahan seiring dengan bertambahnya pengguna dalam beban kerja. Anda dapat mengatur pemberian izin terkait siapa yang dapat membuat perubahan beban kerja dan mengaudit riwayat perubahan ini.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keandalan.

REL6: Bagaimana Anda memantau sumber daya beban kerja?

Log dan metrik merupakan alat yang luar biasa untuk mendapatkan wawasan mengenai kondisi beban kerja Anda. Anda dapat mengonfigurasikan beban kerja Anda untuk memantau log dan metrik serta mengirimkan notifikasi ketika ambang batas terlampaui atau ada peristiwa signifikan yang terjadi. Pemantauan memungkinkan beban kerja Anda untuk mengenali ketika ambang batas kinerja rendah terlampaui atau ada kegagalan yang terjadi, sehingga pemulihan dapat terjadi secara otomatis untuk menanggapinya.

REL7: Bagaimana Anda merancang beban kerja Anda untuk beradaptasi dengan perubahan permintaan?

Beban kerja yang dapat diskalakan memberikan elastisitas untuk menambahkan atau mengeluar kan sumber daya secara otomatis sehingga sangat sesuai dengan permintaan saat ini pada titik waktu tertentu.

### REL8: Bagaimana Anda menerapkan perubahan?

Perubahan terkontrol diperlukan untuk melakukan deployment fungsionalitas baru, dan untuk memverifikasi bahwa beban kerja dan lingkungan operasi menjalankan perangkat lunak yang dikenal dan dapat di-patch atau diganti dengan cara yang dapat diprediksi. Jika perubahan-perubahan ini tidak terkontrol, maka akan sulit untuk memprediksi efek dari perubahan-perubahan tersebut, atau untuk mengatasi masalah yang timbul sebagai akibatnya.

Ketika Anda merancang beban kerja agar otomatis menambahkan dan menghapus sumber daya sebagai respons atas perubahan permintaan, ini tidak hanya meningkatkan keandalan, tetapi juga memvalidasi bahwa keberhasilan bisnis tidak akan menambah beban. Dengan pemantauan di tempat, tim Anda akan secara otomatis diperingatkan ketika KPIs menyimpang dari norma yang diharapkan. Pencatatan otomatis log perubahan lingkungan memperkenankan Anda mengaudit dan dengan cepat mengidentifikasi tindakan yang mungkin berdampak terhadap keandalan. Kendali dalam manajemen perubahan akan memastikan bahwa Anda dapat menerapkan aturan untuk memberikan keandalan yang Anda butuhkan.

## Manajemen kegagalan

Dalam sistem apa pun yang memiliki kompleksitas wajar, kegagalan biasanya akan terjadi. Keandalan hanya dapat terwujud jika beban kerja Anda dapat mengidentifikasi kegagalan yang terjadi dan mengambil tindakan untuk menghindari dampaknya terhadap ketersediaan. Beban kerja harus mampu bertahan dari kegagalan serta secara otomatis memperbaiki masalah.

Dengan AWS, Anda dapat memanfaatkan otomatisasi untuk bereaksi terhadap pemantauan data. Misalnya, ketika metrik tertentu melewati ambang batas, Anda dapat menginisiasi tindakan otomatis untuk memperbaiki masalahnya. Selain itu, daripada berupaya untuk mendiagnosis dan memperbaiki sumber daya gagal yang merupakan bagian dari lingkungan produksi, Anda dapat menggantinya dengan yang baru dan melakukan analisis terhadap sumber daya yang gagal tersebut di luar jaringan. Karena cloud memungkinkan Anda menjalankan versi sementara dari keseluruhan sistem dengan harga yang rendah, Anda dapat menggunakan pengujian otomatis untuk memverifikasi proses pemulihan penuh.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keandalan.

### REL9: Bagaimana Anda membuat cadangan data?

Cadangkan data, aplikasi, dan konfigurasi untuk memenuhi kebutuhan Anda untuk tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO).

### REL10: Bagaimana Anda menggunakan isolasi kesalahan untuk melindungi beban kerja Anda?

Batas isolasi kesalahan membatasi efek dari sebuah kegagalan di dalam beban kerja hingga jumlah komponen yang terbatas. Komponen-komponen yang ada di luar batasan-batasan ini tidak terpengaruh oleh kegagalan tersebut. Menggunakan beberapa batasan isolasi kesalahan, Anda dapat membatasi dampak pada beban kerja Anda.

### REL11: Bagaimana Anda mendesain beban kerja Anda untuk menahan kegagalan komponen?

Beban kerja dengan persyaratan ketersediaan tinggi dan waktu rata-rata yang rendah untuk pemulihan (MTTR) harus dirancang untuk ketahanan.

### REL12: Bagaimana Anda menguji keandalan?

Setelah Anda mendesain beban kerja Anda agar tangguh terhadap tekanan produksi, pengujian adalah satu-satunya cara untuk memverifikasi bahwa beban kerja akan beroperasi sesuai desain, dan memberikan ketangguhan yang Anda harapkan.

### REL13: Bagaimana Anda merencanakan pemulihan bencana (DR)?

Memiliki cadangan dan komponen beban kerja berlebih adalah permulaan dari strategi DR Anda. RTOdan RPO merupakan tujuan Anda untuk memulihkan beban kerja Anda. Tetapkan ini berdasarkan kebutuhan bisnis. Implementasikan sebuah strategi untuk memenuhi tujuan-tujuan ini, sambil mempertimbangkan lokasi dan fungsi data dan sumber daya beban kerja. Probabilitas gangguan dan biaya pemulihan juga merupakan faktor penting yang akan membantu menginfor masikan nilai bisnis dari penyediaan pemulihan bencana untuk beban kerja.

Cadangkan data dan uji file cadangan Anda secara rutin untuk memastikan bahwa Anda dapat melakukan pemulihan dari kesalahan fisik dan logis. Kunci untuk mengelola kegagalan adalah pengujian beban kerja secara rutin dan otomatis dengan cara menyebabkan kegagalan, kemudian mengamati bagaimana pemulihan dilakukan. Lakukan hal ini secara terjadwal serta pastikan bahwa pengujian serupa juga dilakukan setelah perubahan beban kerja yang signifikan. Lacak secara aktifKPIs, dan juga tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO), untuk menilai ketahanan beban kerja (terutama dalam skenario pengujian kegagalan). KPIsPelacakan akan membantu Anda mengidentifikasi dan mengurangi satu titik kegagalan. Sasarannya adalah untuk menguji secara keseluruhan proses pemulihan beban kerja Anda sehingga Anda yakin bahwa Anda dapat memulihkan semua data dan terus melayani pelanggan, bahkan saat menghadapi masalah yang berlanjut. Proses pemulihan Anda harus dijalankan dengan baik sebagaimana proses produksi normal Anda.

## Sumber daya

Lihat referensi berikut untuk mempelajari selengkapnya tentang praktik terbaik kami untuk Keandalan.

### Dokumentasi

- Dokumentasi AWS
- AWS Infrastruktur Global
- AWS Auto Scaling: Cara Kerja Penskalaan
- Apa itu AWS Backup?

### Laporan resmi

- Pilar Keandalan: AWS Well-Architected
- Menerapkan Layanan Mikro pada AWS

# Efisiensi kinerja

Pilar efisiensi kinerja mencakup kemampuan untuk menggunakan sumber daya cloud secara efisien untuk memenuhi persyaratan-persyaratan kinerja, dan untuk mempertahankan efisiensi tersebut seiring dengan perubahan permintaan dan perkembangan teknologi yang terjadi.

Pilar efisiensi kinerja memberikan gambaran umum tentang prinsip, praktik terbaik, dan pertanyaan desain. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Efisiensi Kinerja.

Topik



## Prinsip desain

Ada lima prinsip desain untuk meraih efisiensi kinerja di cloud:

- Buat teknologi canggih dapat diakses oleh lebih banyak orang: Buat penerapan teknologi canggih lebih mudah dengan mendelegasikan tugas-tugas kompleks kepada penyedia layanan cloud Anda. Daripada bertanya kepada tim IT Anda tentang hosting dan menjalankan teknologi baru, manfaatkan teknologi sebagai layanan. Misalnya, Tidak ada SQL database, transcoding media, dan pembelajaran mesin adalah semua teknologi yang membutuhkan keahlian khusus. Di cloud, teknologi ini menjadi layanan yang dapat digunakan tim Anda, sehingga mereka dapat berfokus pada pengembangan produk, bukan penyediaan dan manajemen sumber daya.
- Menjadi global dalam hitungan menit: Menyebarkan beban kerja Anda di beberapa AWS Wilayah di seluruh dunia memungkinkan Anda memberikan latensi yang lebih rendah dan pengalaman yang lebih baik bagi pelanggan Anda dengan biaya minimal.
- Gunakan arsitektur nirserver: Dengan arsitektur nirserver, Anda tidak perlu menjalankan dan memelihara server fisik untuk aktivitas komputasi tradisional. Misalnya, layanan penyimpanan nirserver dapat bertindak sebagai situs web statis (tanpa memerlukan server web) dan layanan peristiwa dapat melakukan hosting kode. Dengan demikian, beban operasional untuk mengelola server fisik tidak lagi ada, dan biaya transaksional berkurang karena layanan terkelola dioperasikan pada skala cloud.
- Bereksperimen lebih sering: Dengan sumber daya virtual yang dapat diotomatiskan, Anda dapat melakukan pengujian komparatif dengan cepat menggunakan jenis-jenis instans, penyimpanan, atau konfigurasi yang berbeda-beda.
- Pertimbangkan kesesuaian dengan mekanisme kerja (simpati mekanis): Pahami bagaimana layanan cloud digunakan dan selalu gunakan pendekatan teknologi yang sesuai dengan tujuan

beban kerja Anda. Misalnya, pertimbangkan pola akses data saat Anda memilih pendekatan basis data atau penyimpanan.

## Definisi

Berikut lima area praktik terbaik untuk efisiensi kinerja di cloud:

- Pemilihan arsitektur
- Komputasi dan perangkat keras
- Manajemen data
- Jaringan dan Pengiriman Konten
- Proses dan budaya

Gunakan pendekatan yang didorong data untuk membangun arsitektur dengan kinerja tinggi. Kumpulkan data tentang semua aspek arsitektur, dari desain tingkat tinggi hingga pemilihan dan konfigurasi jenis sumber daya.

Meninjau pilihan Anda secara teratur memvalidasi bahwa Anda memanfaatkan Cloud yang terus berkembang. AWS Pemantauan akan memastikan Anda mengetahui adanya penyimpangan dari kinerja yang diharapkan. Buat kompensasi dalam arsitektur untuk meningkatkan kinerja, seperti menggunakan kompresi atau caching, atau persyaratan konsistensi yang lebih fleksibel.

## Praktik terbaik

Topik

## Pemilihan arsitektur

Solusi yang optimal bervariasi untuk beban kerja tertentu, dan solusi sering kali menggabungkan beberapa pendekatan. Beban kerja yang dirancang dengan baik menggunakan beberapa solusi dan memungkinkan berbagai fitur guna meningkatkan kinerja.

AWS sumber daya tersedia dalam berbagai jenis dan konfigurasi, yang membuatnya lebih mudah untuk menemukan pendekatan yang sesuai dengan kebutuhan Anda. Anda juga dapat menemukan opsi yang tidak mudah dicapai dengan infrastruktur on-premise. Misalnya, layanan terkelola seperti Amazon DynamoDB menyediakan database SQL No yang dikelola sepenuhnya dengan latensi milidetik satu digit pada skala apa pun.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk efisiensi kinerja. (Untuk melihat daftar pertanyaan dan praktik terbaik mengenai efisiensi kinerja, buka Lampiran.).

PERF1: Bagaimana Anda memilih sumber daya cloud dan pola arsitektur yang sesuai untuk beban kerja Anda?

Sering kali, beberapa pendekatan diperlukan untuk performa yang lebih efektif di semua beban kerja. Sistem yang Well-Architected menggunakan beberapa solusi dan fitur untuk meningkatkan kinerja.

## Komputasi dan perangkat keras

Pilihan komputasi yang optimal untuk beban kerja tertentu bervariasi berdasarkan desain aplikasi, pola penggunaan, dan pengaturan konfigurasi. Arsitektur dapat menggunakan pilihan komputasi yang berbeda untuk berbagai komponen, dan memungkinkan fitur yang berbeda untuk meningkatkan kinerja. Memilih pilihan komputasi yang salah untuk arsitektur dapat menyebabkan efisiensi kinerja menjadi lebih rendah.

Dalam AWS, komputasi tersedia dalam tiga bentuk: instance, wadah, dan fungsi:

- Instans adalah server virtual, memungkinkan Anda untuk mengubah kemampuan mereka dengan tombol atau panggilan. API Karena keputusan sumber daya di cloud tidaklah tetap, Anda dapat bereksperimen dengan jenis server yang berbeda-beda. Pada AWS, instance server virtual ini datang dalam keluarga dan ukuran yang berbeda, dan mereka menawarkan berbagai kemampuan, termasuk solid-state drive (SSDs) dan unit pemrosesan grafis (). GPUs

- Container adalah metode virtualisasi sistem operasi yang memungkinkan Anda menjalankan aplikasi dan dependensinya dalam proses yang terisolasi sumber daya. AWS Fargate adalah komputasi tanpa server untuk kontainer atau Amazon EC2 dapat digunakan jika Anda memerlukan kontrol atas penginstalan, konfigurasi, dan pengelolaan lingkungan komputasi Anda. Anda juga dapat memilih dari beberapa platform orkestrasi kontainer: Amazon Elastic Container Service (ECS) atau Amazon Elastic Kubernetes Service (). EKS
- Fungsi mengabstraksikan lingkungan pelaksanaan dari kode yang ingin Anda jalankan. Misalnya, AWS Lambda memungkinkan Anda untuk menjalankan kode tanpa menjalankan instance.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk efisiensi kinerja.

PERF2: Bagaimana Anda memilih dan menggunakan sumber daya komputasi dalam beban kerja Anda?

Solusi komputasi yang lebih efisien untuk sebuah beban kerja bervariasi berdasarkan desain aplikasi, pola penggunaan, dan pengaturan konfigurasi. Arsitektur dapat menggunakan beberapa solusi komputasi untuk beragam komponen, dan mengaktifkan fitur lain untuk meningkatkan performa. Memilih solusi komputasi yang salah untuk arsitektur dapat mengakibatkan efisiensi performa yang lebih rendah.

## Manajemen data

Solusi manajemen data yang optimal untuk sistem tertentu bervariasi berdasarkan jenis tipe data (blok, file, atau objek), pola akses (acak atau sekuensial), throughput yang diperlukan, frekuensi akses (online, offline, arsip), frekuensi pembaruan (, dinamis)WORM, dan kendala ketersediaan dan daya tahan. Beban kerja Well-Architected menggunakan penyimpanan data yang dibuat khusus yang memungkinkan berbagai fitur untuk meningkatkan kinerja.

Dalam AWS, penyimpanan tersedia dalam tiga bentuk: objek, blok, dan file:

- Penyimpanan objek menyediakan platform yang tahan lama dan dapat diskalakan agar data terkait konten yang dibuat pengguna, arsip aktif, komputasi nirserver, penyimpanan Big Data, atau pencadangan dan pemulihan dapat diakses dari lokasi internet mana pun. Amazon Simple Storage Service (Amazon S3) adalah layanan penyimpanan objek yang menawarkan skalabilitas, ketersediaan data, keamanan, dan kinerja terdepan di industri. Amazon S3 dirancang untuk memberikan daya tahan 99,999999999 persen (skenario 11 angka 9) dan menyimpan data jutaan aplikasi untuk perusahaan di seluruh dunia.

- Penyimpanan blok menyediakan penyimpanan blok latensi rendah yang sangat tersedia, konsisten, untuk setiap host virtual dan analog dengan penyimpanan yang terpasang langsung (DAS) atau Jaringan Area Penyimpanan (). SAN Amazon Elastic Block Store (AmazonEBS) dirancang untuk beban kerja yang memerlukan penyimpanan persisten yang dapat diakses oleh EC2 instans yang membantu Anda menyetel aplikasi dengan kapasitas penyimpanan, kinerja, dan biaya yang tepat.
- Penyimpanan file menyediakan akses ke sistem file bersama di seluruh sistem. Solusi penyimpanan file seperti Amazon Elastic File System (AmazonEFS) ideal untuk kasus penggunaan seperti repositori konten besar, lingkungan pengembangan, toko media, atau direktori rumah pengguna. Amazon FSx membuatnya efisien dan hemat biaya untuk meluncurkan dan menjalankan sistem file populer sehingga Anda dapat memanfaatkan rangkaian fitur yang kaya dan kinerja cepat dari sistem file open source dan berlisensi komersial yang banyak digunakan.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk efisiensi kinerja.

PERF3: Bagaimana Anda menyimpan, mengelola, dan mengakses data dalam beban kerja Anda?

Solusi penyimpanan yang lebih efisien untuk suatu sistem bervariasi berdasarkan jenis operasi akses (blok, file, atau objek), pola akses (acak atau sekuensial), throughput yang diperlukan, frekuensi akses (online, offline, arsip), frekuensi pembaruan (, dinamis)WORM, dan kendala ketersediaan dan daya tahan. Sistem yang Well-architected menggunakan beberapa solusi penyimpanan dan mengaktifkan fitur berbeda yang meningkatkan performa dan menggunakan sumber daya secara efisien.

## Jaringan dan Pengiriman Konten

Solusi jaringan optimal untuk beban kerja bervariasi berdasarkan latensi, persyaratan throughput, jitter, dan bandwith. Batas fisik, seperti sumber daya on-premise atau pengguna, menentukan opsi lokasi. Batas-batas ini dapat diimbangi dengan penempatan sumber daya atau lokasi edge.

Pada AWS, jaringan divirtualisasi dan tersedia dalam sejumlah jenis dan konfigurasi yang berbeda. Ini membuatnya lebih mudah untuk mencocokkan kebutuhan jaringan Anda. AWS menawarkan fitur produk (misalnya, Jaringan yang Ditingkatkan, instans yang dioptimalkan EC2 jaringan Amazon, akselerasi transfer Amazon S3, dan CloudFront Amazon dinamis) untuk mengoptimalkan lalu lintas jaringan. AWS juga menawarkan fitur jaringan (misalnya, perutean latensi Amazon Route 53, VPC

titik akhir Amazon AWS Direct Connect, dan AWS Global Accelerator) untuk mengurangi jarak jaringan atau jitter.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk efisiensi kinerja.

PERF4: Bagaimana Anda memilih dan mengkonfigurasi sumber daya jaringan dalam beban kerja Anda?

Pertanyaan ini mencakup panduan dan praktik terbaik untuk merancang, mengonfigurasi, dan mengoperasikan jejaring yang efisien dan solusi penyampaian konten di cloud.

## Proses dan budaya

Saat merancang beban kerja, ada prinsip dan praktik yang dapat Anda adopsi untuk membantu Anda menjalankan beban kerja cloud berkinerja tinggi yang efisien dengan lebih baik. Untuk mengadopsi budaya yang mendorong efisiensi kinerja beban kerja cloud, pertimbangkan prinsip dan praktik utama berikut.

Pertimbangkan prinsip-prinsip utama berikut untuk membangun budaya ini:

- Infrastruktur sebagai kode: Tentukan infrastruktur Anda sebagai kode menggunakan pendekatan seperti AWS CloudFormation templat. Penggunaan templat memungkinkan Anda untuk menempatkan infrastruktur di kontrol sumber bersama dengan konfigurasi dan kode aplikasi Anda. Ini memungkinkan Anda untuk menerapkan praktik yang sama yang Anda gunakan untuk mengembangkan perangkat lunak di infrastruktur Anda sehingga Anda dapat mengulang dengan cepat.
- Pipeline deployment: Gunakan pipeline deployment yang berkelanjutan/terintegrasi terus-menerus (CI/CD) (misalnya, repositori kode sumber, sistem pembangunan, deployment, dan otomatisasi pengujian) untuk melakukan deployment infrastruktur Anda. Ini memungkinkan Anda untuk melakukan deployment dengan cara yang dapat diulang, konsisten, dan murah saat Anda melakukan pengulangan.
- Metrik yang terdefinisi dengan baik: Siapkan dan pantau metrik untuk menangkap indikator kinerja utama (). KPIs Kami menyarankan Anda menggunakan metrik teknis dan metrik bisnis. Untuk situs web atau aplikasi seluler, metrik utama adalah menangkap time-to-first-byte atau merender. Metrik lain yang umumnya berlaku antara lain, hitungan thread, laju pengumpulan sampah, dan keadaan tunggu. Metrik bisnis, seperti biaya kumulatif agregat per permintaan, dapat memberikan peringatan kepada Anda tentang berbagai cara untuk menghemat biaya. Pertimbangkan dengan

hati-hati bagaimana Anda akan menafsirkan metrik. Misalnya, Anda dapat memilih nilai maksimum atau persentil 99 dan bukannya nilai rata-rata.

- Jalankan tes kinerja secara otomatis: Sebagai bagian dari proses deployment Anda, mulai tes kinerja secara otomatis setelah tes yang lebih cepat berhasil dijalankan. Otomatisasi harus menciptakan lingkungan baru, menyiapkan kondisi awal seperti data uji, kemudian jalankan serangkaian uji beban dan tolok ukur. Hasil dari pengujian-pengujian ini harus dikaitkan kembali dengan pembangunan sehingga Anda dapat melacak perubahan performa seiring waktu. Untuk pengujian yang lama, Anda dapat membuat ini sebagai bagian dari alur yang asinkron dari sisa pembangunan. Atau, Anda dapat menjalankan pengujian kinerja semalaman menggunakan Instans EC2 Spot Amazon.
- simulasi pengujian beban kerja: Anda harus membuat serangkaian skenario pengujian yang mereplikasi atau merekam perjalanan pengguna yang direkayasa. Skrip ini harus idempoten dan tidak dipasangkan, dan Anda mungkin harus menyertakan skrip prapemanasan untuk menghasilkan hasil yang valid. Sejauh dapat dilakukan, skrip pengujian Anda harus mereplikasi perilaku penggunaan dalam produksi. Anda dapat menggunakan perangkat lunak atau solusi software-as-a-service (SaaS) untuk menghasilkan beban. Pertimbangkan untuk menggunakan solusi AWS Marketplace dan Instans Spot — yang merupakan cara yang hemat untuk simulasi pengujian beban kerja.
- Visibilitas kinerja: Metrik utama harus dapat dilihat oleh tim Anda, khususnya metrik untuk setiap versi yang dibangun. Ini memungkinkan Anda untuk melihat setiap tren positif atau negatif yang signifikan seiring waktu. Anda juga harus menampilkan metrik atas jumlah kesalahan atau pengecualian untuk memastikan Anda menguji sistem yang berfungsi.
- Visualisasi: Gunakan teknik visualisasi yang membuat jelas di mana terjadi masalah performa, hotspot, keadaan tunggu, atau penggunaan rendah. Lapisi diagram arsitektur dengan metrik performa — kode atau grafik panggilan dapat membantu mengidentifikasi masalah dengan cepat.
- Tinjau proses secara rutin: Arsitektur dengan performa buruk biasanya merupakan akibat dari tidak adanya proses peninjauan performa, atau proses peninjauan performa yang bermasalah. Jika arsitektur Anda memiliki performa buruk, implementasi proses peninjauan performa memungkinkan Anda untuk mendorong peningkatan berulang.
- Optimalisasi terus-menerus: Adopsi budaya untuk terus mengoptimalkan efisiensi kinerja beban kerja cloud Anda.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk efisiensi kinerja.

PERF5: Proses apa yang Anda gunakan untuk mendukung efisiensi kinerja yang lebih besar untuk beban kerja Anda?

Saat merancang beban kerja, ada prinsip dan praktik yang dapat Anda adopsi untuk membantu Anda menjalankan beban kerja cloud berkinerja tinggi yang efisien dengan lebih baik. Untuk mengadopsi budaya yang mendorong efisiensi kinerja beban kerja cloud, pertimbangkan prinsip dan praktik utama berikut.

## Sumber daya

Lihat referensi berikut untuk mempelajari selengkapnya tentang praktik terbaik kami untuk Efisiensi Kinerja.

### Dokumentasi

- Optimalisasi Performa Amazon S3
- Kinerja EBS Volume Amazon

### Laporan resmi

- Pilar Efisiensi Performa

### Video

- AWS re: Ciptakan 2019: EC2 Yayasan Amazon (-R2) CMP211
- AWS Re:invent 2019: Sesi kepemimpinan: Status penyimpanan serikat pekerja (01-L) STG2
- AWS Re:invent 2019: Sesi kepemimpinan: database yang dibuat khusus ( AWS 09-L) DAT2
- AWS re:invent 2019: Konektivitas ke AWS dan arsitektur AWS jaringan hybrid (-R1) NET317
- AWS Re:invent 2019: Memberdayakan EC2 Amazon generasi berikutnya: Menyelam jauh ke dalam sistem Nitro (03-R2) CMP3
- AWS Re:invent 2019: Menskalakan hingga 10 juta pengguna pertama Anda (-R) ARC211

# Optimalisasi Biaya

Pilar Optimalisasi Biaya mencakup kemampuan untuk menjalankan sistem guna menghadirkan nilai bisnis dengan harga yang paling rendah.

Pilar optimalisasi biaya menyediakan gambaran umum tentang prinsip, praktik terbaik, dan pertanyaan desain. Anda dapat menemukan panduan preskriptif tentang implementasi di Dokumen Whitepaper Pilar Optimalisasi Biaya.

Topik



## Prinsip desain

Terdapat lima prinsip desain untuk optimalisasi biaya di cloud:

- Implementasikan Manajemen Keuangan Cloud: Untuk mencapai keberhasilan keuangan dan mempercepat realisasi nilai bisnis di cloud, Anda perlu berinvestasi dalam Manajemen Keuangan Cloud dan Optimisasi Biaya. Organisasi Anda perlu mendedikasikan waktu dan sumber daya untuk mengembangkan kemampuan dalam domain teknologi baru ini dan dalam manajemen penggunaan. Serupa dengan kemampuan Keamanan atau Keunggulan Operasi Anda, Anda perlu mengembangkan kemampuan melalui pengembangan pengetahuan, program, sumber daya, dan proses guna menjadi organisasi yang hemat biaya.
- Adopsi model konsumsi: Bayar hanya untuk sumber daya komputasi yang Anda perlukan dan tingkatkan atau turunkan penggunaan tergantung kebutuhan bisnis Anda, tidak dengan prakiraan yang rumit. Misalnya, lingkungan pengembangan dan pengujian umumnya hanya digunakan selama delapan jam sehari selama jam operasional kerja. Anda dapat menghentikan sumber daya ini ketika tidak digunakan untuk mendapatkan potensi penghematan biaya sebesar 75% (40 jam dibandingkan 168 jam).
- Ukur efisiensi keseluruhan: Ukur output bisnis beban kerja serta biaya terkait penyediaannya. Gunakan pengukuran ini untuk mengetahui keuntungan yang Anda dapatkan dari peningkatan output dan pengurangan biaya.

- Berhenti menghabiskan uang untuk angkat berat yang tidak berdiferensiasi: AWS lakukan pengangkatan berat operasi pusat data seperti racking, stacking, dan powerering server. Ia juga menghilangkan beban operasional dalam mengelola sistem operasi dan aplikasi dengan memanfaatkan layanan terkelola. Hal ini akan memungkinkan Anda berkonsentrasi pada pelanggan dan proyek bisnis Anda, bukan pada infrastruktur IT.
- Analisis dan atribusikan pengeluaran: Cloud menyederhanakan identifikasi penggunaan dan biaya sistem secara akurat, sehingga memungkinkan atribusi biaya IT secara transparan ke pemilik beban kerja masing-masing. Ini membantu mengukur laba atas investasi (ROI) dan memberi pemilik beban kerja kesempatan untuk mengoptimalkan sumber daya mereka dan mengurangi biaya.

## Definisi

Terdapat lima area praktik terbaik untuk optimasi biaya di cloud:

- Mempraktikkan Manajemen Keuangan Cloud
- Kesadaran akan penggunaan dan pengeluaran
- Sumber daya hemat biaya
- Kelola sumber daya pasokan dan permintaan
- Pengoptimalan dari waktu ke waktu

Seperti pilar lain dalam Well-Architected Framework, ada perdagangan yang perlu dipertimbangkan, misalnya, apakah akan mengoptimalkan untuk atau untuk biaya. speed-to-market Di beberapa kasus, akan lebih efisien jika Anda mengoptimalkan untuk kecepatan, memasuki pasar dengan cepat, mengirimkan fitur baru, atau memenuhi tenggat, alih-alih berinvestasi untuk optimisasi biaya di awal. Keputusan desain terkadang disetir oleh sikap terburu-buru, bukan oleh data, dan selalu ada godaan untuk melakukan kompensasi berlebihan “untuk jaga-jaga”, alih-alih meluangkan waktu untuk membandingkan opsi-opsi deployment dengan biaya paling optimal. Hal ini dapat berakibat pada deployment dengan pengadaan yang berlebihan dan kurang optimal. Namun, ini adalah pilihan yang wajar ketika Anda harus melakukan “angkat dan geser” sumber daya dari lingkungan on-premise ke cloud lalu melakukan optimisasi setelahnya. Menginvestasikan upaya yang cukup dalam strategi optimisasi biaya di awal memungkinkan Anda lebih mudah mewujudkan manfaat cloud pada aspek ekonomi dengan mencapai kepatuhan yang konsisten terhadap praktik terbaik dan menghindari pengadaan berlebihan yang tidak diperlukan: Bagian berikutnya menyediakan teknik dan praktik

terbaik untuk melakukan implementasi awal dan berkelanjutan terhadap Manajemen Keuangan Cloud dan optimalisasi biaya untuk beban kerja Anda.

## Praktik terbaik

Topik



### Mempraktikkan Manajemen Keuangan Cloud

Dengan adopsi cloud, tim teknologi berinovasi lebih cepat karena siklus deployment infrastruktur, pengadaan, dan persetujuan yang lebih pendek. Pendekatan baru ke manajemen keuangan di cloud diperlukan untuk merealisasikan nilai bisnis dan keberhasilan keuangan. Pendekatan ini adalah Manajemen Keuangan Cloud, dan mengembangkan kemampuan di organisasi Anda dengan mengimplementasikan pengembangan pengetahuan, program, sumber daya, dan proses di seluruh organisasi.

Banyak organisasi terdiri dari banyak unit yang berbeda dengan prioritas yang berbeda-beda. Kemampuan untuk menyelaraskan organisasi Anda dengan serangkaian sasaran keuangan yang telah disepakati, dan untuk membekali organisasi Anda dengan mekanisme untuk memenuhi sasaran tersebut, akan menciptakan organisasi yang lebih efisien. Organisasi yang mumpuni akan berinovasi dan membangun lebih cepat, menjadi lebih tangkas, dan menyesuaikan diri dengan faktor-faktor internal atau eksternal apa pun.

Di AWS Anda dapat menggunakan Cost Explorer, dan secara opsional Amazon Athena dan QuickSight Amazon dengan Laporan Biaya dan Penggunaan CUR (), untuk memberikan kesadaran biaya dan penggunaan di seluruh organisasi Anda. AWS Budgets menyediakan notifikasi proaktif untuk biaya dan penggunaan. AWS Blog memberikan informasi tentang layanan dan fitur baru untuk memverifikasi Anda tetap up to date dengan rilis layanan baru.

Pertanyaan berikut berfokus pada semua pertimbangan untuk optimasi biaya ini. (Untuk melihat daftar pertanyaan dan praktik terbaik optimasi biaya, buka Lampiran.).

COST1: Bagaimana Anda menerapkan manajemen keuangan cloud?

Menerapkan Cloud Financial Management membantu organisasi mewujudkan nilai bisnis dan kesuksesan finansial saat mereka mengoptimalkan biaya dan penggunaan serta skalanya AWS.

Saat membangun fungsi pengoptimalan biaya, gunakan anggota dan lengkapi tim dengan ahli CFM dan pengoptimalan biaya. Anggota tim yang ada akan memahami bagaimana fungsi organisasi saat ini dan cara mengimplementasikan peningkatan dengan cepat. Pertimbangkan juga untuk menyertakan orang-orang dengan set keterampilan tambahan atau khusus, seperti analitik dan manajemen proyek.

Ketika mengimplementasikan kesadaran biaya di organisasi Anda, tingkatkan atau kembangkan program dan proses yang sudah ada. Jauh lebih cepat melakukan penambahan ke yang sudah ada daripada membangun proses dan program baru. Dengan begitu, hasil akan dicapai dengan jauh lebih cepat.

## Kesadaran akan penggunaan dan pengeluaran

Peningkatan fleksibilitas dan ketangkasan yang disediakan oleh cloud mendorong inovasi serta pengembangan dan deployment yang cepat. Hal ini akan menurunkan proses manual serta waktu terkait pengadaan infrastruktur on-premise, termasuk identifikasi spesifikasi perangkat keras, negosiasi pengajuan harga, pengelolaan pesanan pembelian, penjadwalan pengiriman, lalu deployment sumber daya. Namun, kemudahan penggunaan dan kapasitas sesuai permintaan yang hampir tanpa batas ini memerlukan cara berpikir baru tentang pengeluaran.

Banyak bisnis terdiri dari beberapa sistem yang dijalankan oleh berbagai tim. Kemampuan untuk mengaitkan biaya sumber daya dengan tiap-tiap organisasi atau pemilik produk mendorong perilaku penggunaan yang efisien dan membantu mengurangi pemborosan. Pengaitan biaya yang akurat mengizinkan Anda mengetahui produk mana yang benar-benar menguntungkan, dan mengizinkan Anda mengambil keputusan yang lebih matang tentang target-target alokasi anggaran.

Di AWS, Anda membuat struktur akun dengan AWS Organizations atau AWS Control Tower, yang menyediakan pemisahan dan membantu alokasi biaya dan penggunaan Anda. Anda juga dapat menggunakan tag sumber daya untuk menerapkan informasi bisnis dan organisasi ke penggunaan dan biaya Anda. Gunakan AWS Cost Explorer untuk visibilitas ke biaya dan penggunaan Anda, atau buat dasbor dan analitik yang disesuaikan dengan Amazon Athena dan Amazon. QuickSight Mengontrol biaya dan penggunaan Anda dilakukan dengan pemberitahuan melalui AWS Anggaran, dan kontrol menggunakan AWS Identity and Access Management (IAM), dan Service Quotas.

Pertanyaan-pertanyaan berikut berfokus pada semua pertimbangan untuk optimasi biaya ini.

### COST2: Bagaimana Anda mengatur penggunaan?

Tetapkan kebijakan dan mekanisme untuk memvalidasi bahwa biaya yang ditimbulkan memang diperlukan untuk mencapai tujuan. Dengan menggunakan checks-and-balances pendekatan, Anda dapat berinovasi tanpa pengeluaran berlebihan.

### COST3: Bagaimana Anda memantau penggunaan dan biaya?

Tetapkan kebijakan dan prosedur untuk memantau dan mengalokasikan biaya Anda dengan tepat. Hal ini memungkinkan Anda mengukur dan meningkatkan efisiensi biaya beban kerja ini.

### COST4: Bagaimana Anda menonaktifkan sumber daya?

Menerapkan kontrol perubahan dan manajemen sumber daya dari awal proyek hingga. end-of-life Hal ini akan memudahkan Anda mematikan sumber daya yang tidak digunakan agar tidak boros.

Anda dapat menggunakan tag alokasi biaya untuk mengelompokkan dan melacak penggunaan dan biaya AWS Anda. Saat Anda menerapkan tag ke AWS sumber daya Anda (seperti EC2 instance atau bucket S3), buat AWS laporan biaya dan penggunaan dengan penggunaan dan tag Anda. Anda dapat memberikan tag yang mewakili kategori organisasi (seperti pusat biaya, nama beban kerja, atau pemilik) untuk mengatur biaya Anda di beberapa layanan.

Verifikasi bahwa Anda menggunakan tingkat detail dan granularitas yang tepat pada pelaporan dan pemantauan biaya dan penggunaan. Untuk wawasan dan tren tingkat tinggi, gunakan tingkat detail harian dengan AWS Cost Explorer. Untuk analisis dan inspeksi yang lebih dalam, gunakan perincian per jam di AWS Cost Explorer, atau Amazon Athena dan Amazon QuickSight dengan Laporan Biaya dan Penggunaan (CUR) dengan perincian per jam.

Menggabungkan sumber daya ber-tag dengan pelacakan siklus hidup entitas (karyawan, proyek) memungkinkan identifikasi sumber daya atau proyek “orphan” (tak terpakai) yang sudah tidak menghasilkan nilai untuk organisasi dan harus menjalani dekomisioning. Anda dapat mengatur notifikasi penagihan untuk memberi tahu Anda tentang prediksi pengeluaran yang berlebihan.

## Sumber daya hemat biaya

Penggunaan instans dan sumber daya yang tepat untuk beban kerja Anda merupakan hal utama dalam penghematan biaya. Misalnya, proses laporan mungkin memerlukan lima jam untuk berjalan di server yang lebih kecil tetapi satu jam untuk berjalan di server yang lebih besar dengan biaya dua kali lipat lebih mahal. Kedua server tersebut memberi Anda hasil yang sama, tetapi seiring waktu, server yang lebih kecil akan memakan biaya yang lebih besar.

Beban kerja yang dirancang dengan baik dengan menggunakan sumber daya yang paling hemat biaya, yang dapat memberikan dampak ekonomi positif yang besar. Anda juga memiliki kesempatan untuk menggunakan layanan terkelola untuk memangkas biaya. Misalnya, alih-alih memelihara server untuk mengirimkan email, Anda dapat menggunakan layanan yang mengenakan biaya per pesan.

AWS menawarkan berbagai opsi harga yang fleksibel dan hemat biaya untuk memperoleh instans dari Amazon EC2 dan layanan lain dengan cara yang lebih efektif sesuai dengan kebutuhan Anda. Instans Sesuai Permintaan akan memungkinkan Anda membayar kapasitas komputasi berdasarkan jam, tanpa memerlukan komitmen minimum. Savings Plans dan Instans Terpesan menawarkan Anda penghematan hingga 75% dari harga Sesuai Permintaan. Dengan Instans Spot, Anda dapat memanfaatkan EC2 kapasitas Amazon yang tidak terpakai dan menawarkan penghematan hingga 90% dari harga Sesuai Permintaan. Instans Spot sesuai di mana sistem dapat mentolerir menggunakan armada server di mana server individu dapat datang dan pergi secara dinamis, seperti server web stateless, pemrosesan batch, atau saat menggunakan HPC dan data besar.

Pemilihan layanan yang tepat juga dapat mengurangi penggunaan dan biaya; seperti CloudFront meminimalkan transfer data, atau mengurangi biaya, seperti memanfaatkan Amazon Aurora di RDS Amazon untuk menghapus biaya lisensi database yang mahal.

Pertanyaan-pertanyaan berikut berfokus pada semua pertimbangan untuk optimasi biaya ini.

### COST5: Bagaimana Anda mengevaluasi biaya ketika Anda memilih layanan?

AmazonEC2, AmazonEBS, dan Amazon S3 adalah layanan blok bangunan AWS . Layanan terkelola, seperti Amazon RDS dan Amazon DynamoDB, adalah layanan tingkat yang lebih tinggi, atau tingkat aplikasi. AWS Dengan memilih blok penyusun dan layanan terkelola yang sesuai, Anda dapat mengoptimalkan biaya beban kerja ini. Contohnya, dengan menggunakan layanan terkelola, Anda dapat mengurangi atau menghilangkan sebagian besar dari biaya tambahan untuk

COST5: Bagaimana Anda mengevaluasi biaya ketika Anda memilih layanan?

administrasi dan operasi, sehingga Anda bebas untuk mengerjakan aplikasi dan aktivitas yang terkait dengan bisnis.

COST6: Bagaimana Anda memenuhi target biaya ketika Anda memilih jenis sumber daya, ukuran dan nomor?

Pastikan Anda memilih jumlah sumber daya dan ukuran sumber daya yang sesuai untuk tugas yang ada. Anda meminimalkan pemborosan dengan memilih jenis, ukuran, dan jumlah yang paling hemat.

COST7: Bagaimana Anda menggunakan model harga untuk mengurangi biaya?

Gunakan model harga yang paling sesuai untuk sumber daya Anda untuk meminimalkan pengeluaran.

COST8: Bagaimana Anda merencanakan biaya transfer data?

Pastikan Anda merencanakan dan memantau biaya transfer data sehingga Anda dapat mengambil keputusan arsitektur untuk meminimalkan biaya. Perubahan arsitektur yang kecil, tetapi efektif dapat secara drastis mengurangi biaya operasional Anda seiring waktu.

Dengan memperhitungkan biaya selama pemilihan layanan, dan menggunakan alat seperti Cost Explorer dan AWS Trusted Advisor untuk meninjau AWS penggunaan Anda secara teratur, Anda dapat secara aktif memantau pemanfaatan Anda dan menyesuaikan penerapan Anda sesuai dengan itu.

## Kelola sumber daya pasokan dan permintaan

Ketika Anda beralih ke cloud, Anda hanya perlu membayar sesuai dengan yang Anda butuhkan. Anda dapat memasok sumber daya sesuai dengan permintaan beban kerja pada saat dibutuhkan, sehingga mengurangi pemborosan biaya dan penyediaan berlebih yang tidak terpakai. Anda juga dapat memodifikasi permintaan, menggunakan throttle, buffer, atau antrean untuk melancarkan

permintaan dan melayaninya dengan sumber daya yang lebih sedikit sehingga biayanya juga menjadi lebih rendah, atau memprosesnya di lain waktu dengan layanan batch.

Di AWS, Anda dapat secara otomatis menyediakan sumber daya agar sesuai dengan permintaan beban kerja. Auto Scaling (penskalaan otomatis) yang menggunakan pendekatan berbasis permintaan atau waktu akan mengizinkan Anda menambahkan dan menghapus sumber daya seperlunya. Jika Anda dapat mengantisipasi perubahan sesuai permintaan, Anda dapat menghemat lebih banyak dana dan memvalidasi bahwa sumber daya Anda sesuai dengan kebutuhan beban kerja. Anda dapat menggunakan Amazon API Gateway untuk menerapkan throttling, atau Amazon SQS untuk menerapkan antrian di beban kerja Anda. Keduanya akan mengizinkan Anda memodifikasi permintaan pada komponen beban kerja Anda.

Pertanyaan berikut berfokus pada semua pertimbangan untuk optimasi biaya ini.

COST9: Bagaimana Anda mengelola permintaan, dan sumber daya pasokan?

Untuk beban kerja yang memiliki pengeluaran dan performa seimbang, pastikan semua beban kerja yang Anda bayar akan digunakan dan hindari tingkat penggunaan instans yang terlalu rendah. Metrik pemanfaatan miring di kedua arah memiliki dampak buruk pada organisasi Anda, baik dalam biaya operasional (kinerja yang menurun karena pemanfaatan yang berlebihan), atau pengeluaran yang terbuang AWS (karena penyediaan berlebihan).

Ketika merancang untuk memodifikasi sumber daya pasokan dan permintaan, pikirkan secara aktif tentang pola-pola penggunaan, waktu yang diperlukan untuk menyediakan sumber daya baru, dan prediktabilitas pola permintaan. Ketika mengelola permintaan, verifikasi bahwa Anda memiliki antrean atau buffer dengan ukuran yang tepat, dan Anda merespons permintaan beban kerja dalam waktu yang diperlukan.

## Pengoptimalan dari waktu ke waktu

Saat AWS merilis layanan dan fitur baru, ini adalah praktik terbaik untuk meninjau keputusan arsitektur Anda yang ada untuk memverifikasi bahwa mereka terus menjadi yang paling hemat biaya. Seiring dengan perubahan kebutuhan, jangan ragu untuk melakukan dekomisioning sumber daya, seluruh layanan, dan sistem yang sudah tidak diperlukan.

Implementasi fitur atau jenis sumber daya baru dapat mengoptimalkan beban kerja Anda secara bertahap, sambil meminimalkan upaya yang diperlukan untuk mengimplementasikan perubahan. Hal ini menyediakan peningkatan berkelanjutan dari segi efisiensi seiring waktu dan akan memastikan

Anda tetap menggunakan teknologi paling mutakhir untuk memangkas biaya operasi. Anda juga dapat mengganti atau menambahkan komponen baru ke beban kerja dengan layanan baru. Hal ini dapat menyediakan peningkatan yang signifikan dari segi efisiensi, sehingga penting untuk secara rutin meninjau beban kerja, dan mengimplementasikan layanan dan fitur baru.

Pertanyaan-pertanyaan berikut berfokus pada semua pertimbangan untuk optimasi biaya ini.

**COST10: Bagaimana Anda mengevaluasi layanan baru?**

Saat AWS merilis layanan dan fitur baru, ini adalah praktik terbaik untuk meninjau keputusan arsitektur Anda yang ada untuk memverifikasi bahwa mereka terus menjadi yang paling hemat biaya.

Ketika meninjau deployment secara rutin, nilailah bagaimana layanan yang lebih baru dapat membantu menghemat dana Anda. Misalnya, Amazon Aurora di Amazon RDS dapat mengurangi biaya untuk database relasional. Penggunaan layanan nirserver seperti Lambda dapat menghilangkan kebutuhan untuk mengoperasikan dan mengelola instans guna menjalankan kode.

**COST11: Bagaimana Anda mengevaluasi biaya usaha?**

Evaluasi biaya usaha untuk operasi di cloud, tinjau operasi cloud yang memakan waktu, dan mengotomatiskannya untuk mengurangi upaya dan biaya manusia dengan mengadopsi AWS layanan terkait, produk pihak ketiga, atau alat khusus.

## Sumber daya

Lihat sumber daya berikut untuk mempelajari selengkapnya tentang praktik terbaik kami untuk Optimasi Biaya.

### Dokumentasi

- Dokumentasi AWS

### Laporan resmi

- Pilar Optimalisasi Biaya

# Keberlanjutan

Pilar Keberlanjutan berfokus pada dampak lingkungan, terutama konsumsi dan efisiensi energi, karena ini merupakan pendorong penting bagi arsitek untuk melandasi tindakan langsung untuk mengurangi penggunaan sumber daya. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keberlanjutan.

Topik



## Prinsip desain

Terdapat enam prinsip desain untuk keberlanjutan di cloud:

- Pahami dampak Anda: Ukur dampak beban kerja cloud Anda dan buat model dampak beban kerja Anda untuk masa mendatang. Sertakan semua sumber dampak, termasuk dampak akibat penggunaan produk Anda oleh pelanggan, serta dampak yang muncul dari penonaktifan dan penghentian produk. Bandingkan output produktif dengan total dampak beban kerja cloud Anda dengan meninjau sumber daya dan emisi yang diperlukan per unit kerja. Gunakan data ini untuk menetapkan indikator kinerja utama (KPIs), mengevaluasi cara untuk meningkatkan produktivitas sekaligus mengurangi dampak, dan memperkirakan dampak perubahan yang diusulkan dari waktu ke waktu.
- Tetapkan tujuan keberlanjutan: Untuk tiap-tiap beban kerja cloud, tetapkan tujuan keberlanjutan jangka panjang seperti mengurangi sumber daya komputasi dan penyimpanan yang diperlukan per transaksi. Modelkan laba atas investasi peningkatan keberlanjutan untuk beban kerja yang ada, dan beri pemilik sumber daya yang mereka perlukan untuk berinvestasi dalam tujuan keberlanjutan. Rencanakan pertumbuhan, dan rancang beban kerja Anda agar pertumbuhan menghasilkan penurunan intensitas dampak yang terukur berdasarkan unit yang tepat, seperti per pengguna atau per transaksi. Tujuan ini membantu Anda mendukung tujuan keberlanjutan yang lebih luas untuk bisnis atau organisasi Anda, mengidentifikasi regresi, dan memprioritaskan area-area dengan potensi perbaikan.
- Maksimalkan pemanfaatan: Sesuaikan ukuran beban kerja dan implementasikan desain yang efisien untuk memverifikasi pemanfaatan yang tinggi dan memaksimalkan efisiensi energi

untuk perangkat keras yang digunakan. Dua host yang berjalan dengan pemanfaatan 30% memiliki efisiensi yang lebih rendah daripada satu host dengan pemanfaatan 60% dikarenakan konsumsi daya dasar per host. Pada saat yang sama, kurangi atau minimalkan sumber daya, pemrosesan, dan penyimpanan yang tidak aktif untuk mengurangi total energi yang diperlukan untuk menjalankan beban kerja Anda.

- Antisipasi dan adopsi penawaran perangkat keras dan perangkat lunak baru yang lebih efisien: Dukung peningkatan hulu yang dilakukan mitra dan pemasok Anda untuk membantu Anda mengurangi dampak beban kerja cloud Anda. Terus pantau dan evaluasi penawaran perangkat keras dan perangkat lunak baru yang lebih efisien. Rancang fleksibilitas untuk mengizinkan pengadopsian teknologi efisien baru secara cepat.
- Gunakan layanan terkelola: Berbagi layanan di seluruh basis pelanggan yang luas dapat membantu memaksimalkan pemanfaatan sumber daya, sehingga mengurangi jumlah infrastruktur yang diperlukan untuk mendukung beban kerja cloud. Misalnya, pelanggan dapat berbagi dampak komponen pusat data umum seperti daya dan jaringan dengan memigrasikan beban kerja ke AWS Cloud dan mengadopsi layanan terkelola, seperti AWS Fargate untuk kontainer tanpa server, di mana AWS beroperasi dalam skala besar dan bertanggung jawab atas operasi mereka yang efisien. Gunakan layanan terkelola yang dapat membantu meminimalkan dampak Anda, seperti memindahkan data yang jarang diakses secara otomatis ke penyimpanan dingin dengan konfigurasi Siklus Hidup Amazon S3 atau Amazon EC2 Auto Scaling untuk menyesuaikan kapasitas guna memenuhi permintaan.
- Kurangi dampak hilir beban kerja cloud Anda: Kurangi jumlah energi atau sumber daya yang diperlukan untuk menggunakan layanan-layanan Anda. Kurangi kebutuhan pelanggan untuk meningkatkan perangkat mereka untuk menggunakan layanan Anda. Uji menggunakan device farm untuk memahami dampak yang diperkirakan dan uji dengan pelanggan untuk memahami dampak riil dari penggunaan layanan Anda.

## Definisi

Terdapat enam area praktik terbaik untuk keberlanjutan di cloud:

- Pemilihan wilayah
- Penyelarasan dengan permintaan
- Perangkat lunak dan arsitektur
- Data
- Perangkat keras dan layanan

- Proses dan budaya

Keberlanjutan di cloud adalah sebuah upaya hampir berkelanjutan yang difokuskan terutama pada pengurangan dan efisiensi energi di semua komponen beban kerja dengan mencapai manfaat maksimum dari sumber daya yang disediakan dan meminimalkan total sumber daya yang diperlukan. Upaya ini bisa mencakup pemilihan bahasa pemrograman yang efisien di awal, adopsi algoritme modern, penggunaan teknik penyimpanan data yang efisien, deployment ke infrastruktur komputasi yang efisien dan terukur dengan tepat, serta meminimalkan kebutuhan perangkat keras pengguna akhir yang berdaya tinggi.

## Praktik terbaik

Topik



### Pemilihan wilayah

Pilihan Wilayah untuk beban kerja Anda sangat memengaruhiKPIs, termasuk kinerja, biaya, dan jejak karbon. Untuk meningkatkan iniKPIs, Anda harus memilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan keberlanjutan.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keberlanjutan ini. (Untuk melihat daftar pertanyaan dan praktik terbaik keberlanjutan, buka Lampiran.)

SUS1: Bagaimana Anda memilih Wilayah untuk beban kerja Anda?

Pilihan Wilayah untuk beban kerja Anda sangat memengaruhiKPIs, termasuk kinerja, biaya, dan jejak karbon. Untuk meningkatkan iniKPIs, Anda harus memilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan keberlanjutan.

## Penyelarasan dengan permintaan

Cara pengguna dan aplikasi menggunakan beban kerja Anda dan sumber daya lainnya dapat membantu Anda mengidentifikasi peningkatan untuk memenuhi tujuan keberlanjutan. Skalakan infrastruktur agar dapat terus sesuai dengan permintaan dan pastikan bahwa Anda hanya menggunakan sumber daya minimum yang diperlukan untuk mendukung para pengguna Anda. Selaraskan tingkat layanan dengan kebutuhan para pelanggan. Posisikan sumber daya guna membatasi jaringan yang diperlukan para pengguna dan aplikasi untuk memakainya. Hapus aset yang tidak digunakan. Bekali anggota tim Anda dengan perangkat yang mendukung kebutuhan mereka dan meminimalkan dampak terhadap keberlanjutan.

Pertanyaan-pertanyaan berikut ini berfokus pada pertimbangan untuk keberlanjutan ini:

SUS2: Bagaimana Anda menyelaraskan sumber daya cloud dengan permintaan Anda?

Cara pengguna dan aplikasi menggunakan beban kerja Anda dan sumber daya lainnya dapat membantu Anda mengidentifikasi peningkatan untuk memenuhi tujuan keberlanjutan. Skalakan infrastruktur agar dapat terus sesuai dengan permintaan dan pastikan bahwa Anda hanya menggunakan sumber daya minimum yang diperlukan untuk mendukung para pengguna Anda. Selaraskan tingkat layanan dengan kebutuhan para pelanggan. Posisikan sumber daya guna membatasi jaringan yang diperlukan para pengguna dan aplikasi untuk memakainya. Hapus aset yang tidak digunakan. Bekali anggota tim Anda dengan perangkat yang mendukung kebutuhan mereka dan meminimalkan dampak terhadap keberlanjutan.

Skalakan infrastruktur dengan beban pengguna: Identifikasi periode pemanfaatan rendah atau nol dan skalakan sumber daya untuk mengurangi kapasitas berlebih dan meningkatkan efisiensi.

Selaraskan SLAs dengan tujuan keberlanjutan: Tentukan dan perbarui perjanjian tingkat layanan (SLAs) seperti ketersediaan atau periode retensi data untuk meminimalkan jumlah sumber daya yang diperlukan untuk mendukung beban kerja Anda sambil terus memenuhi persyaratan bisnis.

Kurangi pembuatan dan pemeliharaan aset tak terpakai: Analisis aset aplikasi (seperti laporan pra-kompilasi, set data, dan gambar statis) serta pola akses aset untuk mengidentifikasi redundansi, pemanfaatan yang terlalu rendah, dan potensi target penonaktifan. Gabungkan aset dengan konten sama yang dihasilkan (misalnya laporan bulanan yang output dan set datanya tumpang tindih atau sama) untuk mengurangi sumber daya yang terbuang karena output ganda. Nonaktifkan aset tidak terpakai (misalnya gambar-gambar produk yang sudah tidak dijual) untuk melepas sumber daya yang dikonsumsi dan mengurangi jumlah sumber daya yang digunakan untuk mendukung beban kerja.

Optimalkan penempatan beban kerja secara geografis untuk lokasi pengguna: Analisis pola akses jaringan untuk mengidentifikasi lokasi geografis tempat pelanggan Anda terhubung. Pilih Wilayah dan layanan yang mengurangi jarak yang harus ditempuh oleh lalu lintas jaringan guna menurunkan total sumber daya jaringan yang diperlukan untuk mendukung beban kerja Anda.

Optimalkan sumber daya anggota tim untuk aktivitas yang dijalankan: Optimalkan sumber daya yang disediakan untuk anggota tim untuk meminimalkan dampak keberlanjutan sambil mendukung kebutuhan mereka. Sebagai contoh, lakukan operasi yang kompleks, seperti rendering dan kompilasi, di desktop cloud bersama dengan tingkat pemanfaatan yang tinggi, bukan di sistem pengguna tunggal berdaya tinggi tetapi dengan pemanfaatan yang rendah.

## Perangkat lunak dan arsitektur

Implementasikan pola untuk melancarkan beban dan mempertahankan penggunaan yang tinggi dan konsisten atas sumber daya yang di-deploy guna meminimalkan sumber daya yang dipakai. Komponen dapat menjadi tidak aktif akibat kurangnya pemakaian, karena adanya perubahan perilaku pengguna dari waktu ke waktu. Revisi pola dan arsitektur untuk menggabungkan komponen dengan pemanfaatan rendah guna meningkatkan pemanfaatan secara keseluruhan. Pensiunkan komponen-komponen yang tidak lagi diperlukan. Pahami kinerja dari komponen-komponen beban kerja Anda, dan optimalkan komponen yang memakai sumber daya terbanyak. Ketahui perangkat yang digunakan pelanggan untuk mengakses layanan Anda, dan implementasikan pola untuk meminimalkan kebutuhan pemutakhiran perangkat.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keberlanjutan ini:

SUS3: Bagaimana Anda memanfaatkan pola perangkat lunak dan arsitektur untuk mendukung tujuan keberlanjutan Anda?

Implementasikan pola untuk melancarkan beban dan mempertahankan penggunaan yang tinggi dan konsisten atas sumber daya yang di-deploy guna meminimalkan sumber daya yang dipakai. Komponen dapat menjadi tidak aktif akibat kurangnya pemakaian, karena adanya perubahan perilaku pengguna dari waktu ke waktu. Revisi pola dan arsitektur untuk menggabun gkan komponen dengan pemanfaatan rendah guna meningkatkan pemanfaatan secara keseluruh an. Pensiunkan komponen-komponen yang tidak lagi diperlukan. Pahami kinerja dari komponen-komponen beban kerja Anda, dan optimalkan komponen yang memakai sumber daya terbanyak . Ketahui perangkat yang digunakan pelanggan untuk mengakses layanan Anda, dan implement asikan pola untuk meminimalkan kebutuhan pemutakhiran perangkat.

Optimalkan perangkat lunak dan arsitektur untuk tugas-tugas asinkron dan terjadwal: Gunakan desain dan arsitektur perangkat lunak yang efisien untuk meminimalkan rata-rata sumber daya yang diperlukan per unit kerja. Implementasikan mekanisme yang menghasilkan pemanfaatan komponen yang merata untuk mengurangi sumber daya yang tidak aktif di antara tugas dan meminimalkan dampak lonjakan beban.

Singkirkan atau faktor ulang komponen beban kerja dengan penggunaan rendah atau nol: Pantau aktivitas beban kerja untuk mengidentifikasi perubahan dalam hal pemanfaatan setiap komponen seiring waktu. Singkirkan komponen yang tidak digunakan dan sudah tidak diperlukan, dan faktor ulang komponen dengan sedikit pemanfaatan, untuk membatasi sumber daya yang terbuang.

Optimalkan area-area kode yang memakai waktu atau sumber daya paling banyak: Pantau aktivitas beban kerja untuk mengidentifikasi komponen aplikasi yang memakai sumber daya paling banyak. Optimalkan kode yang berjalan di dalam komponen-komponen tersebut untuk meminimalkan penggunaan sumber daya sambil memaksimalkan kinerja.

Optimalkan dampak terhadap perangkat dan perlengkapan pelanggan: Pahami perangkat dan perlengkapan yang digunakan pelanggan untuk menggunakan layanan Anda, siklus hidup yang diharapkan, serta dampak penggantian komponen tersebut terhadap keuangan dan keberlanjutan. Implementasikan pola dan arsitektur perangkat lunak guna meminimalkan kebutuhan pelanggan untuk mengganti perangkat dan memutakhirkan perlengkapan. Misalnya, implementasikan fitur baru menggunakan kode yang kompatibel dengan versi perangkat keras dan sistem operasi yang lebih awal, atau kelola ukuran payload agar tidak melebihi kapasitas penyimpanan perangkat target.

Gunakan pola dan arsitektur perangkat lunak yang paling mendukung secara efektif pola akses dan penyimpanan data: Pahami bagaimana data digunakan di dalam beban kerja Anda, dipakai oleh pengguna Anda, ditransfer, dan disimpan. Seleksi teknologi untuk meminimalkan persyaratan pemrosesan data dan penyimpanan data.

## Manajemen data

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keberlanjutan ini:

SUS4: Bagaimana Anda memanfaatkan kebijakan dan pola manajemen data untuk mendukung tujuan keberlanjutan Anda?

Implementasikan praktik manajemen data untuk mengurangi penyediaan penyimpanan yang diperlukan untuk mendukung beban kerja Anda, serta untuk mengurangi sumber daya yang

SUS4: Bagaimana Anda memanfaatkan kebijakan dan pola manajemen data untuk mendukung tujuan keberlanjutan Anda?

diperlukan untuk menggunakannya. Pahami data Anda, dan gunakan konfigurasi dan teknologi penyimpanan penggunaan yang paling efektif dalam mendukung nilai bisnis data dan cara data digunakan. Buat siklus hidup data agar penyimpanan menjadi lebih efisien dan memiliki kinerja lebih rendah ketika persyaratan berkurang, dan hapus data yang tidak lagi diperlukan.

Implementasikan kebijakan klasifikasi data: Klasifikasikan data untuk memahami pentingnya setiap data bagi hasil bisnis. Gunakan informasi ini untuk menentukan kapan Anda dapat memindahkan data ke penyimpanan yang lebih hemat energi atau menghapusnya dengan aman.

Gunakan teknologi yang mendukung pola akses dan penyimpanan data: Gunakan penyimpanan yang paling mendukung cara data Anda diakses dan disimpan untuk meminimalkan sumber daya yang disediakan sambil mendukung beban kerja Anda. Misalnya, perangkat solid state (SSDs) lebih intensif energi daripada drive magnetik dan harus digunakan hanya untuk kasus penggunaan data aktif. Gunakan penyimpanan kelas arsip yang hemat energi untuk data yang jarang diakses.

Gunakan kebijakan siklus hidup untuk menghapus data yang tidak diperlukan: Kelola siklus hidup semua data Anda dan terapkan lini waktu penghapusan secara otomatis untuk meminimalkan total kebutuhan penyimpanan beban kerja Anda.

Minimalkan pengadaan yang berlebihan dalam penyimpanan blok: Untuk meminimalkan total penyimpanan yang disediakan, buat penyimpanan blok dengan alokasi ukuran yang tepat untuk beban kerja. Gunakan volume elastis untuk memperluas penyimpanan seiring pertumbuhan data tanpa harus mengubah ukuran penyimpanan yang dilampirkan ke sumber daya komputasi. Secara rutin tinjau volume elastis dan kecilkan volume dengan penyediaan berlebih agar sesuai dengan ukuran data saat ini.

Singkirkan data yang tidak diperlukan atau redundan: Gandakan data hanya saat diperlukan untuk meminimalkan total penyimpanan yang digunakan. Gunakan teknologi pencadangan yang menghilangkan data ganda pada tingkat file dan blok. Batasi penggunaan konfigurasi Redundant Array of Independent Drives (RAID) kecuali jika diperlukan untuk memenuhi. SLAs

Gunakan sistem file dan penyimpanan objek bersama untuk mengakses data umum: Adopsi penyimpanan bersama dan sumber kebenaran tunggal untuk menghindari data ganda dan mengurangi kebutuhan penyimpanan total untuk beban kerja Anda. Ambil data dari penyimpanan bersama hanya saat diperlukan. Lepaskan volume yang tidak digunakan untuk melepas sumber

daya. Minimalkan perpindahan data di jaringan: Gunakan penyimpanan bersama dan akses data dari penyimpanan data Wilayah untuk meminimalkan total sumber daya jaringan yang diperlukan untuk mendukung perpindahan data untuk beban kerja Anda.

Cadangkan data hanya saat data sulit dibuat ulang: Untuk meminimalkan penggunaan penyimpanan, hanya cadangkan data yang memiliki nilai bisnis atau diperlukan untuk memenuhi persyaratan kepatuhan. Periksa kebijakan pencadangan dan jangan sertakan penyimpanan sementara yang tidak memberikan nilai dalam skenario pemulihan.

## Perangkat keras dan layanan

Cari peluang untuk mengurangi dampak beban kerja terhadap keberlanjutan dengan membuat perubahan pada praktik manajemen perangkat keras Anda. Minimalkan jumlah perangkat keras yang perlu disediakan dan di-deploy, serta pilih perangkat keras dan layanan yang paling efisien untuk setiap beban kerja Anda.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keberlanjutan ini:

SUS5: Bagaimana Anda memilih dan menggunakan perangkat keras dan layanan cloud dalam arsitektur Anda untuk mendukung tujuan keberlanjutan Anda?

Cari peluang untuk mengurangi dampak beban kerja terhadap keberlanjutan dengan membuat perubahan pada praktik manajemen perangkat keras Anda. Minimalkan jumlah perangkat keras yang perlu disediakan dan di-deploy, serta pilih perangkat keras dan layanan yang paling efisien untuk setiap beban kerja Anda.

Gunakan perangkat keras dalam jumlah minim untuk memenuhi kebutuhan Anda: Menggunakan kemampuan cloud, Anda dapat membuat perubahan secara sering pada implementasi beban kerja Anda. Perbarui komponen yang di-deploy seiring perubahan kebutuhan Anda.

Gunakan tipe instans dengan dampak paling sedikit: Terus pantau perilisan tipe-tipe instans baru dan manfaatkan peningkatan efisiensi energi, termasuk tipe instans yang dirancang untuk mendukung beban kerja khusus seperti pelatihan dan inferensi machine learning, dan transkode video.

Gunakan layanan terkelola: Layanan terkelola mengalihkan tanggung jawab untuk mempertahankan pemanfaatan rata-rata tinggi, dan optimalisasi keberlanjutan perangkat keras yang digunakan, ke. AWS Gunakan layanan terkelola untuk meratakan dampak layanan terhadap keberlanjutan ke semua tenant layanan, sehingga kontribusi individu Anda dapat berkurang.

Optimalkan penggunaanGPUs: Unit pemrosesan grafis (GPUs) dapat menjadi sumber konsumsi daya tinggi, dan banyak GPU beban kerja sangat bervariasi, seperti rendering, transcoding, dan pelatihan dan pemodelan pembelajaran mesin. Hanya jalankan GPUs instance untuk waktu yang dibutuhkan, dan nonaktifkan dengan otomatisasi bila tidak diperlukan untuk meminimalkan sumber daya yang dikonsumsi.

## Proses dan budaya

Cari peluang untuk mengurangi dampak operasi Anda terhadap keberlanjutan dengan membuat perubahan pada praktik deployment, pengujian, dan pengembangan.

Pertanyaan berikut ini berfokus pada semua pertimbangan untuk keberlanjutan ini:

SUS6: Bagaimana proses organisasi Anda mendukung tujuan keberlanjutan Anda?

Cari peluang untuk mengurangi dampak operasi Anda terhadap keberlanjutan dengan membuat perubahan pada praktik deployment, pengujian, dan pengembangan.

Adopsi metode yang dapat memulai peningkatan keberlanjutan dengan cepat: Uji dan validasi potensi peningkatan sebelum menerapkannya ke produksi. Pertimbangkan biaya pengujian saat menghitung potensi manfaat dari sebuah peningkatan di masa mendatang. Kembangkan metode pengujian berbiaya rendah untuk memungkinkan penerapan peningkatan-peningkatan kecil.

Perbarui beban kerja Anda: sistem Up-to-date operasi, pustaka, dan aplikasi dapat meningkatkan efisiensi beban kerja dan menciptakan adopsi teknologi yang lebih efisien. Up-to-datePerangkat lunak mungkin juga menyertakan fitur untuk mengukur dampak keberlanjutan beban kerja Anda secara lebih akurat, karena vendor memberikan fitur untuk memenuhi tujuan keberlanjutan mereka sendiri.

Tingkatkan pemanfaatan lingkungan build: Gunakan otomatisasi dan infrastruktur sebagai kode untuk mengaktifkan lingkungan pra-produksi saat diperlukan dan menonaktifkannya saat tidak digunakan. Hal yang umum dilakukan adalah menjadwalkan periode ketersediaan yang bertepatan dengan jam kerja anggota tim pengembangan. Hibernasi adalah alat yang berguna untuk mempertahankan status dan mengaktifkan instans dengan cepat hanya pada saat dibutuhkan. Gunakan jenis instans dengan kapasitas lonjakan, Instans Spot, layanan basis data elastis, kontainer, dan teknologi lainnya untuk menyesuaikan pengembangan dan menguji kapasitas dengan penggunaan.

Gunakan device farm terkelola untuk pengujian: Device farm yang terkelola meratakan dampak manufaktur perangkat keras dan penggunaan sumber daya terhadap keberlanjutan ke beberapa

tenant. Device farm terkelola menawarkan tipe perangkat yang beragam sehingga Anda dapat mendukung perangkat keras yang lebih awal dan kurang populer, serta menghindari dampak keberlanjutan pelanggan akibat pemutakhiran perangkat yang tidak perlu.

## Sumber daya

Buka sumber daya berikut untuk mempelajari selengkapnya tentang praktik terbaik kami untuk keberlanjutan.

### Laporan resmi

- Pilar Keberlanjutan

### Video

- Sumpah Iklim

# Proses peninjauan

Peninjauan arsitektur harus dilakukan dengan cara yang konsisten, tanpa menyalahkan pihak mana pun untuk mendorong pengamatan yang mendalam. Hal ini harus berupa proses yang ringan (hitungan jam, bukan hari) dengan berdiskusi, bukan proses audit. Tujuan dari peninjauan arsitektur adalah untuk mengidentifikasi masalah yang sangat penting dan perlu ditangani atau area yang dapat ditingkatkan. Hasil dari peninjauan adalah serangkaian tindakan yang dapat meningkatkan pengalaman pelanggan dalam menggunakan beban kerja.

Seperti yang telah dibahas di bagian “Tentang Arsitektur”, Anda ingin semua tim bertanggung jawab atas kualitas dari setiap arsitekturnya. Untuk anggota tim yang membangun arsitektur menggunakan Kerangka Kerja Well-Architected, sebaiknya tinjau arsitektur secara berkelanjutan, tanpa perlu terpaku pada pertemuan peninjauan yang formal. Pendekatan yang hampir berkelanjutan mengizinkan anggota tim Anda memperbarui berbagai solusi seiring dengan berkembangnya arsitektur, serta meningkatkan arsitektur saat Anda memberikan fitur.

AWS Well-Architected Framework selaras dengan cara meninjau sistem dan layanan secara AWS internal. Hal ini didasarkan pada seperangkat prinsip desain yang mempengaruhi pendekatan arsitektur, dan pertanyaan yang memverifikasi bahwa orang tidak mengabaikan area yang sering ditampilkan dalam Root Cause Analysis (RCA). Setiap kali ada masalah signifikan dengan sistem internal, AWS layanan, atau pelanggan, kami melihat RCA untuk melihat apakah kami dapat meningkatkan proses peninjauan yang kami gunakan.

Ulasan harus diterapkan pada tonggak penting dalam siklus hidup produk, di awal fase desain untuk menghindari pintu satu arah yang sulit diubah, dan kemudian sebelum tanggal untuk go-live. (Banyak keputusan yang dapat dibalik, pintu dua arah. Keputusan-keputusan tersebut dapat menggunakan proses yang ringan. Pintu satu arah sulit atau tidak mungkin untuk dibalik dan memerlukan pemeriksaan lebih lanjut sebelum membuatnya.) Setelah Anda masuk ke dalam proses produksi, beban kerja Anda akan meningkat saat Anda menambahkan fitur baru dan mengubah implementasi teknologi. Arsitektur beban kerja berubah seiring waktu. Anda harus menerapkan praktik pemeliharaan yang sesuai untuk mempertahankan karakteristik arsitektur agar tidak memburuk saat Anda mengembangkannya. Saat Anda membuat perubahan arsitektur yang signifikan, Anda harus melakukannya sesuai dengan aturan proses pemeliharaan termasuk peninjauan Well-Architected.

Jika Anda ingin menggunakan hasil tinjauan sebagai snapshot sekali pakai atau pengukuran independen, Anda perlu memverifikasi bahwa Anda berdiskusi dengan orang yang tepat. Kita sering

kali mendapati bahwa tinjauan adalah hal pertama yang dapat membuat tim benar-benar paham tentang apa yang telah mereka implementasikan. Pendekatan yang sangat cocok digunakan saat meninjau beban kerja tim lain adalah melakukan percakapan informal tentang arsitektur mereka dengan mencermati jawaban-jawaban dari pertanyaan. Kemudian Anda dapat menindaklanjutinya dengan satu atau dua pertemuan untuk mendapatkan kejelasan atau mendalami area yang masih ambigu atau berisiko.

Berikut adalah beberapa saran item untuk memfasilitasi rapat Anda:

- Ruang rapat yang dilengkapi papan tulis
- Hasil cetak diagram atau catatan desain
- Daftar tindakan pertanyaan yang memerlukan out-of-band penelitian untuk menjawab (misalnya, “apakah kita mengaktifkan enkripsi atau tidak?”)

Setelah peninjauan selesai, Anda harus membuat daftar masalah yang dapat diprioritaskan berdasarkan konteks bisnis Anda. Anda juga ingin mempertimbangkan dampak dari masalah tersebut pada day-to-day pekerjaan tim Anda. Jika masalah tersebut segera diatasi, Anda memiliki lebih banyak waktu untuk meningkatkan nilai bisnis ketimbang sibuk dengan masalah yang berulang. Saat Anda mengatasi masalah, Anda dapat memperbarui tinjauan Anda untuk melihat perkembangan arsitektur.

Saat nilai dari tinjauan sudah diketahui dengan jelas, pada awalnya tim baru mungkin akan menafikannya. Berikut adalah beberapa kekurangan yang dapat ditangani dengan mengedukasi tim terkait manfaat tinjauan:

- “Kami terlalu sibuk!” (Sering diucapkan ketika tim bersiap untuk peluncuran yang signifikan.)
  - Jika Anda bersiap untuk peluncuran besar, Anda menginginkan hal tersebut berjalan lancar. Tinjauan mengizinkan Anda memahami masalah apa pun yang mungkin terlewat.
  - Sebaiknya, lakukan peninjauan sedini mungkin dalam siklus hidup produk untuk mengetahui risiko dan menyiapkan rencana mitigasi yang selaras dengan peta strategi (roadmap) penyediaan fitur.
- “Kami tidak punya waktu untuk mengubah hasilnya!” (Sering diucapkan ketika ada acara dengan jadwal yang sudah tetap, seperti Super Bowl, yang ditargetkan.)
  - Acara-acara tersebut memiliki jadwal yang tetap. Apakah Anda benar-benar ingin melakukan sesuatu tanpa mengetahui risikonya terhadap arsitektur Anda? Bahkan jika Anda tidak mengatasi masalah-masalah ini, Anda masih dapat menggunakan buku panduan untuk menanganinya apabila hal itu terjadi.

- “Kita harus menjaga rahasia implementasi solusi kita dari pihak lain!”
  - Jika Anda menyodorkan pertanyaan kepada tim terkait Kerangka Kerja Well-Architected, mereka akan melihat bahwa tidak ada satu pun dari pertanyaan tersebut yang mengarah pada informasi hak milik teknis atau komersial apa pun.

Saat Anda melakukan beberapa peninjauan terhadap beberapa tim dalam organisasi, Anda mungkin mengidentifikasi masalah-masalah yang mendasar. Misalnya, Anda mungkin mendapati grup dari beberapa tim memiliki kumpulan masalah dalam pilar atau topik tertentu. Anda perlu melihat semua tinjauan secara menyeluruh, kemudian mengidentifikasi mekanisme, pelatihan, atau pembicaraan teknis utama yang dapat membantu Anda mengetahui masalah mendasar tersebut.

# Kesimpulan

AWS Well-Architected Framework memberikan praktik terbaik arsitektur di enam pilar untuk merancang dan mengoperasikan sistem yang andal, aman, efisien, hemat biaya, dan berkelanjutan di cloud. Kerangka kerja ini memberikan serangkaian pertanyaan yang memungkinkan Anda meninjau arsitektur yang ada atau yang diusulkan. Ini juga menyediakan serangkaian praktik AWS terbaik untuk setiap pilar. Menggunakan Kerangka kerja dalam arsitektur Anda akan membantu Anda menghasilkan sistem yang stabil dan efisien, sehingga memungkinkan Anda berfokus pada kebutuhan fungsional Anda.

# Kontributor

Individu dan organisasi berikut berkontribusi terhadap dokumen ini:

- Brian Carlson, Operations Lead Well-Architected, Amazon Web Services
- Ben Potter, Security Lead Well-Architected, Amazon Web Services
- Seth Eliot, Reliability Lead Well-Architected, Amazon Web Services
- Eric Pullen, Sr. Solutions Architect, Amazon Web Services
- Rodney Lester, Arsitek Solusi Utama, Amazon Web Services
- Jon Steele, Manajer Akun Teknis Senior, Amazon Web Services
- Max Ramsay, Keamanan Prinsipal Solutions Architect, Amazon Web Services
- Callum Hughes, Solutions Architect, Amazon Web Services
- Ben Mergen, Cost Lead Solutions Architect Senior, Amazon Web Services
- Chris Kozlowski, Spesialis Senior Manajer Akun Teknis, Dukungan Perusahaan, Amazon Web Services
- Alex Livingstone, Arsitek Solusi Spesialis Utama, Operasi Cloud, Amazon Web Services
- Paul Moran, Ahli Teknologi Utama, Dukungan Perusahaan, Amazon Web Services
- Peter Mullen, Advisory Consultant, Layanan Profesional, Amazon Web Services
- Chris Pates, Spesialis Senior Manajer Akun Teknis, Dukungan Perusahaan, Amazon Web Services
- Arvind Raghunathan, Spesialis Utama Manajer Akun Teknis, Dukungan Perusahaan, Amazon Web Services
- Sam Mokhtari, Senior Efficiency Lead Solutions Architect, Amazon Web Services

# Sumber bacaan lebih lanjut

Pusat Arsitektur AWS

Kepatuhan AWS Cloud

AWS Program Mitra Well-Architected

AWS Well-Architected Tool

AWS Beranda Well-Architected

Laporan resmi PilarKeunggulan Operasional

Laporan resmi Pilar Keamanan

Laporan Resmi Pilar Keandalan

Laporan Resmi Pilar Efisiensi Kinerja

Laporan Resmi Pilar Optimasi Biaya

Laporan resmi Pilar Keberlanjutan

Amazon Builders' Library

# Revisi dokumen

Untuk diberitahu tentang pembaruan pada whitepaper ini, berlangganan feed. RSS

| Perubahan | Deskripsi | Tanggal |
| --- | --- | --- |
| Panduan praktik terbaik yang sudah diperbarui | Pembaruan praktik terbaik skala besar sudah dilakukan di seluruh pilar. Praktik-p raktik terbaik yang baru telah ditambahkan untuk mengoptim alkan keamanan dan biaya. | 27 Juni 2024 |
| Pembaruan besar | Pembaruan pilar utama. | 3 Oktober 2023 |
| Pembaruan untuk Kerangka Kerja baru | Praktik terbaik diperbarui dengan panduan preskript if dan praktik terbaik baru ditambahkan. Pertanyaan baru ditambahkan ke pilar Keamanan dan Optimasi Biaya. | 10 April 2023 |
| Pembaruan kecil | Penambahan definisi untuk tingkat upaya dan pembaruan praktik terbaik di lampiran. | 20 Oktober 2022 |
| Laporan resmi diperbarui | Penambahan Pilar Keberlanj utan dan pembaruan tautan. | 2 Desember 2021 |
| Pembaruan besar | Pilar Keberlanjutan ditambahk an ke kerangka kerja. | 20 November 2021 |
| Pembaruan kecil | Bahasa noninklusif dihilangk an. | 22 April 2021 |
| Pembaruan kecil | Perbaikan banyak tautan. | 10 Maret 2021 |

| | | |
|---|---|---|
| Pembaruan kecil | Perubahan editorial kecil di seluruh dokumen. | 15 Juli 2020 |
| Pembaruan untuk Kerangka Kerja baru | Peninjauan dan penulisan ulang sebagian besar pertanyaan dan jawaban. | 8 Juli 2020 |
| Laporan resmi diperbarui | Penambahan AWS Well-Architected Tool, tautan ke AWS Well-Architected Labs, dan AWS Well-Architected Partners, perbaikan kecil untuk mengaktifkan beberapa versi bahasa kerangka kerja. | 1 Juli 2019 |
| Laporan resmi diperbarui | Peninjauan dan penulisan ulang sebagian besar pertanyaan dan jawaban, untuk memastikan pertanyaan berfokus pada satu topik pada satu waktu. Ini menyebabk an beberapa pertanyaan sebelumnya dipecah menjadi lebih dari satu pertanyaan. Penambahan istilah umum ke definisi (beban kerja, komponen, dll.). Perubahan penyajian pertanyaan di bagian utama agar menyertak an teks deskriptif. | 1 November 2018 |
| Laporan resmi diperbarui | Pembaruan untuk menyederh anakan teks pertanyaan, menstandarkan jawaban, dan meningkatkan keterbacaan. | 1 Juni 2018 |

| | | |
|---|---|---|
| Laporan resmi diperbarui | Pilar Keunggulan Operasional dipindah ke depan dan ditulis ulang agar memayungi pilar-pilar lain. Segarkan pilar lain untuk mencerminkan evolusi. AWS | 1 November 2017 |
| Laporan resmi diperbarui | Pembaruan Framework agar menyertakan pilar keunggulan operasional, dan revisi serta pembaruan pilar-pilar lain untuk mengurangi duplikasi dan memasukkan pembelajaran dari pelaksanaan peninjauan dengan ribuan pelanggan. | 1 November 2016 |
| Pembaruan kecil | Memperbarui Lampiran dengan informasi CloudWatch Log Amazon saat ini. | 1 November 2015 |
| Publikasi awal | AWS Well-Architected Framework diterbitkan. | 1 Oktober 2015 |

> **Note**
>
> Untuk berlangganan RSS pembaruan, Anda harus mengaktifkan RSS plugin untuk browser yang Anda gunakan.

Versi kerangka kerja:

- 10-03-2023 (saat ini)
- 2023-04-10
- 2022-03-31

# Lampiran: Pertanyaan dan praktik terbaik

Lampiran ini merangkum semua pertanyaan dan praktik terbaik di Kerangka Kerja AWS Well-Architected.

Pilar

- Keunggulan operasional
- Keamanan
- Keandalan
- Efisiensi kinerja
- Optimalisasi Biaya
- Keberlanjutan

## Keunggulan operasional

Pilar Keunggulan Operasional mencakup kemampuan untuk mendukung pengembangan dan menjalankan beban kerja dengan efektif, mendapatkan wawasan tentang operasi Anda, serta meningkatkan proses dan prosedur pendukung secara terus-menerus untuk memberikan nilai bisnis. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keunggulan Operasional.

Area praktik terbaik

- Organisasi
- Persiapkan
- Jalankan
- Kembangkan

### Organisasi

Pertanyaan

- OPS1. Bagaimana Anda menentukan apa prioritas Anda?
- OPS2. Bagaimana cara menyusun struktur organisasi untuk mendukung hasil bisnis Anda?
- OPS3. Bagaimana budaya organisasi Anda mendukung hasil bisnis Anda?

## OPS1. Bagaimana Anda menentukan apa prioritas Anda?

Setiap orang harus memahami peran mereka dalam mewujudkan kesuksesan bisnis. Miliki sasaran bersama guna menetapkan prioritas untuk sumber daya. Ini akan memaksimalkan manfaat dari upaya Anda.

Praktik terbaik

- OPS01-BP01 Mengevaluasi kebutuhan pelanggan
- OPS01-BP02 Mengevaluasi kebutuhan nasabah internal
- OPS01-BP03 Mengevaluasi persyaratan tata kelola
- OPS01-BP04 Mengevaluasi persyaratan kepatuhan
- OPS01-BP05 Mengevaluasi lanskap ancaman
- OPS01-BP06 Mengevaluasi pengorbanan sambil mengelola manfaat dan risiko

### OPS01-BP01 Mengevaluasi kebutuhan pelanggan

Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk menentukan ke mana harus memfokuskan usaha terkait kebutuhan pelanggan eksternal. Hal ini memverifikasi bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasi yang dibutuhkan untuk mencapai hasil bisnis yang diinginkan.

Hasil yang diinginkan:

- Anda bekerja dengan berpatokan pada hasil pelanggan.
- Anda memahami bagaimana praktik operasional Anda mendukung hasil dan tujuan bisnis.
- Anda melibatkan semua pihak yang relevan.
- Anda memiliki mekanisme untuk merekam kebutuhan pelanggan.

Anti-pola umum:

- Anda memutuskan untuk tidak menyediakan dukungan pelanggan di luar jam kerja, tetapi Anda belum meninjau riwayat data permintaan dukungan. Anda tidak tahu apakah hal ini akan memengaruhi pelanggan Anda.
- Anda mengembangkan fitur baru, tetapi belum melibatkan pelanggan untuk mencari tahu apakah hal tersebut diinginkan—jika diinginkan, dalam bentuk apa—dan belum menjalankan eksperimen untuk memvalidasi kebutuhan serta metode penyediaannya.

Manfaat penerapan praktik terbaik: Pelanggan yang kebutuhannya terpenuhi akan sangat berpotensi menjadi pelanggan tetap. Mengevaluasi dan memahami kebutuhan pelanggan eksternal akan menginformasikan cara Anda memprioritaskan usaha untuk memberikan nilai bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pahami kebutuhan bisnis: Kesuksesan bisnis terwujud dengan tujuan dan pemahaman bersama di seluruh pemangku kepentingan, termasuk tim bisnis, pengembangan, dan operasional.

Tinjau tujuan bisnis, kebutuhan, dan prioritas pelanggan eksternal: Libatkan para pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk mendiskusikan tujuan, kebutuhan, dan prioritas pelanggan eksternal. Hal ini memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasional yang dibutuhkan untuk mencapai hasil bisnis dan pelanggan.

Tetapkan pemahaman bersama: Tetapkan pemahaman bersama terkait fungsi bisnis beban kerja, peran masing-masing tim dalam mengoperasikan beban kerja, dan bagaimana faktor-faktor ini mendukung tujuan bisnis bersama bagi seluruh pelanggan internal dan eksternal.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP03 Menerapkan loop umpan balik

### OPS01-BP02 Mengevaluasi kebutuhan nasabah internal

Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk menentukan ke mana harus memfokuskan usaha terkait kebutuhan pelanggan internal. Hal ini akan memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasi yang dibutuhkan untuk mencapai hasil bisnis yang diinginkan.

Hasil yang diinginkan:

- Anda menggunakan prioritas yang ditetapkan untuk memfokuskan usaha peningkatan yang dapat memberikan dampak paling besar (misalnya, mengembangkan keterampilan tim, meningkatkan kinerja beban kerja, mengurangi biaya, mengotomatiskan runbook, atau meningkatkan pemantauan).

- Anda memperbarui prioritas Anda sesuai perubahan kebutuhan.

Anti-pola umum:

- Anda memutuskan untuk mengubah alokasi alamat IP untuk tim produk tanpa berkonsultasi dengan mereka agar manajemen jaringan menjadi lebih mudah. Anda tidak tahu dampak yang akan ditimbulkan kepada tim produk.
- Anda mengimplementasikan alat pengembangan baru tetapi belum melibatkan pelanggan internal untuk mencari tahu apakah alat itu dibutuhkan atau kompatibel dengan praktik yang sudah ada.
- Anda mengimplementasikan sistem pemantauan baru, tetapi belum menghubungi pelanggan internal untuk mencari tahu apakah mereka memiliki kebutuhan pemantauan atau pelaporan yang perlu dipertimbangkan.

Manfaat menerapkan praktik terbaik: Mengevaluasi dan memahami kebutuhan pelanggan internal akan menginformasikan cara Anda memprioritaskan usaha untuk memberikan nilai bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Pahami kebutuhan bisnis: Kesuksesan bisnis diciptakan dengan tujuan dan pemahaman bersama di seluruh pemangku kepentingan termasuk tim bisnis, pengembangan, dan operasional.
- Tinjau tujuan bisnis, kebutuhan, dan prioritas pelanggan internal: Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk mendiskusikan tujuan, kebutuhan, dan prioritas pelanggan internal. Hal ini memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasional yang dibutuhkan untuk mencapai hasil bisnis dan pelanggan.
- Tetapkan pemahaman bersama: Tetapkan pemahaman bersama terkait fungsi bisnis beban kerja, peran masing-masing tim dalam mengoperasikan beban kerja, dan bagaimana faktor-faktor ini mendukung tujuan bisnis bersama bagi seluruh pelanggan internal dan eksternal.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP03 Menerapkan loop umpan balik

### OPS01-BP03 Mengevaluasi persyaratan tata kelola

Tata kelola adalah serangkaian kebijakan, aturan, atau kerangka kerja yang digunakan perusahaan untuk mencapai sasaran bisnisnya. Persyaratan tata kelola dibuat dari dalam organisasi Anda. Persyaratan ini dapat memengaruhi jenis teknologi yang Anda pilih atau memengaruhi cara Anda mengoperasikan beban kerja Anda. Sertakan persyaratan tata kelola organisasi ke dalam beban kerja Anda. Konformitas adalah kemampuan untuk menunjukkan bahwa Anda telah mengimplementasikan persyaratan tata kelola.

Hasil yang diinginkan:

- Persyaratan tata kelola disertakan ke dalam operasi dan desain arsitektur beban kerja Anda.
- Anda dapat memberikan bukti bahwa Anda telah mengikuti persyaratan tata kelola.
- Persyaratan tata kelola ditinjau dan diperbarui secara teratur.

Anti-pola umum:

- Organisasi Anda memerintahkan agar akun root memiliki autentikasi multi-faktor. Anda gagal mengimplementasikan persyaratan ini dan akun root terancam bahaya.
- Selama desain beban kerja Anda, Anda memilih jenis instans yang tidak disetujui oleh departemen IT. Anda tidak dapat meluncurkan beban kerja Anda dan harus mendesain ulang.
- Anda diwajibkan memiliki rencana pemulihan bencana. Anda tidak membuat rencana pemulihan bencana dan beban kerja Anda mengalami pemadaman yang berdurasi lama.
- Tim Anda ingin menggunakan instans baru tetapi persyaratan tata kelola Anda belum diperbarui untuk memungkinkannya.

Manfaat menjalankan praktik terbaik ini:

- Mengikuti persyaratan tata kelola akan menyelaraskan beban kerja Anda dengan kebijakan lebih besar dalam organisasi.
- Persyaratan tata kelola mencerminkan standar industri dan praktik terbaik untuk organisasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Identifikasi persyaratan tata kelola melalui kerja sama dengan pemangku kepentingan dan organisasi tata kelola. Sertakan persyaratan tata kelola ke dalam beban kerja Anda. Dapat menunjukkan bukti bahwa Anda telah mengikuti persyaratan tata kelola.

Contoh pelanggan

Di AnyCompany Retail, tim operasi cloud bekerja dengan pemangku kepentingan di seluruh organisasi untuk mengembangkan persyaratan tata kelola. Misalnya, mereka melarang SSH akses ke EC2 instans Amazon. Jika tim memerlukan akses ke sistem, mereka harus menggunakan AWS Systems Manager Session Manager. Tim operasi cloud secara teratur memperbarui persyaratan tata kelola saat layanan baru tersedia.

Langkah-langkah implementasi

1. Identifikasi pemangku kepentingan untuk beban kerja Anda, termasuk semua tim tersentralisasi.
2. Bekerja sama dengan pemangku kepentingan untuk mengidentifikasi persyaratan tata kelola.
3. Setelah Anda membuat daftar, prioritaskan item untuk ditingkatkan, dan mulai implementasikan ke dalam beban kerja Anda.
   a. Gunakan layanan seperti AWS Configuntuk membuat governance-as-code dan memvalidasi bahwa persyaratan tata kelola diikuti.
   b. Jika Anda menggunakan AWS Organizations, Anda dapat memanfaatkan Kebijakan Kontrol Layanan untuk menerapkan persyaratan tata kelola.
4. Berikan dokumentasi yang memvalidasi implementasinya.

Tingkat upaya untuk rencana implementasi: Sedang. Mengimplementasikan persyaratan tata kelola yang tidak ada dapat mengakibatkan beban kerja Anda harus dikerjakan ulang.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP04 Mengevaluasi persyaratan kepatuhan - Kepatuhan itu seperti tata kelola tetapi kepatuhan berasal dari luar organisasi.

Dokumen terkait:

- AWS Panduan Lingkungan Cloud Manajemen dan Tata Kelola

- Praktik Terbaik untuk Kebijakan Kontrol AWS Organizations Layanan di Lingkungan Multi-Akun
- Tata Kelola dalam AWS Cloud: Keseimbangan yang Tepat Antara Kelincahan dan Keamanan
- Apa itu Tata Kelola, Risiko, dan Kepatuhan (GRC)?

Video terkait:

- AWS Manajemen dan Tata Kelola: Konfigurasi, Kepatuhan, dan Audit - AWS Pembicaraan Teknologi Online
- AWS re: Inforce 2019: Tata Kelola untuk Zaman Awan (-R1) DEM12
- AWS re:invent 2020: Mencapai kepatuhan sebagai menggunakan kode AWS Config
- AWS Re:invent 2020: Tata kelola tangkas di AWS GovCloud (US)

Contoh terkait:

- AWS Config Sampel Paket Kesesuaian

Layanan terkait:

- AWS Config
- AWS Organizations - Kebijakan Pengendalian Layanan

OPS01-BP04 Mengevaluasi persyaratan kepatuhan

Persyaratan kepatuhan peraturan, industri, dan internal merupakan pendorong penting dalam menentukan prioritas organisasi Anda. Kerangka kerja kepatuhan Anda dapat menghalangi Anda untuk menggunakan teknologi atau lokasi geografi tertentu. Terapkan uji tuntas jika tidak ada kerangka kerja kepatuhan eksternal yang diidentifikasi. Buatlah audit atau laporan yang memvalidasi kepatuhan.

Jika Anda mengiklankan bahwa produk Anda memenuhi standar kepatuhan tertentu, maka Anda harus memiliki proses internal untuk memastikan kepatuhan yang berkelanjutan. Contoh standar kepatuhan termasuk PCIDSS, FedRAMP, danHIPAA. Standar kepatuhan yang berlaku akan ditentukan oleh berbagai faktor, seperti jenis data yang disimpan atau dikirim oleh solusi, serta wilayah geografis mana yang didukung oleh solusi.

Hasil yang diinginkan:

- Persyaratan kepatuhan berdasarkan regulasi, industri, dan internal disertakan ke dalam pemilihan arsitektur.
- Anda dapat memvalidasi kepatuhan dan membuat laporan audit.

Anti-pola umum:

- Bagian dari beban kerja Anda termasuk dalam kerangka Standar Keamanan Data Industri Kartu Pembayaran (PCI-DSS) tetapi beban kerja Anda menyimpan data kartu kredit tanpa terenkripsi.
- Arsitek dan pengembang perangkat lunak Anda tidak mengetahui kerangka kerja kepatuhan yang harus ditaati oleh organisasi Anda.
- Audit tahunan Sistem dan Organizations Control (SOC2) Tipe II akan segera terjadi dan Anda tidak dapat memverifikasi bahwa kontrol sudah ada.

Manfaat menjalankan praktik terbaik ini:

- Mengevaluasi dan memahami persyaratan kepatuhan yang berlaku untuk beban kerja Anda akan menginformasikan bagaimana Anda memprioritaskan upaya-upaya Anda untuk memberikan nilai bisnis.
- Anda memilih teknologi dan lokasi yang tepat yang selaras dengan kerangka kerja kepatuhan Anda.
- Mendesain beban kerja Anda agar dapat diaudit akan membantu Anda membuktikan bahwa Anda menaati kerangka kerja kepatuhan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Mengimplementasikan praktik terbaik ini berarti Anda menyertakan persyaratan kepatuhan ke dalam proses desain arsitektur Anda. Anggota tim Anda mengetahui kerangka kerja kepatuhan yang diperlukan. Anda memvalidasi bahwa kepatuhan selaras dengan kerangka kerja.

Contoh pelanggan

AnyCompany Toko ritel informasi kartu kredit untuk pelanggan. Pengembang di tim penyimpanan kartu memahami bahwa mereka harus mematuhi DSS kerangka kerja. PCI Mereka telah mengambil langkah-langkah untuk memverifikasi bahwa informasi kartu kredit disimpan dan diakses dengan

aman sesuai dengan DSS kerangka kerja. PCI Setiap tahun mereka bekerja sama dengan tim keamanan mereka untuk melakukan validasi kepatuhan.

Langkah-langkah implementasi

1. Bekerjasamalah dengan tim tata kelola dan kepatuhan Anda untuk menentukan kerangka kerja kepatuhan industri, peraturan, atau internal apa yang harus ditaati oleh beban kerja Anda. Sertakan kerangka kerja kepatuhan ke dalam beban kerja Anda.
   a. Validasi kepatuhan berkelanjutan AWS sumber daya dengan layanan seperti AWS Compute Optimizerdan. AWS Security Hub
2. Didik anggota tim Anda mengenai persyaratan kepatuhan sehingga mereka dapat mengoperasikan dan mengubah beban kerja sesuai dengan persyaratan kepatuhan. Persyaratan kepatuhan harus disertakan dalam pilihan-pilihan berkaitan dengan arsitektur dan teknologi.
3. Tergantung pada kerangka kerja kepatuhan yang diterapkan, Anda mungkin diharuskan untuk membuat laporan kepatuhan atau audit. Bekerjasamalah dengan organisasi Anda untuk melakukan otomatisasi terhadap proses ini sebanyak mungkin.
   a. Gunakan layanan-layanan seperti AWS Audit Manager untuk menghasilkan validasi kepatuhan dan menghasilkan laporan audit.
   b. Anda dapat mengunduh dokumen AWS keamanan dan kepatuhan dengan AWS Artifact.

Tingkat upaya untuk rencana implementasi: Sedang. Mengimplementasikan kerangka kerja kepatuhan bisa jadi sulit dilakukan. Membuat laporan audit atau dokumen kepatuhan menambahkan kompleksitas tambahan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan pengendalian - Tujuan pengendalian keamanan adalah bagian penting dari kepatuhan secara keseluruhan.
- SEC01-BP06 Mengotomatiskan pengujian dan validasi kontrol keamanan di pipeline - Sebagai bagian dari pipeline Anda, validasi kontrol keamanan. Anda juga dapat membuat dokumentasi kepatuhan untuk perubahan-perubahan baru.
- SEC07-BP02 Tentukan kontrol perlindungan data - Banyak kerangka kerja kepatuhan memiliki penanganan data dan kebijakan penyimpanan berbasis.
- SEC10-BP03 Siapkan kemampuan forensik - Kemampuan forensik terkadang dapat digunakan dalam kepatuhan audit.

Dokumen terkait:

- AWS Pusat Kepatuhan
- AWS Sumber Daya Kepatuhan
- AWS Whitepaper Risiko dan Kepatuhan
- AWS Model Tanggung Jawab Bersama
- AWS layanan dalam lingkup oleh program kepatuhan

Video terkait:

- AWS re:invent 2020: Mencapai kepatuhan sebagai menggunakan kode AWS Compute Optimizer
- AWS re:invent 2021 - Kepatuhan, jaminan, dan audit cloud
- AWS Summit ATL 2022 - Menerapkan kepatuhan, jaminan, dan audit pada AWS (02) COP2

Contoh terkait:

- PCIDSSdan Praktik Terbaik Keamanan AWS Dasar tentang AWS

Layanan terkait:

- AWS Artifact
- AWS Audit Manager
- AWS Compute Optimizer
- AWS Security Hub

OPS01-BP05 Mengevaluasi lanskap ancaman

Evaluasi ancaman pada bisnis (misalnya, persaingan, risiko dan kewajiban bisnis, risiko operasional, serta ancaman keamanan informasi) dan pelihara informasi yang ada di registri risiko. Sertakan dampak risiko ketika menentukan ke mana upaya harus difokuskan.

Kerangka Kerja Well-Architected menekankan pembelajaran, pengukuran, dan peningkatan. Ini memberikan pendekatan yang konsisten bagi Anda untuk mengevaluasi arsitektur, dan menerapkan desain yang akan skala dari waktu ke waktu. AWS menyediakan AWS Well-Architected Tooluntuk membantu Anda meninjau pendekatan Anda sebelum pengembangan, keadaan beban kerja Anda

sebelum produksi, dan keadaan beban kerja Anda dalam produksi. Anda dapat membandingkannya dengan praktik terbaik AWS arsitektur terbaru, memantau status keseluruhan beban kerja Anda, dan mendapatkan wawasan tentang potensi risiko.

AWS pelanggan memenuhi syarat untuk Tinjauan Well-Architected yang dipandu dari beban kerja kritis misi mereka untuk mengukur arsitektur mereka terhadap praktik terbaik. AWS Pelanggan Dukungan Perusahaan memenuhi syarat untuk Peninjauan Operasi, yang dirancang untuk membantu mereka mengidentifikasi kesenjangan yang ada dalam pendekatan mereka untuk beroperasi di cloud.

Interaksi lintas tim pada tinjauan ini akan membantu Anda dalam membangun pemahaman bersama tentang beban kerja Anda serta bagaimana peran tim akan membantu meraih keberhasilan. Kebutuhan yang diidentifikasi melalui tinjauan tersebut dapat membantu membentuk prioritas Anda.

AWS Trusted Advisor adalah sebuah alat yang menyediakan akses ke set inti pemeriksaan yang menyarankan optimalisasi yang dapat membantu membentuk prioritas Anda. Pelanggan Dukungan Bisnis dan Perusahaan menerima akses ke pemeriksaan tambahan yang berfokus pada keamanan, keandalan, kinerja, dan optimisasi biaya yang dapat membantu membentuk prioritas mereka lebih lanjut.

Hasil yang diinginkan:

- Anda secara teratur meninjau dan bertindak berdasarkan Well-Architected dan output Trusted Advisor
- Anda mengetahui status patch terbaru layanan Anda
- Anda memahami risiko dan dampak ancaman yang diketahui dan bertindak sebagaimana mestinya
- Anda mengimplementasikan mitigasi sesuai keperluan
- Anda mengomunikasikan tindakan dan konteks

Anti-pola umum:

- Anda menggunakan pustaka perangkat lunak versi lama dalam produk Anda. Anda tidak mengetahui bahwa ada pembaruan keamanan pustaka untuk masalah-masalah yang mungkin memiliki dampak yang tidak diinginkan pada beban kerja Anda.
- Kompetitor Anda baru saja merilis sebuah versi produk mereka yang dapat mengatasi keluhan pelanggan Anda tentang produk Anda. Anda belum memprioritaskan penanganan masalah-masalah yang sudah diketahui ini.

- Pembuat peraturan telah menyasar perusahaan-perusahaan yang tidak mematuhi persyaratan kepatuhan hukum seperti Anda. Anda belum memprioritaskan penanganan persyaratan kepatuhan Anda yang belum terpenuhi.

Manfaat menerapkan praktik terbaik ini: Anda dapat mengidentifikasi dan memahami ancaman terhadap organisasi dan beban kerja Anda, hal ini akan membantu Anda menentukan ancaman mana yang harus ditangani, tingkat prioritasnya, serta sumber daya yang diperlukan untuk melakukannya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Evaluasi lanskap ancaman: Evaluasi ancaman terhadap bisnis (misalnya kompetisi, risiko dan kewajiban bisnis, risiko operasional, dan ancaman keamanan informasi), sehingga Anda dapat menyertakan dampaknya ketika menentukan ke mana upaya perlu difokuskan.
  - Buletin Keamanan Terkini AWS
  - AWS Trusted Advisor
- Pelihara model ancaman: Buat dan pelihara model ancaman yang mengidentifikasi potensi ancaman, mitigasi terencana dan sedang diterapkan, serta prioritasnya. Tinjau kemungkinan ancaman yang berwujud insiden, biaya untuk melakukan pemulihan dari insiden tersebut serta perkiraan bahaya yang ditimbulkan, dan biaya untuk mencegah terjadinya insiden tersebut. Revisi proritas seiring perubahan konten model ancaman.

Sumber daya

Praktik terbaik terkait:

- SEC01-BP07 Mengidentifikasi ancaman dan memprioritaskan mitigasi menggunakan model ancaman

Dokumen terkait:

- Kepatuhan AWS Cloud
- Buletin Keamanan Terkini AWS
- AWS Trusted Advisor

Video terkait:

- AWS re: Inforce 2023 - Alat untuk membantu Anda meningkatkan pemodelan ancaman Anda

OPS01-BP06 Mengevaluasi pengorbanan sambil mengelola manfaat dan risiko

Kepentingan yang saling berbenturan dari berbagai pihak dapat menyulitkan Anda dalam membuat prioritas upaya, pembangunan kemampuan, dan pemberian hasil yang selaras dengan strategi bisnis. Misalnya, Anda mungkin diminta speed-to-market untuk mempercepat fitur baru daripada mengoptimalkan biaya infrastruktur TI. Hal ini dapat menjadikan dua pihak yang berkepentingan mengalami konflik satu sama lain. Dalam situasi seperti ini, keputusan perlu dibawa ke otoritas yang lebih tinggi untuk menyelesaikan konflik. Data diperlukan agar tidak ada keterikatan emosional dalam proses pengambilan keputusan.

Tantangan yang sama dapat terjadi pada tingkatan taktis. Misalnya, ketika ada pilihan antara menggunakan teknologi basis data relasional atau non-relasional, hal itu dapat berdampak signifikan pada pengoperasian aplikasi. Sangat penting bagi Anda untuk memahami hasil yang dapat diprediksi dari berbagai pilihan.

AWS dapat membantu Anda mendidik tim Anda tentang AWS dan layanannya untuk meningkatkan pemahaman mereka tentang bagaimana pilihan mereka dapat berdampak pada beban kerja Anda. Gunakan sumber daya yang disediakan oleh AWS Support (Pusat Pengetahuan AWS, Forum Diskusi AWS, dan Pusat AWS Support) serta Dokumentasi AWS untuk memberikan edukasi bagi tim Anda. Untuk pertanyaan lebih lanjut, hubungi AWS Support.

AWS juga membagikan praktik dan pola terbaik operasional di Amazon Builders' Library. Berbagai macam informasi berguna lainnya tersedia melalui AWS Blog dan The Official AWS Podcast.

Hasil yang diharapkan: Anda memiliki kerangka kerja pengambilan keputusan yang jelas untuk memudahkan Anda mengambil keputusan penting di setiap level dalam organisasi pengiriman cloud Anda. Kerangka kerja ini mencakup fitur-fitur seperti pencatatan risiko, peran yang ditentukan untuk wewenang pengambilan keputusan, dan model yang ditentukan untuk masing-masing tingkat keputusan yang dapat diambil. Kerangka kerja ini juga menetapkan di awal bagaimana konflik akan diselesaikan, data apa yang perlu disajikan, dan bagaimana opsi diprioritaskan, sehingga setelah keputusan diambil, Anda dapat menjalankannya tanpa jeda. Kerangka pengambilan keputusan ini mencakup pendekatan terstandardisasi dalam meninjau dan menimbang manfaat-manfaat serta risiko dari setiap keputusan yang diambil untuk memahami kompromi. Ini mungkin mencakup faktor-faktor eksternal, seperti misalnya, kepatuhan terhadap persyaratan kepatuhan berdasarkan regulasi.

Anti-pola umum:

- Investor Anda meminta Anda menunjukkan kepatuhan terhadap Standar Keamanan Data Industri Kartu Pembayaran (PCIDSS). Anda tidak mempertimbangkan kompromi antara memenuhi permintaan mereka dan melanjutkan upaya-upaya pengembangan yang Anda lakukan saat ini. Alih-alih, Anda melanjutkan upaya-upaya pengembangan tanpa menunjukkan kepatuhan. Investor Anda menghentikan dukungan untuk perusahaan Anda karena mengkhawatirkan keamanan platform Anda serta investasi yang mereka tanamkan.
- Anda telah memutuskan untuk menyertakan pustaka yang ditemukan di internet oleh salah satu pengembang Anda. Anda belum mengevaluasi risiko adopsi pustaka ini dari sumber tak dikenal dan Anda tidak tahu jika pustaka ini memiliki kerentanan atau kode berbahaya.
- Pembenaran bisnis asli untuk migrasi Anda didasarkan pada modernisasi 60% beban kerja aplikasi Anda. Namun demikian, karena hambatan teknis, akhirnya diputuskan untuk melakukan modernisasi terhadap hanya 20% beban kerja tersebut, yang mengakibatkan berkurangnya manfaat yang direncanakan dalam jangka panjang, bertambahnya beban operasional bagi tim infrastruktur untuk mendukung sistem warisan secara manual, dan ketergantungan yang lebih besar pada pengembangan keterampilan baru di dalam tim infrastruktur yang tidak merencanakan perubahan ini.

Manfaat menerapkan praktik terbaik ini: Sepenuhnya menyelaraskan dan mendukung prioritas bisnis tingkat dewan, memahami risiko untuk mencapai kesuksesan, membuat keputusan yang tepat berdasarkan informasi, dan bertindak dengan tepat ketika ada risiko yang menghambat peluang untuk mencapai kesuksesan. Memahami implikasi dan konsekuensi keputusan yang Anda ambil akan membantu Anda dalam menyusun prioritas opsi dan menghadirkan kesepakatan para pemimpin dengan lebih cepat, yang akan mengarahkan Anda pada hasil bisnis yang lebih baik. Mengidentifikasi manfaat-manfaat yang bisa dihadirkan oleh pilihan Anda dan menyadari risiko yang ditimbulkannya terhadap organisasi akan membantu Anda mengambil keputusan dengan berlandaskan data, bukan opini.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Manajemen manfaat dan risiko harus ditentukan oleh badan pengatur yang mendorong persyaratan untuk pengambilan keputusan utama. Anda ingin keputusan diambil dan diprioritaskan berdasarkan bagaimana keputusan tersebut menguntungkan organisasi, dengan memahami risiko-risiko yang mungkin dihadapi. Informasi yang akurat sangat penting untuk mengambil keputusan organisasi.

Hal ini harus didasarkan pada pengukuran yang solid dan ditentukan oleh praktik industri umum mengenai analisis manfaat biaya. Untuk mengambil keputusan semacam ini, bangun keseimbangan antara otoritas tersentralisasi dan terdesentralisasi. Kompromi akan selalu ada, dan penting untuk memahami bagaimana setiap pilihan akan memengaruhi strategi yang ditentukan dan hasil bisnis yang diinginkan.

Langkah-langkah implementasi

1. Susun praktik pengukuran manfaat dalam sebuah kerangka kerja tata kelola cloud yang menyeluruh.
   a. Seimbangkan kontrol pengambilan keputusan tersentralisasi dengan otoritas terdesentralisasi untuk beberapa keputusan.
   b. Pahami bahwa proses pengambilan keputusan yang memberatkan yang diterapkan pada setiap keputusan akan memperlambat Anda.
   c. Sertakan faktor-faktor eksternal ke dalam proses pengambilan keputusan Anda (seperti persyaratan kepatuhan).
2. Tetapkan kerangka kerja pengambilan keputusan yang telah disepakati untuk berbagai tingkat keputusan, yang menyertakan orang yang diminta untuk menengahi keputusan yang mengalami benturan kepentingan.
   a. Pusatkan pengambilan keputusan satu arah yang tidak dapat dibatalkan.
   b. Izinkan pengambilan keputusan dua arah oleh pemimpin organisasi tingkat bawah.
3. Pahami dan kelola manfaat serta risiko. Seimbangkan manfaat keputusan sesuai risiko yang terlibat.
   a. Identifikasi manfaat: Identifikasi manfaat berdasarkan tujuan, kebutuhan, dan prioritas bisnis. Contohnya termasuk dampak kasus bisnis time-to-market, keamanan, keandalan, kinerja, dan biaya.
   b. Identifikasi risiko: Identifikasi risiko berdasarkan tujuan, kebutuhan, dan prioritas bisnis. Contohnya termasuk time-to-market, keamanan, keandalan, kinerja, dan biaya.
   c. Evaluasi manfaat dibandingkan risiko dan ambil keputusan yang bijaksana: Tentukan dampak manfaat dan risiko berdasarkan tujuan, kebutuhan, dan prioritas pemangku kepentingan utama Anda, termasuk bagian bisnis, pengembangan, dan operasi. Lakukan evaluasi terhadap nilai dari manfaat yang didapatkan dibanding dengan probabilitas terjadinya risiko dan kerugian yang ditimbulkan oleh dampaknya. Misalnya, menekankan speed-to-market pada keandalan mungkin memberikan keunggulan kompetitif. Tetapi, hal ini dapat mengakibatkan berkurangnya waktu aktif jika ada masalah keandalan yang terjadi.

4. Secara terprogram, berlakukan keputusan utama yang mengotomatiskan kesesuaian Anda terhadap persyaratan-persyaratan kepatuhan.
5. Manfaatkan kerangka kerja dan kemampuan industri yang dikenal, seperti Analisis Aliran Nilai danLEAN, untuk mendasarkan kinerja keadaan saat ini, metrik bisnis, dan menentukan iterasi kemajuan menuju peningkatan metrik ini.

Tingkat upaya untuk rencana implementasi: Sedang cenderung Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP05 Mengevaluasi lanskap ancaman

Dokumen terkait:

- Elemen-elemen Amazon, Hari 1 Budaya | Buat keputusan berkualitas tinggi dan berkecepatan tinggi
- Tata Kelola Cloud
- Manajemen dan Tata Kelola Lingkungan Cloud
- Tata Kelola di Cloud dan di Era Digital: Bagian Satu & Dua

Video terkait:

- Siniar | Jeff Bezos | Tentang cara membuat keputusan

Contoh terkait:

- Membuat keputusan berdasarkan informasi menggunakan data (The DevOps Sagas)
- Menggunakan pemetaan aliran nilai pengembangan untuk mengidentifikasi kendala hasil DevOps

## OPS2. Bagaimana cara menyusun struktur organisasi untuk mendukung hasil bisnis Anda?

Tim Anda harus memahami peran mereka dalam mencapai hasil bisnis. Tim harus memahami peran mereka dalam kesuksesan tim lain, peran tim lain dalam kesuksesan mereka, dan memiliki

sasaran bersama. Memahami tanggung jawab, kepemilikan, bagaimana keputusan diambil, dan siapa yang memiliki otoritas untuk mengambil keputusan akan membantu memfokuskan upaya dan memaksimalkan manfaat dari tim Anda.

Praktik terbaik

- OPS02-BP01 Sumber daya telah mengidentifikasi pemilik
- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik
- OPS02-BP03 Kegiatan operasi telah mengidentifikasi pemilik yang bertanggung jawab atas kinerjanya
- OPS02-BP04 Mekanisme ada untuk mengelola tanggung jawab dan kepemilikan
- OPS02-BP05 Mekanisme ada untuk meminta penambahan, perubahan, dan pengecualian
- OPS02-BP06 Tanggung jawab antar tim telah ditentukan sebelumnya atau dinegosiasikan

OPS02-BP01 Sumber daya telah mengidentifikasi pemilik

Sumber daya untuk beban kerja Anda harus memiliki pemilik yang teridentifikasi untuk pengontrolan perubahan, penyelesaian masalah, dan fungsi-fungsi lainnya. Pemilik ditetapkan untuk beban kerja, akun, infrastruktur, platform, dan aplikasi. Kepemilikan dicatat menggunakan alat seperti daftar sentral atau metadata yang dilampirkan ke sumber daya. Nilai bisnis komponen menginformasikan proses dan prosedur yang diterapkan kepadanya.

Hasil yang diinginkan:

- Sumber daya telah mengidentifikasi pemilik dengan menggunakan metadata atau daftar sentral.
- Anggota tim dapat mengidentifikasi siapa pemilik sumber daya.
- Akun memiliki satu pemilik apabila mungkin.

Anti-pola umum:

- Kontak alternatif untuk Akun AWS Anda tidak diisi.
- Sumber daya tidak memiliki tag yang mengidentifikasi tim mana yang memilikinya.
- Anda memiliki ITSM antrian tanpa pemetaan email.
- Dua tim sama-sama merupakan pemilik bagian penting dari infrastruktur.

Manfaat menjalankan praktik terbaik ini:

- Kontrol perubahan untuk sumber daya akan menjadi mudah dilakukan dengan ditetapkannya kepemilikan.
- Anda dapat melibatkan pemilik yang benar ketika menyelesaikan masalah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Tentukan pentingnya kepemilikan untuk kasus penggunaan sumber daya di lingkungan Anda. Kepemilikan dapat berarti siapa yang mengawasi perubahan pada sumber daya, yang mendukung sumber daya selama penyelesaian masalah, atau siapa yang bertanggung jawab terhadapnya secara finansial. Sebutkan dan catat pemilik untuk sumber daya, termasuk nama, informasi kontak, organisasi, serta tim.

Contoh pelanggan

AnyCompany Ritel mendefinisikan kepemilikan sebagai tim atau individu yang memiliki perubahan dan dukungan untuk sumber daya. Mereka memanfaatkan AWS Organizations untuk mengelola mereka Akun AWS. Kontak akun alternatif dikonfigurasi menggunakan kotak masuk grup. Setiap ITSM antrian memetakan ke alias email. Tag mengidentifikasi siapa yang memiliki AWS sumber daya. Untuk infrastruktur dan platform-platform lainnya, mereka memiliki halaman wiki yang mengidentifikasi kepemilikan dan informasi kontak.

Langkah-langkah implementasi

1. Mulai dengan menentukan kepemilikan untuk organisasi Anda. Kepemilikan dapat menyiratkan siapa yang memiliki risiko untuk sumber daya, siapa yang memiliki perubahan pada sumber daya, atau siapa yang mendukung sumber daya ketika melakukan penyelesaian masalah. Kepemilikan juga dapat menyiratkan kepemilikan sumber daya secara finansial atau administratif.
2. Gunakan AWS Organizations untuk mengelola akun. Anda dapat mengelola kontak alternatif untuk akun Anda secara terpusat.
    a. Penggunaan alamat email dan nomor telepon milik perusahaan untuk informasi kontak akan membantu Anda untuk mengaksesnya meskipun orang yang memilikinya sudah tidak bekerja di organisasi Anda. Misalnya, buatlah daftar distribusi email terpisah untuk penagihan, operasional, dan keamanan lalu konfigurasikan ketiganya sebagai kontak Penagihan, Keamanan, dan Operasional di masing-masing Akun AWS yang aktif. Beberapa orang akan menerima AWS pemberitahuan dan dapat merespons, bahkan jika seseorang sedang berlibur, mengubah peran, atau meninggalkan perusahaan.

   b. Jika akun tidak dikelola oleh AWS Organizations, kontak akun alternatif dapat membantu AWS menghubungi personel yang tepat jika diperlukan. Konfigurasikan kontak alternatif akun sehingga menunjuk ke sebuah grup, bukan ke individu perseorangan.

3. Gunakan tag untuk mengidentifikasi pemilik AWS sumber daya. Anda dapat menentukan pemilik maupun informasi kontak mereka dalam tag terpisah.
   a. Anda dapat menggunakan aturan AWS Config untuk menegaskan bahwa sumber daya memiliki tag kepemilikan yang diperlukan.
   b. Untuk panduan mendalam tentang cara membangun strategi pemberian tag untuk organisasi Anda, silakan lihat laporan resmi mengenai Praktik Terbaik Pemberian Tag AWS.
4. Gunakan Amazon Q Business, sebuah asisten percakapan yang menggunakan AI generatif untuk meningkatkan produktivitas tenaga kerja, menjawab pertanyaan, dan menyelesaikan tugas berdasarkan informasi dalam sistem perusahaan Anda.
   a. Hubungkan Amazon Q Business ke sumber data perusahaan Anda. Amazon Q Business menawarkan konektor bawaan ke lebih dari 40 sumber data yang didukung, termasuk Amazon Simple Storage Service (Amazon S3), SharePoint Microsoft, Salesforce, dan Atlassian Confluence. Untuk informasi selengkapnya, silakan lihat Konektor Amazon Q.
5. Untuk sumber daya, platform, dan infrastruktur lainnya, buatlah dokumentasi yang mengidentifikasi kepemilikan. Dokumentasi ini harus dapat diakses oleh semua anggota tim.

Tingkat upaya untuk rencana implementasi: Rendah. Manfaatkan informasi kontak akun dan tag untuk menetapkan kepemilikan AWS sumber daya. Untuk sumber daya lain, Anda dapat menggunakan sesuatu yang sederhana seperti tabel di wiki untuk mencatat kepemilikan dan informasi kontak, atau menggunakan ITSM alat untuk memetakan kepemilikan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik
- OPS02-BP04 Mekanisme ada untuk mengelola tanggung jawab dan kepemilikan

Dokumen terkait:

- Manajemen Akun AWS - Memperbarui informasi kontak
- AWS Organizations - Memperbarui kontak alternatif di organisasi Anda

- Laporan resmi Praktik Terbaik Pemberian Tag AWS
- Bangun aplikasi AI generatif perusahaan pribadi dan aman dengan Amazon Q Business and AWS IAM Identity Center
- Amazon Q Business, sekarang tersedia secara umum dan dapat membantu Anda meningkatkan produktivitas tenaga kerja dengan AI generatif
- AWS Cloud Blog Operasi & Migrasi - Menerapkan kontrol penandaan otomatis dan terpusat dengan dan AWS ConfigAWS Organizations
- AWS Blog Keamanan - Perpanjang kait pra-komit Anda dengan AWS CloudFormation Guard
- AWS DevOps Blog - Mengintegrasikan AWS CloudFormation Guard ke dalam pipa CI/CD

Lokakarya terkait:

- Lokakarya - Pemberian Tag AWS

Contoh terkait:

- Aturan AWS Config - Amazon EC2 dengan tag yang diperlukan dan nilai yang valid

Layanan terkait:

- Aturan AWS Config - tag yang dibutuhkan
- AWS Organizations

OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik

Pahami siapa pemegang kepemilikan atas definisi dari masing-masing proses dan prosedur, alasan prosedur dan proses tertentu digunakan, serta alasan adanya kepemilikan tersebut. Dengan memahami alasan untuk menggunakan proses dan prosedur tertentu, peluang pengembangan dapat lebih mudah diidentifikasi.

Hasil yang diinginkan: Organisasi Anda memiliki serangkaian proses dan prosedur yang terdefinisi dengan baik dan terpelihara untuk tugas-tugas operasional. Proses dan prosedur-prosedur tersebut disimpan di lokasi terpusat dan tersedia untuk anggota tim Anda. Proses dan prosedur-prosedur sering diperbarui, dengan kepemilikan yang ditetapkan dengan jelas. Jika memungkinkan, skrip, templat, dan dokumen otomatisasi diimplementasikan sebagai kode.

Anti-pola umum:

- Proses tidak didokumentasikan. Mungkin terdapat skrip yang terfragmentasi di stasiun kerja operator yang terisolasi.
- Pengetahuan tentang cara menggunakan skrip dipegang oleh beberapa individu perorangan atau secara informal sebagai pengetahuan tim.
- Proses warisan sudah harus diperbarui, tetapi kepemilikan pembaruan masih tidak jelas, dan penulis aslinya sudah bukan bagian dari organisasi.
- Proses dan skrip tidak dapat ditemukan, sehingga tidak tersedia saat diperlukan (misalnya, dalam merespons sebuah insiden).

Manfaat menjalankan praktik terbaik ini:

- Proses dan prosedur-prosedur akan meningkatkan upaya Anda untuk mengoperasikan beban kerja Anda.
- Anggota tim baru menjadi lebih efektif dengan lebih cepat.
- Mengurangi waktu mitigasi insiden.
- Anggota tim (dan tim) yang berbeda dapat menggunakan proses dan prosedur yang sama secara konsisten.
- Tim dapat menskalakan proses mereka dengan proses yang dapat diulang.
- Proses dan prosedur standar membantu mengurangi dampak pengalihan tanggung jawab beban kerja antar-tim.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Proses dan prosedur-prosedur memiliki pemilik yang jelas untuk bertanggung jawab atas penetapannya.
  - Identifikasi aktivitas operasi yang dijalankan untuk mendukung beban kerja Anda. Buatlah dokumentasi dari aktivitas ini di lokasi yang mudah ditemukan.
  - Identifikasi secara khusus individu atau tim yang bertanggung jawab atas spesifikasi dari sebuah aktivitas. Mereka bertanggung jawab untuk memverifikasi bahwa aktivitas dapat dijalankan dengan sukses oleh anggota tim yang memiliki keterampilan memadai serta memiliki izin, akses, serta alat yang sesuai. Jika terdapat masalah saat menjalankan aktivitas tersebut, maka anggota

tim yang menjalankannya bertanggung jawab untuk memberikan umpan balik mendetail yang diperlukan agar aktivitas tersebut dapat ditingkatkan.

- Tangkap kepemilikan dalam metadata artefak aktivitas melalui layanan seperti AWS Systems Manager, melalui dokumen, dan. AWS Lambda Rekam kepemilikan sumber daya menggunakan grup sumber daya atau tag, yang menentukan informasi kontak dan kepemilikan. Gunakan AWS Organizations untuk membuat kebijakan penandaan dan menangkap kepemilikan dan informasi kontak.

- Seiring waktu, prosedur ini harus dikembangkan agar dapat dijalankan sebagai kode, sehingga mengurangi kebutuhan akan campur tangan manusia.
  - Misalnya, pertimbangkan AWS Lambda fungsi, CloudFormation templat, atau dokumen otomatisasi AWS Systems Manager.
  - Jalankan kontrol versi di repositori yang sesuai.
  - Sertakan tag sumber daya yang sesuai sehingga pemilik dan dokumentasi dapat diidentifikasi dengan mudah.

Contoh pelanggan

AnyCompany Ritel mendefinisikan kepemilikan sebagai tim atau individu yang memiliki proses untuk aplikasi atau kelompok aplikasi (yang berbagi praktik dan teknologi arsitek umum). Awalnya, proses dan prosedur didokumentasikan sebagai step-by-step panduan dalam sistem manajemen dokumen, dapat ditemukan menggunakan tag pada Akun AWS yang menghosting aplikasi dan pada kelompok sumber daya tertentu dalam akun. Mereka memanfaatkan AWS Organizations untuk mengelola mereka Akun AWS. Seiring waktu, proses ini diubah menjadi kode, dan sumber daya didefinisikan menggunakan infrastruktur sebagai kode (seperti CloudFormation atau AWS Cloud Development Kit (AWS CDK) templat). Proses operasional menjadi dokumen otomatisasi dalam AWS Systems Manager atau AWS Lambda fungsi, yang dapat dimulai sebagai tugas terjadwal, dalam menanggapi peristiwa seperti AWS CloudWatch alarm atau AWS EventBridge peristiwa, atau dimulai oleh permintaan dalam platform manajemen layanan TI (ITSM). Semua proses memiliki tag untuk mengidentifikasi kepemilikan. Dokumentasi untuk otomatisasi dan proses dipertahankan di halaman wiki yang dihasilkan oleh repositori kode untuk proses tersebut.

Langkah-langkah implementasi

1. Dokumentasikan proses dan prosedur yang ada.
   a. Tinjau dan simpan up-to-date.
   b. Identifikasi pemilik untuk setiap proses atau prosedur.

   c. Tempatkan mereka di bawah kontrol versi.
   d. Jika memungkinkan, bagikan proses dan prosedur-prosedur itu di seluruh beban kerja dan lingkungan yang berbagi desain arsitektur.
2. Buat mekanisme untuk umpan balik dan perbaikan.
   a. Tentukan kebijakan untuk frekuensi peninjauan proses.
   b. Tentukan proses untuk peninjau dan pemberi persetujuan.
   c. Implementasikan permasalahan-permasalahan atau antrean tiket untuk umpan balik yang akan diberikan dan dilacak.
   d. Jika memungkinkan, proses dan prosedur harus memiliki pra-persetujuan dan klasifikasi risiko dari dewan persetujuan perubahan ()CAB.
3. Verifikasi bahwa proses dan prosedur dapat diakses dan ditemukan oleh orang-orang yang perlu menjalankannya.
   a. Gunakan tag untuk menunjukkan di mana proses dan prosedur dapat diakses untuk beban kerja.
   b. Gunakan pesan kesalahan dan peristiwa yang dapat dipahami untuk menunjukkan proses atau prosedur yang sesuai untuk mengatasi sebuah permasalahan.
   c. Gunakan wiki dan manajemen dokumen, dan jadikan proses dan prosedur dapat dicari secara konsisten di seluruh organisasi.
4. Lakukan otomatisasi jika perlu.
   a. Otomasi harus dikembangkan ketika layanan dan teknologi menyediakan. API
   b. Berikan edukasi secara memadai tentang proses. Kembangkan kisah dan persyaratan pengguna untuk mengotomatiskan proses-proses tersebut.
   c. Ukur penggunaan proses dan prosedur Anda dengan sukses, dengan masalah-masalah untuk mendukung perbaikan berulang.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP01 Sumber daya telah mengidentifikasi pemilik
- OPS02-BP04 Mekanisme ada untuk mengelola tanggung jawab dan kepemilikan
- OPS11-BP04 Melakukan manajemen pengetahuan

Dokumen terkait:

- AWS Whitepaper - Pengantar pada DevOps AWS
- AWS Whitepaper - Praktik Terbaik untuk Menandai Sumber Daya AWS
- AWS Whitepaper - Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun
- AWS Cloud Blog Operasi & Migrasi - Bangun Praktik Otomasi Cloud untuk Keunggulan Operasional: Praktik Terbaik dari AWS Managed Services
- AWS Cloud Blog Operasi & Migrasi - Menerapkan kontrol penandaan otomatis dan terpusat dengan dan AWS ConfigAWS Organizations
- AWS Blog Keamanan - Perpanjang kait pra-komit Anda dengan AWS CloudFormation Guard
- AWS DevOps Blog - Mengintegrasikan AWS CloudFormation Guard ke dalam pipa CI/CD

Lokakarya terkait:

- Lokakarya Keunggulan Operasional Well-Architected AWS
- Lokakarya - Pemberian Tag AWS

Video terkait:

- Cara mengotomatiskan Operasi TI di AWS
- AWS Re:invent 2020 - Otomatiskan apa pun dengan Systems Manager AWS
- AWS Re: Inforce 2022 - Mengotomatiskan manajemen patch dan kepatuhan menggunakan (06) AWS NIS3
- AWS Support s You - Menyelam Jauh ke dalam AWS Systems Manager

Layanan terkait:

- AWS Systems Manager - Otomasi
- AWS Service Management Connector

OPS02-BP03 Kegiatan operasi telah mengidentifikasi pemilik yang bertanggung jawab atas kinerjanya

Pahami siapa yang bertanggung jawab untuk menjalankan aktivitas tertentu terhadap beban kerja yang ditentukan serta alasan adanya tanggung jawab tersebut. Memahami siapa yang bertanggung

jawab untuk menjalankan aktivitas dapat memberikan informasi tentang siapa yang akan melakukan aktivitas tersebut, memvalidasi hasilnya, serta memberikan umpan balik kepada pemilik aktivitas.

Hasil yang diinginkan:

Organisasi Anda secara jelas menetapkan tanggung jawab untuk menjalankan aktivitas tertentu pada beban kerja yang ditentukan dan merespons peristiwa-peristiwa yang dihasilkan oleh beban kerja tersebut. Organisasi mendokumentasikan kepemilikan proses dan pemenuhan dan membuat informasi ini dapat ditemukan. Anda meninjau dan memperbarui tanggung jawab ketika ada perubahan yang terjadi pada organisasi, dan tim melacak serta mengukur kinerja aktivitas identifikasi kekurangan dan inefisiensi. Anda mengimplementasikan mekanisme umpan balik untuk melacak kekurangan dan perbaikan serta mendukung perbaikan berulang.

Anti-pola umum:

- Anda tidak mendokumentasikan tanggung jawab.
- Terdapat skrip yang terfragmentasi di stasiun kerja operator yang terisolasi. Hanya sedikit orang saja yang tahu cara menggunakannya atau secara informal menyebutnya sebagai pengetahuan tim.
- Proses warisan sudah harus diperbarui, tetapi tidak ada yang tahu siapa yang memiliki proses tersebut, dan penulis aslinya sudah tidak lagi menjadi bagian dari organisasi.
- Proses dan skrip tidak dapat ditemukan, dan tidak tersedia saat diperlukan (misalnya, saat merespons insiden).

Manfaat menjalankan praktik terbaik ini:

- Anda memahami siapa yang bertanggung jawab untuk menjalankan sebuah aktivitas, siapa yang harus mendapatkan notifikasi saat diperlukan tindakan, dan siapa yang melakukan tindakan, memvalidasi hasilnya, serta memberikan umpan balik kepada pemilik aktivitas tersebut.
- Proses dan prosedur-prosedur akan meningkatkan upaya Anda untuk mengoperasikan beban kerja Anda.
- Anggota tim baru menjadi lebih efektif dengan lebih cepat.
- Anda mengurangi waktu yang dibutuhkan untuk memitigasi insiden.
- Tim yang berbeda menggunakan proses dan prosedur yang sama untuk melakukan tugas-tugas secara konsisten.
- Tim dapat menskalakan proses mereka dengan proses yang dapat diulang.

- Proses dan prosedur standar membantu mengurangi dampak pengalihan tanggung jawab beban kerja antar-tim.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk mulai menentukan tanggung jawab, mulailah dengan dokumentasi yang sudah ada sekarang, seperti matriks tanggung jawab, proses dan prosedur, peran dan tanggung jawab, serta alat-alat dan otomatisasi. Tinjau dan lakukan diskusi mengenai tanggung jawab untuk proses yang terdokumentasi. Lakukan peninjauan bersama tim untuk mengidentifikasi ketidakselarasan antara tanggung jawab dokumen dan proses. Diskusikan layanan-layanan yang ditawarkan dengan pelanggan internal tim tersebut untuk mengidentifikasi perbedaan ekspektasi di antara tim.

Analisis dan atasi perbedaan. Identifikasi setiap peluang perbaikan, dan cari aktivitas padat sumber daya yang sering diminta, yang biasanya merupakan kandidat kuat untuk perbaikan. Jelajahi praktik terbaik, pola, dan panduan preskriptif untuk menyederhanakan dan melakukan standardisasi perbaikan. Dokumentasikan peluang-peluang perbaikan, dan lacak perbaikan hingga selesai.

Seiring waktu, prosedur-prosedur ini harus dikembangkan agar dapat dijalankan sebagai kode, sehingga akan mengurangi kebutuhan akan campur tangan manusia. Misalnya, prosedur dapat dimulai sebagai AWS Lambda fungsi, AWS CloudFormation templat, atau dokumen AWS Systems Manager Otomasi. Pastikan bahwa semua prosedur ini memiliki kontrol versi di repositori yang sesuai, dan sertakan tag sumber daya yang sesuai sehingga tim dapat mengidentifikasi pemilik dan dokumentasi dengan mudah. Buatlah dokumentasi tanggung jawab untuk melaksanakan aktivitas, kemudian pantau otomatisasi untuk inisiasi dan operasi yang berhasil, serta kinerja hasil yang diinginkan.

Contoh pelanggan

AnyCompany Ritel mendefinisikan kepemilikan sebagai tim atau individu yang memiliki proses untuk aplikasi atau kelompok aplikasi yang berbagi praktik arsitektur dan teknologi umum. Awalnya, perusahaan mendokumentasikan proses dan prosedur sebagai step-by-step panduan dalam sistem manajemen dokumen. Mereka membuat prosedur dapat ditemukan menggunakan tag pada Akun AWS yang meng-host aplikasi dan pada kelompok sumber daya tertentu dalam akun, menggunakan AWS Organizations untuk mengelola mereka. Akun AWS Seiring waktu, AnyCompany Retail mengubah proses ini menjadi kode dan mendefinisikan sumber daya menggunakan infrastruktur sebagai kode (melalui layanan seperti CloudFormation atau AWS Cloud Development Kit (AWS

CDK) templat). Proses operasional menjadi dokumen Otomasi di AWS Systems Manager atau AWS Lambda fungsi, yang dapat dimulai sebagai tugas terjadwal dalam menanggapi peristiwa seperti CloudWatch alarm Amazon atau EventBridge peristiwa Amazon atau oleh permintaan dalam platform manajemen layanan TI (ITSM). Semua proses memiliki tag untuk mengidentifikasi pemiliknya. Tim mengelola dokumentasi untuk otomatisasi dan proses di dalam halaman wiki yang dihasilkan oleh repositori kode untuk proses tersebut.

Langkah-langkah implementasi

1. Dokumentasikan proses dan prosedur yang ada.
    a. Tinjau dan verifikasi bahwa mereka up-to-date.
    b. Pastikan bahwa setiap proses atau prosedur mempunyai pemilik.
    c. Tempatkan prosedur di bawah kontrol versi.
    d. Jika memungkinkan, bagikan proses dan prosedur-prosedur itu di seluruh beban kerja dan lingkungan yang berbagi desain arsitektur.
2. Buat mekanisme untuk umpan balik dan perbaikan.
    a. Tentukan kebijakan untuk frekuensi peninjauan proses.
    b. Tentukan proses untuk peninjau dan pemberi persetujuan.
    c. Implementasikan permasalahan-permasalahan atau antrean tiket untuk memberikan dan melacak umpan balik.
    d. Jika memungkinkan, berikan pra-persetujuan dan klasifikasi risiko untuk proses dan prosedur dari dewan persetujuan perubahan (CAB).
3. Buat proses dan prosedur dapat diakses dan ditemukan oleh pengguna yang perlu menjalankannya.
    a. Gunakan tag untuk menunjukkan di mana proses dan prosedur dapat diakses untuk beban kerja.
    b. Gunakan pesan kesalahan dan peristiwa yang dapat dipahami untuk menunjukkan proses atau prosedur yang sesuai untuk mengatasi masalah.
    c. Gunakan wiki atau manajemen dokumen agar proses dan prosedur dapat dicari secara konsisten di seluruh organisasi.
4. Lakukan otomatisasi jika perlu.
    a. Di mana layanan dan teknologi menyediakanAPI, kembangkan otomatisasi.
    b. Pastikan bahwa proses tersebut dapat dipahami dengan baik, dan kembangkan kisah serta persyaratan pengguna untuk mengotomatiskan proses tersebut.

c. Ukur penggunaan proses dan prosedur yang sukses, dengan pelacakan masalah untuk mendukung perbaikan berulang.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP01 Sumber daya telah mengidentifikasi pemilik
- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik
- OPS02-BP04 Mekanisme ada untuk mengelola tanggung jawab dan kepemilikan
- OPS02-BP05 Mekanisme ada untuk mengidentifikasi tanggung jawab dan kepemilikan
- OPS11-BP04 Melakukan manajemen pengetahuan

Dokumen terkait:

- AWS Whitepaper | Pengantar pada DevOps AWS
- AWS Whitepaper | Praktik Terbaik untuk Menandai Sumber Daya AWS
- AWS Whitepaper | Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun
- AWS Cloud Blog Operasi & Migrasi | Bangun Praktik Otomasi Cloud untuk Keunggulan Operasional: Praktik Terbaik dari AWS Managed Services
- Lokakarya - Pemberian Tag AWS
- Konektor Manajemen Layanan AWS

Video terkait:

- AWS Pusat Pengetahuan Langsung | Tagging Resources AWS
- AWS re: Invent 2020 | Otomatiskan apa pun dengan Systems Manager AWS
- AWS Re: Inforce 2022 | Mengotomatiskan manajemen patch dan kepatuhan menggunakan (06) AWS NIS3
- AWS Support S You | Menyelam Jauh ke dalam AWS Systems Manager

Contoh terkait:

- Lokakarya Keunggulan Operasional Well-Architected AWS

OPS02-BP04 Mekanisme ada untuk mengelola tanggung jawab dan kepemilikan

Pahami tanggung jawab yang dimiliki oleh peran Anda dan bagaimana Anda berkontribusi terhadap hasil bisnis, karena pemahaman ini akan mendasari penentuan prioritas tugas Anda dan mengapa peran Anda itu penting. Hal ini akan membantu anggota tim untuk mengenali kebutuhan dan merespons dengan tepat. Ketika anggota tim mengetahui peran mereka, mereka dapat membangun kepemilikan, mengidentifikasi peluang perbaikan, dan memahami cara untuk memengaruhi atau membuat perubahan yang sesuai.

Kadang-kadang, sebuah tanggung jawab mungkin tidak memiliki pemilik yang jelas. Dalam situasi seperti ini, rancang sebuah mekanisme untuk mengatasi kesenjangan ini. Buat jalur eskalasi yang ditentukan dengan baik kepada seseorang yang memiliki wewenang untuk menetapkan kepemilikan atau rencana untuk memenuhi kebutuhan tersebut.

Hasil yang diinginkan: Tim-tim yang ada dalam organisasi Anda memiliki tanggung jawab yang sudah ditentukan dengan jelas yang mencakup bagaimana kaitan mereka dengan sumber daya, tindakan yang harus dilakukan, proses, dan prosedur. Tanggung jawab ini selaras dengan tanggung jawab dan sasaran tim, serta tanggung jawab yang dimiliki tim lain. Anda mendokumentasikan rute eskalasi dengan cara yang konsisten dan dapat ditemukan dan memasukkan keputusan tersebut ke dalam artefak dokumentasi, misalnya matriks tanggung jawab, definisi tim, atau halaman wiki.

Anti-pola umum:

- Tanggung jawab tim yang bersifat ambigu atau tidak ditentukan dengan baik.
- Tim tidak menyelaraskan peran dengan tanggung jawab.
- Tim tidak menyelaraskan tujuan dan sasarannya dengan tanggung jawabnya, sehingga kesuksesan menjadi sulit diukur.
- Tanggung jawab anggota tim tidak selaras dengan tim dan organisasi yang lebih luas.
- Tim Anda tidak menyimpan tanggung jawab up-to-date, yang membuat mereka tidak konsisten dengan tugas yang dilakukan oleh tim.
- Jalur eskalasi untuk menentukan tanggung jawab tidak ditetapkan atau tidak jelas.
- Jalur eskalasi tidak memiliki pemilik utas tunggal untuk memastikan pemberian respons yang cepat.
- Peran, tanggung jawab, dan jalur eskalasi tidak dapat ditemukan, dan tidak tersedia saat diperlukan (misalnya, saat merespons insiden).

Manfaat menjalankan praktik terbaik ini:

- Ketika Anda memahami siapa yang memegang tanggung jawab atau kepemilikan, Anda dapat menghubungi tim atau anggota tim yang tepat untuk melakukan permintaan atau pengalihann tugas.
- Untuk mengurangi risiko tidak adanya tindakan dan kebutuhan yang tidak tertangani, Anda telah mengidentifikasi seseorang yang memiliki wewenang untuk menetapkan tanggung jawab atau kepemilikan.
- Ketika Anda mendefinisikan cakupan suatu tanggung jawab dengan jelas, anggota tim Anda mendapatkan otonomi dan kepemilikan.
- Tanggung jawab Anda akan mendasari keputusan yang Anda ambil, tindakan yang akan Anda lakukan, dan penyerahan aktivitas Anda ke pemiliknya yang benar.
- Tanggung jawab yang ditinggalkan dapat dengan mudah diidentifikasi karena Anda memiliki pemahaman yang jelas tentang hal-hal yang berada di luar tanggung jawab tim Anda, sehingga membantu Anda dalam melakukan eskalasi untuk meminta klarifikasi.
- Tim menghindari kebingungan dan ketegangan, dan mereka dapat mengelola beban kerja serta sumber daya dengan lebih memadai.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Identifikasi peran dan tanggung jawab anggota tim, dan pastikan bahwa mereka memahami apa yang diharapkan dari peran mereka. Buat informasi ini dapat ditemukan sehingga anggota-anggota organisasi Anda dapat mengidentifikasi siapa yang perlu mereka hubungi saat ada kebutuhan khusus, baik berupa tim atau perorangan. Ketika organisasi berusaha memanfaatkan peluang untuk bermigrasi dan memodernisasi AWS, peran dan tanggung jawab juga dapat berubah. Jaga agar tim Anda dan para anggotanya tetap menyadari tanggung jawab mereka, dan latih mereka dengan semestinya untuk melaksanakan tugas selama perubahan ini.

Tentukan peran atau tim yang harus menerima eskalasi untuk mengidentifikasi tanggung jawab dan kepemilikan. Tim ini dapat berinteraksi dengan berbagai pemangku kepentingan untuk mengambil suatu keputusan. Namun, mereka harus memiliki wewenang manajemen proses pengambilan keputusan.

Sediakan mekanisme yang dapat diakses bagi anggota organisasi untuk menemukan dan mengidentifikasi kepemilikan dan tanggung jawab. Mekanisme ini memberi tahu mereka siapa yang harus dihubungi saat ada kebutuhan khusus.

Contoh pelanggan

AnyCompany Retail baru-baru ini menyelesaikan migrasi beban kerja dari lingkungan lokal ke landing zone mereka AWS dengan pendekatan lift dan shift. Mereka melakukan peninjauan operasi untuk merenungkan cara mereka menyelesaikan tugas-tugas operasional umum dan memastikan bahwa matriks tanggung jawab mereka yang ada sekarang sudah sesuai dengan operasi yang ada di lingkungan baru. Ketika mereka bermigrasi dari lokal ke tempat AWS, mereka mengurangi tanggung jawab tim infrastruktur yang berkaitan dengan perangkat keras dan infrastruktur fisik. Langkah ini juga mengungkap adanya peluang baru untuk mengembangkan model operasi untuk beban kerja mereka.

Di saat mereka mengidentifikasi, menangani, dan mendokumentasikan sebagian besar tanggung jawab, mereka juga menetapkan rute eskalasi untuk tanggung jawab apa pun yang terlewatkan atau untuk tanggung jawab yang mungkin perlu diubah sesuai perkembangan praktik operasi. Untuk mengeksplorasi peluang baru untuk membakukan dan meningkatkan efisiensi di seluruh beban kerja Anda, sediakan akses ke alat operasi seperti AWS Systems Manager dan alat keamanan seperti dan Amazon. AWS Security Hub GuardDuty AnyCompanyRetail mengumpulkan tinjauan tanggung jawab dan strategi berdasarkan perbaikan yang ingin mereka tangani terlebih dahulu. Ketika perusahaan ini mengadopsi cara-cara kerja dan pola teknologi baru, mereka memperbarui matriks tanggung jawab mereka agar sesuai dengan itu semua.

Langkah-langkah implementasi

1. Mulailah dengan dokumentasi yang sudah ada. Beberapa dokumen sumber yang umum antara lain:
   a. Matriks tanggung jawab atau bertanggung jawab, akuntabel, dikonsultasikan, dan diinformasikan () RACI
   b. Definisi tim atau halaman wiki
   c. Definisi dan penawaran layanan
   d. Deskripsi peran atau pekerjaan
2. Tinjau dan lakukan diskusi tentang tanggung jawab yang didokumentasikan:
   a. Lakukan peninjauan bersama tim untuk mengidentifikasi ketidakselarasan yang terjadi antara tanggung jawab yang terdokumentasi dan tanggung jawab yang umumnya dijalankan oleh tim.

   b. Diskusikan layanan-layanan potensial yang ditawarkan oleh pelanggan internal untuk mengidentifikasi adanya perbedaan ekspektasi di antara tim.
3. Lakukan analisis dan atasi perbedaan.
4. Identifikasi peluang perbaikan.
   a. Identifikasi permintaan padat sumber daya yang sering kali mendapat permintaan, yang biasanya merupakan kandidat kuat untuk perbaikan.
   b. Cari praktik terbaik, pola, dan panduan preskriptif, serta lakukan penyederhanaan dan standardisasi perbaikan dengan panduan ini.
   c. Dokumentasikan peluang-peluang perbaikan, dan lacak hingga selesai.
5. Jika tim belum memiliki tanggung jawab untuk mengelola dan melacak penugasan tanggung jawab, identifikasi seseorang yang ada di dalam tim untuk memegang tanggung jawab ini.
6. Tentukan proses bagi tim untuk meminta klarifikasi tanggung jawab.
   a. Tinjau prosesnya, dan pastikan bahwa proses tersebut jelas dan mudah digunakan.
   b. Pastikan seseorang memiliki dan melacak proses eskalasi hingga selesai.
   c. Buat metrik operasional untuk mengukur efektivitas.
   d. Ciptakan sebuah mekanisme umpan balik untuk memastikan bahwa tim dapat menyoroti peluang-peluang perbaikan.
   e. Implementasikan sebuah mekanisme untuk peninjauan berkala.
7. Buatlah dokumentasi di sebuah lokasi yang dapat ditemukan dan dapat diakses.
   a. Wiki atau portal dokumentasi adalah pilihan umum.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP06 Mengevaluasi pengorbanan
- OPS03-BP02 Anggota tim diberdayakan untuk mengambil tindakan ketika hasil berisiko
- OPS03-BP03 Eskalasi didorong
- OPS03-BP07 Tim sumber daya dengan tepat
- OPS09-BP01 Mengukur tujuan operasi dan dengan metrik KPIs
- OPS09-BP03 Meninjau metrik operasi dan memprioritaskan peningkatan

- OPS11-BP01 Memiliki proses untuk perbaikan berkelanjutan

Dokumen terkait:

- AWS Whitepaper - Pengantar pada DevOps AWS
- AWS Whitepaper - Kerangka AWS Cloud Adopsi: Perspektif Operasi
- Keunggulan Operasional Kerangka Kerja AWS Well-Architected - Topologi model operasi Tingkat beban kerja
- Panduan Preskriptif AWS - Membangun Model Operasi Cloud Anda
- AWS Panduan Preskriptif - Buat RASCI matriks RACI atau untuk model operasi cloud
- AWS Cloud Blog Operasi & Migrasi - Memberikan Nilai Bisnis dengan Tim Platform Cloud
- AWS Cloud Blog Operasi & Migrasi - Mengapa Model Operasi Cloud?
- AWS DevOps Blog - Bagaimana organisasi memodernisasi untuk operasi cloud

Video terkait:

- Summit Online AWS - Model Operasi Cloud untuk Transformasi yang Dipercepat
- re:Invent 2023 AWS - Keamanan cloud yang tahan masa depan: Model operasi baru

OPS02-BP05 Mekanisme ada untuk meminta penambahan, perubahan, dan pengecualian

Anda dapat mengajukan permintaan kepada pemilik proses, prosedur, dan sumber daya. Permintaan tersebut mencakup penambahan, perubahan, dan pengecualian. Permintaan ini diajukan melalui sebuah proses manajemen perubahan. Buatlah keputusan-keputusan yang matang berdasarkan informasi untuk menyetujui permintaan apabila memungkinkan dan dianggap tepat setelah dilakukan evaluasi manfaat dan risiko.

Hasil yang diinginkan:

- Anda dapat mengajukan permintaan untuk mengubah proses, prosedur, dan sumber daya berdasarkan kepemilikan yang ditetapkan.
- Perubahan harus dibuat dengan penuh pertimbangan, dengan memikirkan manfaat dan risikonya.

Anti-pola umum:

- Anda harus memperbarui cara Anda melakukan deployment terhadap aplikasi Anda, tetapi perubahan proses deployment tidak dapat diminta dari tim operasi.
- Rencana pemulihan bencana harus diperbarui, tetapi tidak ada pemilik yang diidentifikasi untuk dapat dimintai perubahan.

Manfaat menjalankan praktik terbaik ini:

- Proses, prosedur, dan sumber daya dapat berubah seiring dengan terjadinya perubahan persyaratan.
- Pemilik dapat mengambil keputusan yang matang berdasarkan informasi ketika harus membuat perubahan.
- Perubahan harus dibuat dengan cara yang penuh pertimbangan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Untuk mengimplementasikan praktik terbaik ini, Anda harus dapat membuat permintaan perubahan proses, prosedur, dan sumber daya. Proses manajemen perubahan bisa jadi hal yang ringan. Buatlah dokumentasi proses manajemen perubahan.

Contoh pelanggan

AnyCompany Ritel menggunakan matriks penugasan tanggung jawab (RACI) untuk mengidentifikasi siapa yang memiliki perubahan untuk proses, prosedur, dan sumber daya. Mereka memiliki proses manajemen perubahan terdokumentasi yang ringan dan mudah diikuti. Dengan menggunakan RACI matriks dan prosesnya, siapa pun dapat mengirimkan permintaan perubahan.

Langkah-langkah implementasi

1. Identifikasi proses, prosedur, dan sumber daya untuk beban kerja Anda dan pemilik untuk masing-masing. Buatlah dokumentasi tentang itu semua dalam sistem manajemen pengetahuan Anda.
   a. Jika Anda belum menerapkan OPS02-BP01 Sumber daya telah mengidentifikasi pemilik, OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik, atau OPS02-BP03 Kegiatan operasi telah mengidentifikasi pemilik yang bertanggung jawab atas kinerjanya, mulailah dengan yang pertama.

2. Bekerjasamalah dengan para pemangku kepentingan yang ada di organisasi Anda untuk mengembangkan sebuah proses manajemen perubahan. Proses tersebut harus meliputi penambahan, perubahan, dan pengecualian untuk sumber daya, proses, dan prosedur.
   a. Anda dapat menggunakan Manajer Perubahan AWS Systems Manager sebagai sebuah platform manajemen perubahan untuk sumber daya beban kerja.
3. Buatlah dokumentasi proses manajemen perubahan dalam sistem manajemen pengetahuan Anda.

Tingkat upaya untuk rencana implementasi: Sedang. Mengembangkan sebuah proses manajemen perubahan memerlukan penyelarasan dengan banyak pemangku kepentingan yang ada di seluruh organisasi Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP01 Sumber daya telah mengidentifikasi pemilik - Sumber daya membutuhkan pemilik yang teridentifikasi sebelum Anda membangun sebuah proses manajemen perubahan.
- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik - Proses membutuhkan pemilik yang teridentifikasi sebelum Anda membangun sebuah proses manajemen perubahan.
- OPS02-BP03 Kegiatan operasi telah mengidentifikasi pemilik yang bertanggung jawab atas kinerjanya - Kegiatan Operasi membutuhkan pemilik yang teridentifikasi sebelum Anda membangun sebuah proses manajemen perubahan.

Dokumen terkait:

- AWS Panduan Preskriptif - Buku palybook dasar untuk migrasi AWS besar: Membuat matriks RACI
- Manajemen Perubahan dalam Laporan Resmi Cloud

Layanan terkait:

- AWS Systems Manager Ubah Manajer

OPS02-BP06 Tanggung jawab antar tim telah ditentukan sebelumnya atau dinegosiasikan

Miliki perjanjian yang telah ditetapkan atau dinegosiasikan antara tim yang menjelaskan bagaimana mereka akan bekerja sama dan saling mendukung satu sama lain (contohnya, waktu respons, tujuan

tingkat layanan, atau perjanjian tingkat layanan). Saluran komunikasi antar-tim didokumentasikan. Memahami dampak dari pekerjaan tim terhadap hasil bisnis, dan hasil dari tim dan organisasi yang lain akan membuat Anda tahu tentang penentuan prioritas tugas mereka dan membantu mereka merespons dengan tepat.

Ketika ada tanggung jawab dan kepemilikan yang tidak ditetapkan atau tidak diketahui, maka Anda akan menanggung risiko tidak menangani aktivitas yang diperlukan secara tepat waktu serta risiko munculnya upaya yang berulang dan kemungkinan bertentangan untuk menangani kebutuhan-kebutuhan tersebut.

Hasil yang diinginkan:

- Perjanjian bekerja atau mendukung antar-tim sudah disetujui dan didokumentasikan.
- Tim-tim yang mendukung atau bekerja dengan satu sama lain memiliki ekspektasi respons dan saluran komunikasi yang telah ditetapkan sebelumnya.

Anti-pola umum:

- Ada sebuah masalah yang terjadi dalam produksi dan dua tim terpisah mulai menyelesaikan masalahnya sendiri-sendiri. Upaya terpisah mereka memperpanjang masa henti produksi.
- Tim operasi membutuhkan bantuan dari tim pengembangan, tetapi tidak ada waktu respons yang disepakati. Permintaannya tetap berada dalam timbunan yang belum dikerjakan (backlog).

Manfaat menjalankan praktik terbaik ini:

- Tim mengetahui cara berinteraksi dan mendukung satu sama lain.
- Ekspektasi untuk tingkat responsivitas sudah diketahui.
- Saluran komunikasi sudah ditetapkan dengan jelas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Dengan mengimplementasikan praktik terbaik ini, artinya bahwa sudah tidak ada lagi ambiguitas tentang bagaimana tim bekerja dengan satu sama lain. Perjanjian resmi mengatur tentang bagaimana tim bekerja sama atau mendukung satu sama lain. Saluran komunikasi antar-tim sudah didokumentasikan.

Contoh pelanggan

AnyCompany SRETim Retail memiliki perjanjian tingkat layanan dengan tim pengembangan mereka. Setiap kali tim pengembangan mengajukan sebuah permintaan dalam sistem tiket mereka, mereka dapat mengantisipasi bahwa respons akan diterima dalam waktu lima belas menit. Jika ada pemadaman situs, SRE tim memimpin dalam penyelidikan dengan dukungan dari tim pengembangan.

Langkah-langkah implementasi

1. Melalui kerja sama dengan para pemangku kepentingan yang ada di seluruh organisasi Anda, buatlah perjanjian antara tim berdasarkan proses dan prosedur.
   a. Jika sebuah proses atau prosedur dimiliki bersama antara dua tim, kembangkan sebuah runbook tentang cara tim akan bekerja sama.
   b. Jika ada dependensi antar tim, setujui respons SLA untuk permintaan.
2. Buatlah dokumentasi tanggung jawab dalam sistem manajemen pengetahuan Anda.

Tingkat upaya untuk rencana implementasi: Sedang. Jika belum ada perjanjian yang dibuat antara tim, mungkin akan diperlukan upaya agar para pemangku kepentingan di seluruh organisasi Anda bisa sepakat.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik - Kepemilikan proses harus diidentifikasi sebelum menetapkan perjanjian antar tim.
- OPS02-BP03 Kegiatan operasi telah mengidentifikasi pemilik yang bertanggung jawab atas kinerjanya - Kepemilikan kegiatan operasi harus diidentifikasi sebelum menetapkan perjanjian antar tim.

Dokumen terkait:

- AWS Executive Insights - Memberdayakan Inovasi dengan Tim Dua Pizza
- Pengantar DevOps tentang AWS - Tim Dua Pizza

## OPS3. Bagaimana budaya organisasi Anda mendukung hasil bisnis Anda?

Berikan dukungan kepada anggota tim Anda sehingga mereka dapat menjadi lebih efektif dalam mengambil tindakan dan mendukung hasil bisnis Anda.

Praktik terbaik

- OPS03-BP01 Memberikan sponsor eksekutif
- OPS03-BP02 Anggota tim diberdayakan untuk mengambil tindakan ketika hasil berisiko
- OPS03-BP03 Eskalasi didorong
- OPS03-BP04 Komunikasi tepat waktu, jelas, dan dapat ditindaklanjuti
- OPS03-BP05 Eksperimen dianjurkan
- OPS03-BP06 Anggota tim didorong untuk mempertahankan dan mengembangkan keahlian mereka
- OPS03-BP07 Tim sumber daya dengan tepat

### OPS03-BP01 Memberikan sponsor eksekutif

Pada tingkat tertinggi, pimpinan senior bertindak sebagai sponsor eksekutif untuk menetapkan ekspektasi dan arah untuk hasil organisasi dengan jelas, termasuk mengevaluasi keberhasilannya. Sponsor mendukung dan mendorong penggunaan praktik-praktik terbaik serta perkembangan organisasi.

Hasil yang diinginkan: Organisasi-organisasi yang berusaha untuk mengadopsi, mentransformasi, dan mengoptimalkan operasi cloud mereka menetapkan garis kepemimpinan dan akuntabilitas yang jelas untuk hasil yang diinginkan. Organisasi memahami setiap kemampuan yang dibutuhkan oleh organisasi untuk mencapai hasil baru dan menetapkan kepemilikan kepada tim fungsional untuk pengembangan. Kepemimpinan secara aktif menetapkan arah ini, memberikan kepemilikan, bertanggung jawab, dan mendefinisikan pekerjaan. Hasilnya, individu yang ada di seluruh organisasi dapat termobilisasi, merasa terinspirasi, dan secara aktif bekerja menuju tujuan yang diinginkan.

Anti-pola umum:

- Terdapat sebuah mandat bagi pemilik beban kerja untuk memigrasikan beban kerja AWS tanpa sponsor yang jelas dan membuat rencana untuk operasi cloud. Hal ini mengakibatkan tim berkolaborasi untuk melakukan peningkatan dan mematangkan kemampuan operasional tanpa ada kesadaran. Kurangnya standar praktik terbaik operasional membuat tim menjadi kewalahan (seperti kerja keras operator, kondisi selalu siaga, dan utang teknis), yang membatasi inovasi.

- Sasaran organisasi baru telah ditetapkan untuk mengadopsi sebuah teknologi baru tanpa memberikan sponsor dan strategi kepemimpinan. Tim menafsirkan tujuan-tujuan secara berbeda, yang menyebabkan kebingungan di antara mereka tentang ke mana upaya harus difokuskan, alasan mengapa tujuan-tujuan itu penting, dan bagaimana dampaknya harus diukur. Akibatnya, organisasi kehilangan momentum dalam mengadopsi teknologi.

Manfaat menerapkan praktik terbaik ini: Ketika sponsor eksekutif secara jelas mengomunikasikan dan berbagi visi, arah, dan tujuan yang hendak dicapainya, anggota tim akan tahu apa yang diharapkan dari mereka. Individu dan tim mulai memfokuskan upaya secara intens ke arah yang sama untuk mencapai tujuan-tujuan yang sudah ditentukan ketika para pemimpin terlibat secara aktif. Hasilnya, organisasi memaksimalkan kemampuan untuk mencapai keberhasilan. Ketika Anda melakukan evaluasi atas kesuksesan, Anda dapat mengidentifikasi hambatan dengan lebih baik sehingga hambatan tersebut dapat diatasi melalui campur tangan sponsor eksekutif.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Pada setiap fase perjalanan cloud (migrasi, adopsi, atau optimalisasi), kesuksesan membutuhkan keterlibatan aktif di tingkat kepemimpinan tertinggi dengan sponsor eksekutif yang ditunjuk. Sponsor eksekutif menyelaraskan pola pikir tim, serangkaian keahlian, dan cara bekerja dengan strategi yang ditentukan.
  - Jelaskan alasannya: Berikan kejelasan dan uraikan alasan-alasan di balik visi dan strategi.
  - Tetapkan ekspektasi: Tentukan dan publikasikan tujuan-tujuan yang ingin diraih organisasi Anda, termasuk cara mengukur keberhasilannya.
  - Lacak pencapaian tujuan: Ukur pencapaian tujuan bertahap secara teratur (bukan hanya penyelesaian tugasnya saja). Bagikan hasilnya sehingga tindakan yang tepat dapat dilakukan jika hasil sedang dalam risiko.
  - Sediakan sumber daya yang diperlukan untuk mencapai tujuan Anda: Satukan orang-orang dan tim untuk saling berkolaborasi dan membangun solusi yang tepat yang mewujudkan hasil yang ingin dicapai. Hal ini mengurangi atau menghilangkan gesekan organisasi.
  - Dukung tim Anda: Tetap berinteraksi dengan tim Anda sehingga Anda memahami bagaimana kondisi mereka dan mengetahui jika ada faktor eksternal yang memengaruhi mereka. Identifikasi rintangan yang memperlambat kemajuan tim Anda. Bertindaklah atas nama tim Anda untuk membantu mengatasi hambatan dan menghilangkan beban yang tidak perlu. Ketika ada faktor

eksternal yang memengaruhi kinerja tim Anda, lakukan evaluasi kembali tujuan dan sesuaikan target sebagaimana mestinya.

- Dorong penerapan praktik terbaik: Kenali praktik terbaik yang memberikan manfaat-manfaat yang dapat diukur, dan kenali para pembuat dan pengguna praktik tersebut. Dukung adopsi lebih lanjut untuk memperbesar manfaat yang dapat dicapai dengannya.
- Dorong evolusi tim Anda: Ciptakan budaya perbaikan terus-menerus, dan secara proaktif belajarlah dari kemajuan yang dibuat, dan belajarlah pula dari kegagalan yang dialami. Dukung pertumbuhan dan perkembangan perorangan maupun organisasi. Gunakan data dan anekdot untuk mengembangkan visi dan strategi.

Contoh pelanggan

AnyCompany Ritel sedang dalam proses transformasi bisnis melalui penemuan kembali pengalaman pelanggan yang cepat, peningkatan produktivitas, dan percepatan pertumbuhan melalui AI generatif.

Langkah-langkah implementasi

1. Bangun kepemimpinan yang berutas tunggal (single-threaded), dan tunjuk sponsor eksekutif utama untuk memimpin dan mendorong transformasi.
2. Tentukan hasil bisnis yang jelas dari transformasi Anda, dan tetapkan kepemilikan serta pertanggunjawabannya. Berdayakan pejabat eksekutif utama dengan wewenang untuk memimpin dan mengambil keputusan-keputusan penting.
3. Pastikan bahwa strategi transformasional Anda sudah sangat jelas dan telah dikomunikasikan secara luas oleh sponsor eksekutif ke setiap tingkat organisasi.
   a. Tetapkan tujuan-tujuan bisnis yang jelas untuk inisiatif IT dan cloud.
   b. Buatlah dokumentasi metrik bisnis utama untuk mendorong transformasi IT dan cloud.
   c. Komunikasikan visi secara konsisten kepada semua tim dan individu perorangan yang bertanggung jawab atas bagian-bagian dari strategi tersebut.
4. Kembangkan matriks perencanaan komunikasi yang menentukan pesan apa yang perlu disampaikan kepada pemimpin, manajer, dan kontributor individu perorangan tertentu. Tentukan orang atau tim yang harus menyampaikan pesan ini.
   a. Jalankan rencana komunikasi secara konsisten dan andal.
   b. Tetapkan dan kelola ekspektasi melalui acara tatap muka yang diadakan secara rutin.
   c. Terima umpan balik mengenai efektivitas komunikasi, dan sesuaikan komunikasinya, dan buatlah rencana komunikasi sesuai dengan itu.

   d. Jadwalkan acara-acara komunikasi untuk memahami tantangan dari tim secara proaktif, dan buatlah sebuah loop umpan balik yang konsisten yang akan memungkinkan penyesuaian arah, jika diperlukan.
5. Libatkan secara aktif setiap inisiatif dari perspektif pimpinan untuk memastikan bahwa semua tim yang terkena dampak memahami hasil yang menjadi tanggung jawab mereka.
6. Pada setiap rapat status, sponsor eksekutif harus mencari penghalang, memeriksa metrik yang ditetapkan, anekdot, atau umpan balik dari tim, dan mengukur kemajuan pencapaian tujuan-tujuan yang ditetapkan.

Tingkat upaya untuk rencana implementasi Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS03-BP04 Komunikasi tepat waktu, jelas, dan dapat ditindaklanjuti
- OP11-BP01 Memiliki proses untuk perbaikan berkelanjutan
- OPS11-BP07 Lakukan tinjauan metrik operasi

Dokumen terkait:

- Mengurai Kekusutan Organisasi Anda: Sangat Selaras
- Transformasi yang Hidup: Perubahan-perubahan dengan pendekatan pragmatis
- Menjadi Perusahaan yang Siap Menghadapi Masa Depan
- 7 Jebakan yang Harus Dihindari Saat Membangun CCOE
- Menavigasi Cloud: Indikator Kinerja Utama untuk Mencapai Keberhasilan

Video terkait:

- AWS re: Invent 2023: Panduan pemimpin untuk AI generatif: Menggunakan sejarah untuk membentuk masa depan (04) SEG2

Contoh terkait:

- Prosci: Peran & Pentingnya Sponsor Utama

OPS03-BP02 Anggota tim diberdayakan untuk mengambil tindakan ketika hasil berisiko

Perilaku kepemilikan yang membudaya yang ditanamkan oleh pimpinan akan menimbulkan perasaan diberdayakan pada diri karyawan untuk bertindak atas nama seluruh perusahaan di luar cakupan peran dan pertanggungjawaban mereka. Karyawan dapat bertindak untuk mengidentifikasi risiko secara proaktif saat risiko-risiko itu muncul dan melakukan tindakan sebagaimana mestinya. Dengan budaya seperti ini, karyawan dapat mengambil keputusan bernilai tinggi dengan kesadaran terhadap situasi yang dihadapinya.

Misalnya, Amazon menggunakan Prinsip Kepemimpinan sebagai pedoman untuk mendorong perilaku yang diinginkan bagi para karyawan untuk bergerak maju saat menghadapi kesulitan, memecahkan masalah, menangani konflik, dan mengambil tindakan.

Hasil yang diinginkan: Para pimpinan telah menyebarkan pengaruh budaya baru yang memungkinkan individu perseorangan dan tim membuat keputusan-keputusan penting, bahkan pada tingkat organisasi yang lebih rendah (selama keputusan-keputusan itu ditentukan dengan izin yang dapat diaudit dan mekanisme keselamatan). Kegagalan tidak perlu dipermalukan, dan tim secara berulang belajar untuk memperbaiki cara-cara mereka dalam mengambil keputusan dan dalam memberikan respons untuk mengatasi situasi-situasi serupa di kesempatan lain. Jika tindakan seseorang mengakibatkan terjadinya perbaikan yang dapat menguntungkan tim lain, mereka membagikan pengetahuan yang didapatkan dari tindakan tersebut secara proaktif. Pimpinan mengukur perbaikan operasional dan memberikan insentif kepada individu perseorangan dan organisasi yang mengadopsi pola-pola tersebut.

Anti-pola umum:

- Tidak ada panduan atau mekanisme yang jelas dalam organisasi terkait hal-hal yang harus dilakukan ketika ada risiko yang diidentifikasi. Misalnya, ketika seorang karyawan melihat adanya serangan phishing, lalu karyawan tersebut tidak melapor ke tim keamanan, dan mengakibatkan sebagian besar organisasi menjadi korban dari serangan tersebut. Hal ini mengakibatkan terjadinya pembobolan data.
- Pelanggan Anda mengeluhkan bahwa layanan tidak tersedia, yang terutama disebabkan oleh deployment yang gagal. SRETim Anda bertanggung jawab atas alat penyebaran, dan rollback otomatis untuk penerapan ada dalam peta jalan jangka panjang mereka. Dalam peluncuran (rollout) aplikasi baru-baru ini, salah satu rekayasawan menemukan solusi untuk mengotomatiskan rollback aplikasi ke versi sebelumnya. Meskipun solusi mereka dapat menjadi pola bagi SRE tim, tim lain tidak mengadopsi, karena tidak ada proses untuk melacak peningkatan tersebut.

Organisasi terus diganggu dengan deployment yang gagal dan berdampak pada para pelanggan dan menyebabkan sentimen negatif lebih lanjut.

- Agar tetap patuh, tim infosec Anda mengawasi proses yang telah lama ada untuk memutar SSH kunci bersama secara teratur atas nama operator yang terhubung ke instans Amazon Linux mereka. EC2 Dibutuhkan beberapa hari bagi tim infosec untuk menyelesaikan proses rotasi kunci ini, dan Anda pun tidak dapat terkoneksi ke instans tersebut. Tidak ada seorang pun di dalam atau di luar infosec yang menyarankan menggunakan opsi lain AWS untuk mencapai hasil yang sama.

Manfaat menerapkan praktik terbaik ini:  Dengan melakukan desentralisasi atas otoritas untuk membuat keputusan dan memberdayakan tim Anda untuk membuat keputusan kunci, Anda dapat mengatasi masalah dengan lebih cepat dan dengan tingkat keberhasilan yang meningkat. Selain itu, tim mulai menyadari adanya rasa kepemilikan, dan kegagalan dapat diterima. Eksperimen menjadi andalan yang membudaya. Manajer dan direktur tidak merasakan bahwa mereka seolah-olah dikontrol secara berlebihan di setiap aspek pekerjaan mereka.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

1. Kembangkan budaya yang menganggap kegagalan sebagai sebuah hal lumrah.
2. Tetapkan kepemilikan dan pertanggungjawaban yang jelas untuk berbagai area fungsional di dalam organisasi.
3. Komunikasikan kepemilikan dan akuntabilitas kepada semua orang sehingga setiap individu tahu siapa yang dapat membantu mereka memfasilitasi keputusan yang terdesentralisasi.
4. Tentukan keputusan satu arah dan dua arah Anda untuk membantu individu perseorangan mengetahui kapan mereka perlu melakukan eskalasi ke tingkat kepemimpinan yang lebih tinggi.
5. Ciptakan kesadaran organisasi bahwa semua karyawan diberdayakan untuk melakukan tindakan di berbagai tingkatan ketika ada masalah yang membahayakan hasil. Bekali para anggota tim Anda dengan dokumentasi tata kelola, tingkat izin, alat, dan peluang untuk mempraktikkan keterampilan yang diperlukan untuk memberikan respons secara efektif.
6. Berikan kepada para anggota tim Anda peluang untuk melatih keterampilan yang diperlukan untuk memberikan respons atas berbagai keputusan. Setelah tingkat keputusan ditentukan, jalankan game day untuk memastikan bahwa semua kontributor perorangan memahami dan dapat mendemonstrasikan proses.
   a. Sediakan lingkungan alternatif yang aman di mana proses dan prosedur dapat diuji dan dilatih.

   b. Akui dan ciptakan kesadaran bahwa para anggota tim memiliki wewenang untuk melakukan tindakan ketika hasil mengandung risiko pada tingkat yang telah ditentukan.
   c. Tentukan otoritas yang dimiliki anggota tim Anda untuk mengambil tindakan dengan memberikan izin dan akses ke beban kerja dan komponen yang mereka dukung.
7. Berikan kemampuan kepada tim untuk berbagi pembelajaran mereka (baik keberhasilan maupun kegagalan yang berkaitan dengan operasi).
8. Berdayakan tim untuk menantang status quo, dan sediakan mekanisme untuk melacak dan mengukur perbaikan, serta mengukur dampaknya terhadap organisasi.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP06 Mengevaluasi pengorbanan sambil mengelola manfaat dan risiko
- OPS02-BP05 Mekanisme ada untuk mengidentifikasi tanggung jawab dan kepemilikan

Dokumen terkait:

- Posting Blog AWS | Perusahaan yang tangkas
- Posting Blog AWS | Mengukur kesuksesan: Paradoks dan rencana
- Posting Blog AWS | Melepaskan: Mengaktifkan otonomi dalam tim
- Sentralisasi atau Desentralisasi?

Video terkait:

- Re:invent 2023 | Bagaimana tidak menyabotase transformasi Anda (01) SEG2
- re:Invent 2021 | Amazon Builders' Library: Keunggulan Operasional di Amazon
- Sentralisasi vs Desentralisasi

Contoh terkait:

- Menggunakan catatan keputusan arsitektur untuk merampingkan proses pengambilan keputusan teknis untuk proyek pengembangan perangkat lunak

### OPS03-BP03 Eskalasi didorong

Anggota tim diimbau oleh para pimpinan untuk menyampaikan masalah dan kekhawatiran mereka kepada pengambil keputusan dan pemangku kepentingan pada level yang lebih tinggi, jika mereka meyakini bahwa hal itu menimbulkan risiko pada hasil dan standar yang diharapkan tidak terpenuhi. Ini adalah bagian dari budaya organisasi dan didorong pada semua level. Eskalasi (penyampaian) harus dilakukan sejak dini dan sering kali agar risiko-risiko semacam itu dapat diidentifikasi, dan dicegah sebelum menyebabkan insiden. Pimpinan tidak menegur individu perseorangan karena menyampaikan masalah ke level pimpinan yang lebih tinggi.

Hasil yang diinginkan: Individu perseorangan yang ada di seluruh organisasi merasa nyaman untuk menyampaikan masalah ke tingkat pimpinan langsung dan lebih tinggi. Pimpinan telah dengan sengaja dan sadar menetapkan ekspektasi bahwa tim mereka harus merasa aman untuk menyampaikan masalah apa pun ke pimpinan yang lebih tinggi. Terdapat mekanisme untuk menyampaikan masalah ke pimpinan yang lebih tinggi di setiap tingkat dalam organisasi. Ketika karyawan melakukan eskalasi ke manajer mereka, mereka bersama-sama memutuskan tingkat dampaknya dan apakah masalah tersebut harus disampaikan ke tingkat pimpinan yang lebih tinggi. Untuk memulai eskalasi, karyawan diharuskan untuk menyertakan rencana kerja yang direkomendasikan untuk mengatasi masalah tersebut. Jika manajemen langsung tidak melakukan tindakan dengan tepat waktu, maka karyawan diimbau untuk menyampaikan masalah tersebut ke tingkat pimpinan tertinggi jika mereka merasa sangat yakin bahwa risiko terhadap organisasi tersebut benar-benar perlu dieskalasi.

Anti-pola umum:

- Para pemimpin eksekutif tidak mengajukan pertanyaan yang cukup cermat selama pelaksanaan rapat status program transformasi cloud Anda untuk menemukan letak terjadinya masalah dan hambatan. Yang disajikan sebagai status hanyalah kabar baik. Mereka CIO telah menjelaskan bahwa dia hanya suka mendengar kabar baik, karena tantangan apa pun yang muncul membuat CEO orang berpikir bahwa program tersebut gagal.
- Anda adalah seorang rekayasawan operasi cloud dan Anda melihat bahwa sistem manajemen pengetahuan yang baru tidak diadopsi secara luas oleh tim aplikasi. Perusahaan menginvestasikan waktu satu tahun dan dana beberapa juta dolar untuk mengimplementasikan sistem manajemen pengetahuan baru tersebut, tetapi orang-orang masih menulis runbook mereka secara lokal dan membagikannya di layanan berbagi cloud organisasi, sehingga pengetahuan terkait beban kerja yang didukung itu menjadi sulit ditemukan. Anda mencoba menyampaikan hal ini kepada pimpinan, karena penggunaan sistem ini secara konsisten dapat meningkatkan efisiensi operasional. Ketika Anda menyampaikannya kepada direktur yang memimpin implementasi sistem manajemen

pengetahuan tersebut, ia akan menegur Anda karena hal ini dianggap dapat menciptakan keraguan pada investasi.

- Tim infosec yang bertanggung jawab untuk mengeraskan sumber daya komputasi telah memutuskan untuk menerapkan proses yang mengharuskan melakukan pemindaian yang diperlukan untuk memastikan bahwa EC2 instance sepenuhnya diamankan sebelum tim komputasi merilis sumber daya untuk digunakan. Ini telah menciptakan penundaan waktu seminggu tambahan untuk sumber daya yang akan digunakan, yang merusak merekaSLA. Tim komputasi tidak berani menyampaikan masalah ini kepada VP melalui cloud karena hal ini merusak citra VP keamanan informasi.

Manfaat menjalankan praktik terbaik ini:

Masalah-masalah yang kompleks atau kritis ditangani sebelum berdampak pada bisnis. Lebih sedikit waktu yang terbuang. Risiko diminimalkan. Tim menjadi lebih proaktif dan fokus pada hasil ketika memecahkan masalah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kemauan dan kemampuan untuk melakukan eskalasi secara bebas di setiap tingkatan di dalam organisasi adalah sebuah landasan organisasi dan budaya yang harus dikembangkan secara sadar melalui pelatihan yang ditekankan, komunikasi kepemimpinan, penetapan ekspektasi, dan deployment mekanisme di seluruh organisasi pada setiap tingkat.

Langkah-langkah implementasi

1. Tentukan kebijakan, standar, dan ekspektasi untuk organisasi Anda.
   a. Pastikan adopsi dan pemahaman yang luas tentang kebijakan, ekspektasi, dan standar.
2. Dorong, latih, dan berdayakan pekerja untuk melakukan eskalasi waktu secara dini dan sering kali ketika standar tidak terpenuhi.
3. Akui pada tingkat organisasi bahwa eskalasi yang dilakukan sejak dini dan sering kali merupakan praktik terbaik. Akui bahwa eskalasi mungkin saja terbukti tidak berdasar, tetapi lebih baik mengambil kesempatan untuk mencegah terjadinya insiden daripada melewatkan kesempatan tersebut dengan tidak melakukan eskalasi.
   a. Bangun sebuah sebuah mekanisme untuk melakukan eskalasi (seperti sistem kabel Andon).
   b. Miliki prosedur terdokumentasi yang menetapkan kapan dan bagaimana eskalasi harus dilakukan.

   c. Tentukan sekelompok personel dengan otoritas berjenjang untuk melakukan atau menyetujui tindakan, beserta informasi kontak dari setiap pemangku kepentingan.

4. Ketika eskalasi terjadi, eskalasi tersebut harus berlanjut sampai anggota tim yakin bahwa risiko telah ditangani melalui tindakan yang didorong dari para pimpinan.
   a. Eskalasi harus mencakup:
      i. Deskripsi situasi, dan sifat risiko
      ii. Tingkat kekritisan situasi
      iii. Siapa atau apa yang terkena dampak
      iv. Seberapa besar dampak tersebut
      v. Urgensi jika dampak terjadi
      vi. Saran perbaikan dan rencana penanganan
   b. Lindungi karyawan yang melakukan eskalasi. Miliki kebijakan yang melindungi para anggota tim dari tindakan pembalasan jika mereka melakukan eskalasi di sekitar pengambil keputusan atau pemangku kepentingan yang tidak responsif. Terapkan mekanisme untuk mengidentifikasi apakah hal ini terjadi dan beri respons yang tepat.
5. Tumbuhkan budaya loop umpan balik perbaikan berkelanjutan dalam segala hal yang dihasilkan oleh organisasi. Loop umpan balik bertindak sebagai eskalasi kecil kepada individu yang bertanggung jawab, dan mereka mengidentifikasi peluang-peluang perbaikan, bahkan ketika eskalasi tidak diperlukan. Budaya perbaikan berkelanjutan mendorong setiap orang untuk menjadi lebih proaktif.
6. Pimpinan harus menekankan ulang secara berkala kebijakan, standar, mekanisme, dan keinginan untuk mewujudkan eskalasi yang terbuka dan loop umpan balik berkelanjutan tanpa tindakan pembalasan.

Tingkat upaya untuk Rencana Implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP05 Mekanisme ada untuk meminta penambahan, perubahan, dan pengecualian

Dokumen terkait:

- Bagaimana Anda menumbuhkan budaya perbaikan berkelanjutan dan belajar dari Andon dan sistem eskalasi?
- Kabel Andon (Revolusi TI)
- AWS DevOps Panduan | Tetapkan jalur eskalasi yang jelas dan dorong ketidaksepakatan yang konstruktif

Video terkait:

- Jeff Bezos tentang cara membuat keputusan (& meningkatkan kecepatan)
- Sistem Produk Toyota: Menghentikan Produksi, sebuah Tombol, dan Papan Listrik Andon
- Kabel Andon di Manufaktur LEAN

Contoh terkait:

- Bekerja dengan rencana eskalasi di Incident Manager

OPS03-BP04 Komunikasi tepat waktu, jelas, dan dapat ditindaklanjuti

Pimpinan bertanggung jawab untuk menciptakan komunikasi yang kuat dan efektif, terutama ketika organisasi mengadopsi strategi, teknologi, atau cara kerja baru. Pemimpin harus menetapkan ekspektasi bagi semua staf untuk bekerja untuk mencapai tujuan-tujuan perusahaan. Rancang mekanisme komunikasi yang menciptakan dan memelihara kesadaran di antara tim yang bertanggung jawab untuk menjalankan rencana yang didanai dan disponsori oleh pimpinan. Manfaatkan keragaman lintas organisasi, dan dengarkan dengan penuh perhatian berbagai perspektif yang unik. Gunakan perspektif ini untuk meningkatkan inovasi, menantang asumsi Anda, dan mengurangi risiko bias konfirmasi. Tumbuhkan inklusi, keragaman, dan kemudahan akses dalam tim Anda untuk mendapatkan perspektif yang bermanfaat.

Hasil yang diinginkan: Organisasi Anda merancang strategi-strategi komunikasi untuk mengatasi dampak yang ditimbulkan oleh perubahan-perubahan terhadap organisasi. Tim tetap mendapatkan informasi terbaru dan termotivasi untuk terus bekerja sama satu sama lain, bukan saling melawan. Individu perseorangan memahami betapa pentingnya peran mereka untuk mencapai tujuan-tujuan yang telah ditetapkan. Email hanyalah mekanisme pasif untuk komunikasi dan digunakan sebagaimana mestinya. Manajemen meluangkan waktu dengan kontributor individual perorangan mereka untuk membantu mereka memahami tanggung jawab, tugas yang harus diselesaikan, dan bagaimana pekerjaan mereka berkontribusi pada keseluruhan misi. Jika perlu, para pemimpin

melibatkan personel secara langsung di lokasi-lokasi yang lebih kecil untuk menyampaikan pesan dan memastikan bahwa pesan-pesan ini tersampaikan secara efektif. Sebagai hasil dari strategi komunikasi yang baik, organisasi menunjukkan kinerja sesuai atau melampaui harapan pimpinan. Pimpinan mendorong dan meminta pendapat yang beragam di dalam dan di seluruh tim.

Anti-pola umum:

- Organisasi Anda memiliki rencana lima tahun untuk memigrasi semua beban kerja ke AWS. Kasus bisnis untuk cloud mencakup modernisasi 25% dari semua beban kerja untuk memanfaatkan teknologi nirserver. Strategi ini CIO mengkomunikasikan strategi ini untuk mengarahkan laporan dan mengharapkan setiap pemimpin untuk menyampaikan presentasi ini kepada manajer, direktur, dan kontributor individu tanpa komunikasi langsung. CIOLangkah mundur dan mengharapkan organisasinya untuk melakukan strategi baru.
- Pimpinan tidak menyediakan atau menggunakan mekanisme untuk umpan balik, dan kesenjangan ekspektasi pun tumbuh, yang menyebabkan proyek terhenti.
- Anda diminta untuk membuat perubahan pada grup keamanan Anda, tetapi Anda tidak diberikan detail apa pun tentang perubahan yang perlu dilakukan, dampak perubahan yang dapat terjadi pada semua beban kerja, dan kapan hal tersebut seharusnya terjadi. Manajer meneruskan email dari VP InfoSec dan menambahkan pesan “Buat ini terjadi.”
- Perubahan dilakukan pada strategi migrasi Anda yang mengurangi jumlah modernisasi yang direncanakan dari 25% menjadi 10%. Perubahan ini memiliki efek hilir pada organisasi operasi. Mereka tidak diberi tahu tentang perubahan strategis ini sehingga mereka tidak siap dengan kapasitas personel terampil yang memadai untuk mendukung beban kerja yang diangkat dan digeser dalam jumlah yang lebih besar ke AWS.

Manfaat menjalankan praktik terbaik ini:

- Organisasi Anda selalu menerima informasi tentang strategi baru atau perubahan strategi, dan mereka bertindak sebagaimana mestinya dengan motivasi yang kuat untuk saling membantu guna mencapai keseluruhan tujuan dan metrik yang telah ditetapkan oleh pimpinan.
- Mekanisme tersedia dan digunakan untuk memberikan pengingat secara tepat waktu kepada anggota tim tentang risiko-risiko yang diketahui dan peristiwa-peristiwa yang direncanakan.
- Cara kerja baru (termasuk perubahan pada personel atau organisasi, proses, atau teknologi), beserta keterampilan yang dibutuhkan, diadopsi dengan lebih efektif oleh organisasi, dan organisasi Anda menyadari manfaat-manfaat bisnis dengan lebih cepat.

- Anggota tim memiliki konteks yang diperlukan tentang komunikasi-komunikasi yang diterima, dan mereka dapat bekerja dengan lebih efektif.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk mengimplementasikan praktik terbaik ini, Anda harus bekerja sama dengan para pemangku kepentingan di seluruh organisasi untuk menyepakati standar komunikasi. Publikasikan standar tersebut ke organisasi Anda. Untuk transisi IT yang signifikan, tim perencanaan yang telah sudah dibentuk dan mapan mempunyai kecenderungan lebih untuk berhasil mengelola dampak perubahan terhadap personelnya daripada organisasi yang mengabaikan praktik ini. Manajemen perubahan bisa lebih menantang untuk organisasi yang lebih besar, karena dukungan yang kuat untuk strategi baru sangat diperlukan dari semua kontributor individual perorangan. Dengan tidak adanya tim perencanaan transisi seperti ini, pimpinan memegang 100% tanggung jawab untuk melakukan komunikasi yang efektif. Saat membentuk tim perencanaan transisi, tugaskan para anggota tim untuk bekerja dengan semua pimpinan organisasi untuk menentukan dan mengelola komunikasi yang efektif di setiap tingkat.

Contoh pelanggan

AnyCompany Retail mendaftar untuk AWS Enterprise Support dan bergantung pada penyedia pihak ketiga lainnya untuk operasi cloud-nya. Perusahaan menggunakan obrolan dan chatops sebagai media komunikasi utama mereka untuk aktivitas-aktivitas operasional. Peringatan dan informasi lainnya memenuhi saluran tertentu. Ketika seseorang harus bertindak, mereka menyatakan hasil yang diinginkan dengan jelas, dan dalam banyak kasus, mereka menerima runbook atau playbook yang bisa digunakan. Mereka menjadwalkan perubahan besar pada sistem produksi yang memiliki sebuah kalender perubahan.

Langkah-langkah implementasi

1. Bangunlah sebuah tim inti di dalam organisasi yang memiliki pertanggungjawaban untuk membangun dan memulai rencana komunikasi untuk perubahan yang terjadi di berbagai tingkatan di dalam organisasi.
2. Tetapkan kepemilikan utas tunggal untuk mencapai pengawasan. Bekali masing-masing tim dengan kemampuan untuk melakukan inovasi secara mandiri, dan seimbangkan penggunaan mekanisme yang konsisten, yang memungkinkan inspeksi dan visi direksional pada level yang tepat.

3. Bekerjalah dengan para pemangku kepentingan di seluruh organisasi Anda untuk menyepakati standar komunikasi, praktik, dan rencana.
4. Pastikan bahwa tim komunikasi inti melakukan kolaborasi dengan para pimpinan organisasi dan program untuk menyusun pesan kepada staf yang sesuai atas nama para pemimpin.
5. Membangun mekanisme komunikasi strategis untuk mengelola perubahan melalui pengumuman, kalender bersama, pertemuan semua tangan, dan tatap muka atau one-on-one metode sehingga anggota tim memiliki harapan yang tepat pada tindakan yang harus mereka ambil.
6. Sediakan konteks, detail, dan waktu yang diperlukan (apabila memungkinkan) untuk menentukan apakah tindakan perlu dilakukan. Ketika tindakan diperlukan, beritahukan tindakan apa yang diperlukan beserta dampaknya.
7. Implementasikan alat-alat yang memudahkan komunikasi taktis, seperti obrolan internal, email, dan manajemen pengetahuan.
8. Implementasikan mekanisme untuk mengukur dan memastikan bahwa semua komunikasi memberikan hasil yang diinginkan.
9. Buatlah sebuah loop umpan balik yang mengukur efektivitas semua komunikasi, terutama ketika komunikasi berkaitan dengan resistensi terhadap perubahan di seluruh organisasi.
10. Untuk semua Akun AWS, buat kontak alternatif untuk penagihan, keamanan, dan operasi. Idealnya, setiap kontak harus berupa distribusi email, bukan kontak individu perseorangan tertentu.
11. Buat rencana komunikasi eskalasi dan eskalasi balik untuk terlibat dengan tim internal dan eksternal Anda, termasuk AWS dukungan dan penyedia pihak ketiga lainnya.
12. Mulai dan jalankan strategi komunikasi secara konsisten selama berlangsungnya setiap program transformasi.
13. Prioritaskan tindakan yang dapat diulang, jika memungkinkan, untuk mengaktifkan otomatisasi yang aman dalam skala besar.
14. Ketika komunikasi perlu dilakukan dengan melibatkan tindakan-tindakan otomatis, tujuan komunikasi seharusnya adalah untuk memberikan informasi kepada tim, untuk melakukan audit, atau sebagai bagian dari proses manajemen perubahan.
15. Analisis komunikasi dari sistem peringatan Anda untuk mendeteksi hasil positif palsu atau peringatan yang terus-menerus dibuat. Hapus atau ubah peringatan-peringatan tersebut sehingga peringatan tersebut dimulai ketika diperlukan intervensi manusia. Jika ada sebuah peringatan muncul, berikan runbook atau playbook.
    a. Anda dapat menggunakan AWS Systems Manager Documents untuk membuat playbook dan runbook untuk peringatan.

16. Mekanisme diterapkan untuk memberikan pemberitahuan risiko atau acara-acara yang direncanakan dengan cara yang jelas dan dapat ditindaklanjuti, melalui peringatan yang memadai untuk memberi respons yang sesuai. Gunakan daftar email atau saluran obrolan untuk mengirimkan pemberitahuan sebelum acara yang sudah direncanakan.
    a. AWS Chatbot dapat Anda gunakan untuk mengirim peringatan dan menanggapi peristiwa dalam platform pesan organisasi Anda.
17. Berikan sumber informasi yang dapat diakses di mana acara yang sudah direncanakan dapat ditemukan. Beri pemberitahuan tentang acara-acara yang direncanakan dari sistem yang sama.
    a. AWS Systems Manager Change Calendar dapat digunakan untuk membuat jendela perubahan ketika perubahan dapat terjadi. Hal ini memberikan para anggota tim pemberitahuan mengenai kapan mereka dapat membuat perubahan dengan aman.
18. Pantau pemberitahuan kerentanan dan informasi patch untuk memahami kerentanan yang memiliki risiko tinggi dan risiko potensial yang berkaitan dengan komponen beban kerja Anda. Berikan pemberitahuan kepada para anggota tim agar mereka dapat bertindak.
    a. Anda dapat berlangganan Buletin Keamanan AWS untuk menerima pemberitahuan kerentanan pada AWS.
19. Cari pendapat dan perspektif yang beragam: Dorong kontribusi dari semua orang. Berikan kesempatan komunikasi kepada kelompok yang kurang terwakili. Lakukan rotasi peran dan tanggung jawab dalam rapat.
    a. Perluas peran dan tanggung jawab: Sediakan kesempatan bagi anggota tim untuk mengambil peran yang mungkin jarang bisa mereka ambil. Mereka bisa mendapatkan pengalaman dan perspektif dari peran tersebut serta dari interaksi dengan para anggota tim baru yang mungkin tidak akan berinteraksi dengan mereka di luar peran tersebut. Mereka juga dapat membawa pengalaman dan perspektif mereka ke peran baru tersebut serta untuk para anggota tim yang berinteraksi dengan mereka. Begitu perspektif meningkat, lakukan identifikasi terhadap kesempatan bisnis yang muncul atau peluang perbaikan baru. Lakukan rotasi tugas-tugas umum di antara para anggota di dalam tim agar mereka dapat memahami tuntutan dan dampak yang ditimbulkan dari pelaksanaan tugas-tugas yang biasanya dijalankan oleh anggota yang lain.
    b. Sediakan lingkungan yang aman dan ramah: Miliki kebijakan dan kontrol yang melindungi mental dan keselamatan fisik anggota tim dalam organisasi Anda. Para anggota tim harus bisa berinteraksi tanpa rasa takut akan pembalasan. Ketika para anggota tim merasa aman dan diterima, mereka mungkin menjadi lebih terlibat dan produktif. Makin beragam organisasi Anda, maka akan makin baik pemahaman Anda tentang orang-orang yang Anda dukung termasuk para pelanggan Anda. Ketika anggota tim Anda merasa nyaman, merasa bebas

untuk berbicara, dan meyakini bahwa suara mereka akan didengar, mereka lebih berpeluang untuk membagikan wawasan berharga (misalnya, peluang pemasaran, kebutuhan aksesibilitas, segmen pasar yang belum terlayani, dan risiko-risiko yang tidak diketahui di lingkungan Anda).

c. Dukung anggota tim untuk berpartisipasi penuh: Sediakan sumber daya yang diperlukan bagi karyawan Anda untuk berpartisipasi penuh pada semua aktivitas yang berkaitan dengan pekerjaan. Para anggota tim yang sehari-hari berhadapan dengan tantangan akan mengembangkan keterampilan untuk pekerjaan-pekerjaan di sekitar mereka. Keterampilan yang dikembangkan secara khusus ini bisa memberi keuntungan yang signifikan bagi organisasi Anda. Dukung para anggota tim dengan akomodasi yang diperlukan untuk meningkatkan keuntungan yang bisa Anda terima dari kontribusi mereka.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS03-BP01 Memberikan sponsor eksekutif
- OPS07-BP03 Gunakan runbook untuk melakukan prosedur
- OPS07-BP04 Gunakan pedoman untuk menyelidiki masalah

Dokumen terkait:

- Posting blog AWS | Akuntabilitas dan pemberdayaan adalah kunci bagi organisasi tangkas yang berkinerja tinggi
- AWS Executive Insights | Belajar untuk meningkatkan (menskalakan) inovasi, bukan kompleksitas | Single-threaded Leaders
- Buletin Keamanan AWS
- Terbuka CVE
- AWS Support Aplikasi di Slack untuk Mengelola Kasus Dukungan
- Kelola AWS sumber daya di saluran Slack Anda dengan AWS Chatbot

Contoh terkait:

- Lab Well-Architected - Manajemen Inventaris dan Patch (Level 100)

Layanan terkait:

- AWS Chatbot
- AWS Systems Manager Ubah Kalender
- AWS Dokumen Systems Manager

OPS03-BP05 Eksperimen dianjurkan

Eksperimen adalah katalis untuk mengubah ide baru menjadi produk dan fitur. Eksperimen mempercepat proses pembelajaran dan membuat anggota tim terus tertarik dan terlibat. Anggota tim didorong untuk sering bereksperimen guna mendorong inovasi. Meskipun hasil yang tidak diinginkan terjadi, ada nilai dalam memiliki pengetahuan tentang apa yang sebaiknya tidak dilakukan. Anggota tim tidak dihukum untuk eksperimen yang berhasil dengan hasil yang tidak diinginkan.

Hasil yang diinginkan:

- Organisasi Anda mendorong eksperimen untuk mendukung inovasi.
- Eksperimen digunakan sebagai peluang untuk belajar.

Anti-pola umum:

- Anda ingin menjalankan pengujian A/B tetapi tidak ada mekanisme untuk menjalankan eksperimen tersebut. Anda melakukan deployment perubahan UI tanpa memiliki kemampuan untuk mengujinya. Tindakan tersebut mengakibatkan pengalaman pelanggan yang negatif.
- Perusahaan Anda hanya memiliki lingkungan produksi dan lingkungan pentahapan. Tidak ada lingkungan sandbox untuk melakukan eksperimen dengan fitur atau produk baru sehingga Anda harus melakukan eksperimen di dalam lingkungan produksi.

Manfaat menjalankan praktik terbaik ini:

- Eksperimen mendorong inovasi.
- Anda dapat bereaksi lebih cepat terhadap umpan balik yang diberikan oleh pengguna melalui eksperimen.
- Organisasi Anda mengembangkan budaya belajar.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Eksperimen harus dijalankan dengan cara yang aman. Manfaatkan beberapa lingkungan untuk melakukan eksperimen tanpa membahayakan sumber daya produksi. Gunakan pengujian A/B dan bendera fitur untuk menguji eksperimen. Berikan kepada para anggota tim kemampuan untuk melakukan eksperimen di dalam sebuah lingkungan sandbox.

Contoh pelanggan

AnyCompany Ritel mendorong eksperimen. Anggota tim dapat menggunakan 20% dari hari-hari kerja mereka untuk melakukan eksperimen atau mempelajari teknologi baru. Mereka memiliki sebuah lingkungan sandbox di mana mereka dapat berinovasi. Pengujian A/B digunakan untuk fitur-fitur baru guna memvalidasinya dengan umpan balik nyata dari pengguna.

Langkah-langkah implementasi

1. Bekerjasamalah dengan para pimpinan di seluruh organisasi Anda untuk mendukung eksperimen. Para anggota tim harus didorong untuk melakukan eksperimen dengan cara yang aman.
2. Berikan kepada para anggota tim Anda sebuah lingkungan di mana mereka dapat melakukan eksperimen dengan aman. Mereka harus memiliki akses ke sebuah lingkungan yang mirip dengan lingkungan produksi.
   a. Anda dapat menggunakan terpisah Akun AWS untuk membuat lingkungan kotak pasir untuk eksperimen. AWS Control Towerdapat digunakan untuk menyediakan akun ini.
3. Gunakan bendera fitur dan pengujian A/B untuk melakukan eksperimen dengan aman dan mengumpulkan umpan balik pengguna.
   a. AWS AppConfig Feature Flags menyediakan kemampuan untuk membuat flag fitur.
   b. Amazon CloudWatch Terbukti dapat digunakan untuk menjalankan pengujian A/B selama penerapan terbatas.
   c. Anda dapat menggunakan versi AWS Lambda untuk menerapkan versi baru dari fungsi untuk uji beta.

Tingkat upaya untuk rencana implementasi: Tinggi. Memberikan kepada para anggota tim sebuah lingkungan untuk melakukan eksperimen dan cara yang aman untuk melakukan eksperimen dapat mengharuskan Anda menanamkan investasi besar. Anda juga mungkin harus melakukan modifikasi terhadap kode aplikasi untuk menggunakan bendera fitur atau mendukung pengujian A/B.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP02 Lakukan analisis pasca-insiden - Belajar dari insiden adalah pendorong penting untuk inovasi bersama dengan eksperimen.
- OPS11-BP03 Menerapkan loop umpan balik - Loop umpan balik adalah bagian penting dari eksperimen.

Dokumen terkait:

- Pandangan Bagian Dalam Budaya Amazon: Eksperimen, Kegagalan, dan Obsesi Pelanggan
- Praktik terbaik untuk membuat dan mengelola akun kotak pasir di AWS
- Buat Budaya Eksperimen yang Diaktifkan oleh Cloud
- Mengaktifkan eksperimen dan inovasi di cloud di érica Seguros SulAm
- Eksperimen Lebih Banyak, Lebih Sedikit Gagal
- Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun - Sandbox OU
- Menggunakan Bendera AWS AppConfig Fitur

Video terkait:

- AWS Di Air ft. Amazon CloudWatch Terbukti | Acara AWS
- AWS Di Air San Fran Summit 2022 ft. AWS AppConfig Integrasi Bendera Fitur dengan Jira
- AWS re:invent 2022 - Penerapan bukan rilis: Kontrol peluncuran Anda dengan flag fitur (05-R) BOA3
- Secara pemrograman Buat dengan Akun AWSAWS Control Tower
- Menyiapkan AWS Lingkungan Multi-Akun yang Menggunakan Praktik Terbaik AWS Organizations

Contoh terkait:

- AWS Kotak Pasir Inovasi
- End-to-endPersonalisasi 101 untuk E-Commerce

Layanan terkait:

- Amazon CloudWatch Terbukti
- AWS AppConfig
- AWS Control Tower

OPS03-BP06 Anggota tim didorong untuk mempertahankan dan mengembangkan keahlian mereka

Tim harus mengembangkan tingkat keterampilan mereka untuk mengadopsi perkembangan teknologi, serta untuk mengimbangi perubahan permintaan dan tanggung jawab dalam mendukung beban kerja Anda. Perkembangan keterampilan menggunakan teknologi dapat menjadi sumber kepuasan tim dan mendorong inovasi. Dukung anggota tim Anda untuk mendapatkan dan mempertahankan sertifikasi industri yang memvalidasi dan mengakui perkembangan keterampilan mereka. Terapkan pelatihan silang untuk mendorong transfer pengetahuan dan meminimalkan dampak signifikan yang terjadi karena kehilangan anggota tim berpengalaman yang memiliki keterampilan dan pengetahuan terkait lembaga. Berikan waktu khusus yang terstruktur untuk pembelajaran.

AWS menyediakan sumber daya, termasuk Pusat Sumber Daya AWS Memulai, AWS Blog, Pembicaraan Teknologi AWS Online, AWS Acara dan Webinar, dan Lab AWS Well-Architected, yang memberikan panduan, contoh, dan panduan terperinci untuk mendidik tim Anda.

Sumber daya seperti AWS Support, (AWS re:Post, Pusat AWS Support), dan Dokumentasi AWS akan membantu menghilangkan hambatan teknis dan memperbaiki operasi. Hubungi AWS Support melalui AWS Support Pusat untuk bantuan dengan pertanyaan Anda.

AWS juga membagikan praktik dan pola terbaik yang telah kami pelajari melalui pengoperasian AWS di Amazon Builders' Library dan berbagai macam materi pendidikan berguna lainnya melalui AWS Blog dan The Official Podcast. AWS

AWS Training Sertifikasi mencakup pelatihan gratis melalui kursus digital mandiri, bersama dengan rencana pembelajaran berdasarkan peran atau domain. Anda juga dapat mendaftar untuk pelatihan yang dipimpin instruktur untuk lebih mendukung pengembangan keterampilan tim AWS Anda.

Hasil yang diinginkan: Organisasi Anda terus-menerus melakukan evaluasi terhadap kesenjangan keterampilan yang terjadi dan mengatasinya dengan membuat anggaran dan menanamkan investasi yang terstruktur. Tim mendorong dan mendukung para anggota mereka dengan aktivitas-aktivitas peningkatan keterampilan, misalnya dengan memperoleh sertifikasi industri terkemuka. Tim memanfaatkan program pengetahuan cross-sharing khusus seperti lunch-and-learns, immersion days, hackathon, dan gamedays. Organisasi Anda menjaga sistem pengetahuannya up-to-date dan relevan dengan anggota tim cross-train, termasuk pelatihan orientasi perekrutan baru.

Anti-pola umum:

- Dengan tidak adanya program pelatihan dan anggaran yang terstruktur, tim akan mengalami ketidakpastian saat mereka mencoba mengimbangi perkembangan teknologi, dan hal ini akan mengakibatkan meningkatnya gesekan.
- Sebagai bagian dari migrasi ke AWS, organisasi Anda menunjukkan kesenjangan keterampilan dan berbagai kefasihan cloud di antara tim. Tanpa upaya untuk meningkatkan keterampilan, tim akan mendapati diri mereka terbebani dengan manajemen warisan dan tidak efisien dari lingkungan cloud yang akan menyebabkan meningkatnya kerja keras yang harus dilakukan operator. Kelelahan fisik dan mental ini akan meningkatkan tingkat ketidakpuasan karyawan.

Manfaat menerapkan praktik terbaik ini: Ketika organisasi Anda secara sadar berinvestasi dalam meningkatkan keterampilan timnya, hal itu juga akan membantu mempercepat dan meningkatkan skala adopsi dan pengoptimalan cloud. Program pembelajaran yang tertarget dapat mendorong inovasi dan membangun kemampuan operasional bagi tim agar mereka siap menangani peristiwa-peristiwa yang terjadi. Tim secara sadar berinvestasi untuk mengimplementasikan dan mengembangkan praktik terbaik. Tim memiliki semangat yang tinggi, dan anggota tim menghargai kontribusi mereka terhadap bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Untuk mengadopsi teknologi baru, mendorong inovasi, dan mengimbangi laju perubahan-perubahan yang terjadi pada permintaan dan tanggung jawab untuk mendukung beban kerja Anda, teruslah berinvestasi dalam pertumbuhan profesional tim Anda.

Langkah-langkah implementasi

1. Gunakan program advokasi cloud terstruktur: AWS Skills Guild menyediakan pelatihan konsultatif untuk meningkatkan kepercayaan keterampilan cloud dan memicu budaya pembelajaran yang berkelanjutan.
2. Sediakan sumber daya untuk kepentingan edukasi: Sediakan waktu khusus yang terstruktur, akses ke materi pelatihan, sumber daya lab, dan dukung partisipasi untuk mengikuti konferensi dan organisasi profesional yang memberikan kesempatan untuk belajar dari pendidik dan rekan. Berikan akses kepada anggota tim junior Anda agar dia bisa belajar dari anggota tim senior, atau biarkan anggota tim junior mengamati pekerjaan anggota tim senior serta melihat metode dan

keterampilan mereka. Dorong pembelajaran tentang konten yang tidak terkait langsung dengan pekerjaan agar tim junior tersebut memiliki pandangan yang lebih luas.

3. Dorong penggunaan sumber daya teknis ahli: Manfaatkan sumber daya seperti AWS Re:Post untuk mendapatkan akses ke pengetahuan-pengetahuan pilihan dan komunitas yang dinamis.
4. Membangun dan memelihara repositori up-to-date pengetahuan: Gunakan platform berbagi pengetahuan seperti wiki dan runbook. Buat sumber pengetahuan pakar Anda sendiri yang dapat digunakan kembali dengan AWS re:Post Private untuk merampingkan kolaborasi dan meningkatkan produktivitas serta mempercepat proses onboarding karyawan.
5. Edukasi tim dan interaksi antar-tim: Buat rencana untuk kebutuhan anggota tim terkait pembelajaran berkelanjutan. Berikan kesempatan kepada para anggota tim untuk bergabung dengan tim-tim yang lain (baik sementara atau pun seterusnya) agar mereka bisa berbagi keterampilan dan praktik terbaik yang bermanfaat bagi organisasi Anda.
6. Dukung untuk mendapatkan dan mempertahankan sertifikasi industri: Dukung para anggota tim Anda dalam melakukan proses akuisisi dan pemeliharaan atas sertifikasi industri yang memberikan validasi atas kemampuan yang telah mereka pelajari, serta berikan pengakuan atas pencapaian-pencapaian yang mereka raih.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- OPS03-BP01 Memberikan sponsor eksekutif
- OPS11-BP04 Melakukan manajemen pengetahuan

Dokumen terkait:

- Laporan Resmi AWS | Kerangka Kerja Adopsi Cloud: Perspektif Orang-orang
- Berinvestasi dalam pembelajaran berkelanjutan untuk mengembangkan masa depan organisasi Anda
- AWS Skills Guild
- AWS Training dan Sertifikasi
- AWS Support
- AWS Re: posting

- Pusat Sumber Daya untuk Memulai AWS
- Blog AWS
- Kepatuhan AWS Cloud
- Dokumentasi AWS
- AWS Podcast Resmi.
- Bincang Teknologi Online AWS
- AWS Acara dan Webinar
- Lab AWS Well-Architected
- Amazon Builders' Library

Video terkait:

- AWS re:Invent 2023 | Menyempurnakan keterampilan dengan kecepatan cloud: Mengubah karyawan menjadi wirausahawan
- WS re:Invent 2023 | Membangun budaya keingintahuan melalui gamifikasi

OPS03-BP07 Tim sumber daya dengan tepat

Sediakan anggota tim mahir dalam jumlah yang tepat, serta sediakan alat dan sumber daya untuk mendukung kebutuhan beban kerja Anda. Anggota tim yang terlalu terbebani dapat meningkatkan risiko terjadinya kesalahan manusia. Berinvestasi dalam alat dan sumber daya, seperti otomatisasi, dapat meningkatkan (menskalakan) efektivitas tim Anda dan akan membantu mereka dalam mendukung lebih banyak beban kerja tanpa memerlukan kapasitas tambahan.

Hasil yang diinginkan:

- Anda telah mengatur tim Anda dengan tepat untuk mendapatkan keahlian yang dibutuhkan bagi mereka untuk mengoperasikan beban kerja sesuai dengan rencana migrasi Anda. AWS Karena tim Anda telah meningkatkan dirinya sendiri selama proyek migrasi Anda, mereka telah memperoleh kemahiran dalam AWS teknologi inti yang direncanakan bisnis untuk digunakan saat memigrasi atau memodernisasi aplikasi mereka.
- Anda telah menyelaraskan rencana jumlah personel Anda dengan hati-hati untuk memanfaatkan sumber daya secara efisien dengan memanfaatkan otomatisasi dan alur kerja. Tim yang lebih kecil sekarang dapat mengelola lebih banyak infrastruktur atas nama tim pengembangan aplikasi.

- Dengan perubahan prioritas operasional, kendala-kendala yang dihadapi personel sumber daya diidentifikasi secara proaktif untuk melindungi keberhasilan inisiatif bisnis.
- Metrik-metrik operasional yang melaporkan kerja keras operasional (seperti kelelahan akibat kondisi siaga atau pemanggilan yang berlebihan) ditinjau untuk memastikan bahwa personel tidak kewalahan.

Anti-pola umum:

- Staf Anda belum meningkatkan AWS keterampilan saat Anda mendekati rencana migrasi cloud multi-tahun Anda, yang berisiko mendukung beban kerja dan menurunkan moral karyawan.
- Seluruh organisasi IT Anda sedang beralih ke cara-cara kerja tangkas. Bisnis memprioritaskan portofolio produk dan menetapkan metrik-metrik untuk fitur apa saja yang perlu dikembangkan terlebih dahulu. Proses tangkas Anda tidak mengharuskan tim untuk menetapkan story point ke rencana kerja mereka. Akibatnya, mustahil bagi Anda untuk mengetahui tingkat kapasitas yang dibutuhkan untuk jumlah pekerjaan berikutnya, atau apakah Anda memiliki keterampilan yang tepat yang ditugaskan untuk pekerjaan tersebut.
- Anda meminta AWS mitra memigrasi beban kerja Anda, dan Anda tidak memiliki rencana transisi dukungan untuk tim Anda setelah mitra menyelesaikan proyek migrasi. Tim Anda mengalami kesulitan untuk mendukung beban kerja secara efisien dan efektif.

Manfaat menerapkan praktik terbaik ini: Anda memiliki anggota tim yang terampil yang tersedia di dalam organisasi Anda untuk mendukung beban kerja. Alokasi sumber daya dapat beradaptasi dengan perubahan prioritas tanpa memengaruhi kinerja. Hasilnya adalah tim yang mahir dalam mendukung beban kerja sambil memaksimalkan waktu untuk berkonsentrasi pada inovasi bagi pelanggan, yang pada gilirannya meningkatkan kepuasan karyawan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Perencanaan sumber daya untuk migrasi cloud Anda harus dilakukan pada level organisasi yang selaras dengan rencana migrasi Anda, dan model operasi yang diinginkan yang sedang diimplementasikan untuk mendukung lingkungan cloud baru Anda. Ini harus mencakup pemahaman tentang teknologi-teknologi cloud mana yang di-deploy untuk tim pengembangan aplikasi dan bisnis. Pimpinan infrastruktur dan operasi harus merencanakan analisis kesenjangan keterampilan, pelatihan, dan penetapan peran untuk para rekayasawan yang memimpin adopsi cloud.

Langkah-langkah implementasi

1. Tentukan kriteria keberhasilan untuk keberhasilan tim dengan metrik-metrik operasional yang relevan seperti produktivitas personel (misalnya, biaya untuk mendukung beban kerja tertentu atau jam operator yang dihabiskan selama insiden).
2. Tetapkan perencanaan kapasitas sumber daya dan mekanisme inspeksi untuk memastikan bahwa keseimbangan yang tepat dari kapasitas yang memenuhi syarat benar-benar tersedia saat diperlukan dan dapat disesuaikan dari waktu ke waktu.
3. Ciptakan mekanisme (misalnya, mengirimkan survei bulanan kepada tim) untuk memahami tantangan-tantangan terkait pekerjaan yang memengaruhi tim (seperti meningkatnya tanggung jawab, perubahan teknologi, kehilangan personel, atau peningkatan pelanggan yang didukung).
4. Gunakan mekanisme-mekanisme tersebut untuk berinteraksi dengan tim dan menemukan tren yang mungkin menjadi faktor tantangan produktivitas karyawan. Ketika ada faktor eksternal yang memengaruhi kinerja tim Anda, lakukan evaluasi kembali tujuan dan sesuaikan target sebagaimana mestinya. Identifikasi rintangan yang menghambat kemajuan tim Anda.
5. Lakukan peninjauan secara rutin mengenai apakah sumber daya yang saat ini disediakan masih memadai, atau apakah diperlukan sumber daya tambahan, dan lakukan penyesuaian yang tepat untuk mendukung tim.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS03-BP06 Anggota tim didorong untuk mempertahankan dan mengembangkan keahlian mereka
- OPS09-BP03 Meninjau metrik operasi dan memprioritaskan peningkatan
- OPS10-BP01 Gunakan proses untuk manajemen peristiwa, insiden, dan masalah
- OPS10-BP07 Otomatiskan tanggapan terhadap acara

Dokumen terkait:

- AWS Cloud Kerangka Adopsi: Perspektif Orang
- Menjadi Perusahaan yang Siap Menghadapi Masa Depan
- Prioritaskan Keterampilan Karyawan Anda untuk Mendorong Pertumbuhan Bisnis

- Organisasi berkinerja tinggi - Tim Dua-Pizza Amazon
- Bagaimana Perusahaan dengan Pemanfaatan Cloud yang Matang Meraih Kesuksesannya

## Persiapkan

Pertanyaan

- OPS4. Bagaimana Anda mengimplementasikan observabilitas dalam beban kerja Anda?
- OPS5. Bagaimana cara mengurangi kecacatan, mempermudah perbaikan, dan meningkatkan aliran ke dalam produksi?
- OPS6. Bagaimana cara memitigasi risiko deployment?
- OPS7. Bagaimana cara mengetahui bahwa Anda siap untuk mendukung beban kerja?

### OPS4. Bagaimana Anda mengimplementasikan observabilitas dalam beban kerja Anda?

Terapkan observabilitas dalam beban kerja Anda sehingga Anda dapat memahami statusnya dan membuat keputusan berbasis data berdasarkan persyaratan bisnis.

Praktik terbaik

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS04-BP03 Menerapkan telemetri pengalaman pengguna
- OPS04-BP04 Menerapkan telemetri ketergantungan
- OPS04-BP05 Melaksanakan penelusuran terdistribusi

#### OPS04-BP01 Identifikasi indikator kinerja utama

Untuk mengimplementasikan observabilitas dalam beban kerja, Anda memulainya dengan memahami statusnya dan mengambil keputusan berbasis data berdasarkan persyaratan bisnis. Salah satu cara paling efektif untuk memastikan keselarasan antara kegiatan pemantauan dan tujuan bisnis adalah dengan mendefinisikan dan memantau indikator kinerja utama ()KPIs.

Hasil yang diinginkan: Praktik-praktik observabilitas yang efisien yang sangat selaras dengan tujuan bisnis, sehingga memastikan upaya pemantauan selalu memenuhi hasil bisnis yang nyata.

Anti-pola umum:

- Tidak terdefinisiKPIs: Bekerja tanpa jelas KPIs dapat menyebabkan pemantauan terlalu banyak atau terlalu sedikit, kehilangan sinyal vital.
- StatisKPIs: Tidak meninjau kembali atau menyempurnakan KPIs karena beban kerja atau tujuan bisnis berkembang.
- Ketidakselarasan: Berfokus pada metrik-metrik teknis yang tidak berkorelasi langsung dengan hasil bisnis atau yang lebih sulit untuk berkorelasi dengan masalah-masalah dunia nyata.

Manfaat menjalankan praktik terbaik ini:

- Kemudahan identifikasi masalah: Bisnis KPIs sering memunculkan masalah lebih jelas daripada metrik teknis. Penurunan dalam bisnis KPI dapat menunjukkan masalah dengan lebih efektif daripada memilah-milah berbagai metrik teknis.
- Keselarasan bisnis: Memastikan bahwa kegiatan pemantauan secara langsung adalah aktivitas yang mendukung tujuan bisnis.
- Efisiensi: Prioritaskan untuk melakukan pemantauan sumber daya dan memberikan perhatian pada metrik-metrik yang penting.
- Proaktif: Kenali dan atasi masalah sebelum masalah itu memunculkan dampak bisnis yang lebih luas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk secara efektif mendefinisikan beban kerjaKPIs:

1. Mulai dengan hasil bisnis: Sebelum menyelami metrik, pahami dahulu hasil bisnis yang diinginkan. Apakah peningkatan penjualan, keterlibatan pengguna yang lebih tinggi, atau waktu respons yang lebih cepat?
2. Korelasikan metrik teknis dengan tujuan bisnis: Tidak semua metrik teknis memiliki dampak langsung pada hasil bisnis Anda. Identifikasi mereka yang melakukannya, tetapi seringkali lebih mudah untuk mengidentifikasi masalah menggunakan bisnis. KPI
3. Gunakan Amazon CloudWatch: Gunakan CloudWatch untuk menentukan dan memantau metrik yang mewakili Anda. KPIs

4. Tinjau dan perbarui secara teraturKPIs: Saat beban kerja dan bisnis Anda berkembang, jaga agar tetap relevan. KPIs
5. Libatkan pemangku kepentingan: Libatkan tim teknis dan bisnis dalam mendefinisikan dan meninjau. KPIs

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- the section called “OPS04-BP02 Melaksanakan telemetri aplikasi”
- the section called “OPS04-BP03 Menerapkan telemetri pengalaman pengguna”
- the section called “OPS04-BP04 Menerapkan telemetri ketergantungan”
- the section called “OPS04-BP05 Melaksanakan penelusuran terdistribusi”

Dokumen terkait:

- AWS Praktik Terbaik Observabilitas
- CloudWatch Panduan Pengguna
- AWS Kursus Pembuat Keterampilan Observabilitas

Video terkait:

- Mengembangkan strategi observabilitas

Contoh terkait:

- Lokakarya Satu Observabilitas

OPS04-BP02 Melaksanakan telemetri aplikasi

Telemetri aplikasi berfungsi sebagai fondasi observabilitas beban kerja Anda. Sangat penting bagi Anda untuk menghadirkan telemetri yang menawarkan wawasan yang dapat ditindaklanjuti tentang keadaan aplikasi Anda serta pencapaian hasil teknis dan bisnis. Dari pemecahan masalah hingga mengukur dampak fitur baru atau memastikan keselarasan dengan indikator kinerja kunci

bisnis (KPIs), telemetri aplikasi menginformasikan cara Anda membangun, mengoperasikan, dan mengembangkan beban kerja Anda.

Metrik, log, dan jejak merupakan tiga pilar observabilitas utama. Ketiganya berfungsi sebagai alat diagnostik yang mampu menggambarkan keadaan aplikasi Anda. Seiring waktu, tiga hal ini akan membantu menciptakan garis acuan dan mengidentifikasi setiap anomali. Namun, untuk memastikan keselarasan antara kegiatan pemantauan dan tujuan bisnis, penting untuk menentukan dan memantau. KPIs Bisnis KPIs sering membuatnya lebih mudah untuk mengidentifikasi masalah dibandingkan dengan metrik teknis saja.

Jenis telemetri lainnya, seperti pemantauan pengguna nyata (RUM) dan transaksi sintetis, melengkapi sumber data primer ini. RUMmenawarkan wawasan tentang interaksi pengguna waktu nyata, sedangkan transaksi sintetis mensimulasikan perilaku pengguna potensial, membantu mendeteksi kemacetan sebelum pengguna nyata menghadapinya.

Hasil yang diinginkan: Dapatkan wawasan yang dapat ditindaklanjuti mengenai performa beban kerja Anda. Wawasan ini akan memungkinkan Anda untuk mengambil keputusan yang proaktif tentang optimalisasi performa, mencapai peningkatan stabilitas beban kerja, merampingkan proses CI/CD, dan memanfaatkan sumber daya secara efektif.

Anti-pola umum:

- Observabilitas yang tidak lengkap: Mengabaikan penggunaan observabilitas di setiap lapisan beban kerja, sehingga mengakibatkan titik buta yang dapat membuat performa sistem vital dan wawasan perilaku menjadi tidak jelas.
- Tampilan data terfragmentasi: Ketika data tersebar di beberapa alat dan sistem, mempertahankan pandangan yang menyeluruh tentang kondisi dan performa beban kerja Anda akan menjadi sesuatu yang sulit dilakukan.
- Masalah yang dilaporkan pengguna: Tanda bahwa deteksi masalah proaktif melalui telemetri dan pemantauan bisnis kurang. KPI

Manfaat menjalankan praktik terbaik ini:

- Pengambilan keputusan berdasarkan informasi: Dengan wawasan dari telemetri dan bisnisKPIs, Anda dapat membuat keputusan berbasis data.
- Peningkatan efisiensi operasional: Pemanfaatan sumber daya berbasis data akan menghasilkan efektivitas biaya.

- Penyempurnaan stabilitas beban kerja: Deteksi dan penyelesaian masalah yang lebih cepat akan menghasilkan peningkatan waktu aktif.
- Perampingan proses CI/CD: Wawasan dari data telemetri dapat memfasilitasi penyempurnaan proses dan pengiriman kode yang andal.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk menerapkan telemetri aplikasi untuk beban kerja Anda, gunakan layanan AWS seperti Amazon dan. CloudWatch AWS X-Ray Amazon CloudWatch menyediakan rangkaian alat pemantauan yang komprehensif, memungkinkan Anda mengamati sumber daya dan aplikasi di dalam AWS dan lingkungan lokal. Layanan ini mengumpulkan, melacak, dan menganalisis metrik, menggabungkan dan memantau data log, dan memberikan respons terhadap perubahan yang terjadi dalam sumber daya Anda, menyempurnakan pemahaman Anda tentang bagaimana beban kerja Anda beroperasi. Secara bersamaan, AWS X-Ray memungkinkan Anda melacak, menganalisis, dan men-debug aplikasi Anda, memberi Anda pemahaman mendalam tentang perilaku beban kerja Anda. Dengan fitur seperti peta layanan, distribusi latensi, dan jadwal penelusuran, AWS X-Ray memberikan wawasan tentang kinerja beban kerja Anda dan hambatan yang mempengaruhinya.

Langkah-langkah implementasi

1. Identifikasi data apa yang akan dikumpulkan: Pastikan metrik, log, dan jejak penting yang akan menawarkan wawasan substansial tentang kondisi, performa, dan perilaku beban kerja Anda.
2. Menyebarkan CloudWatchagen: Agen berperan penting dalam pengadaan metrik dan log sistem dan aplikasi dari beban kerja Anda dan infrastruktur dasarnya. CloudWatch CloudWatch Agen juga dapat digunakan untuk mengumpulkan OpenTelemetry atau jejak X-Ray dan mengirimkannya ke X-Ray.
3. Menerapkan deteksi anomali untuk log dan metrik: Gunakan deteksi anomali CloudWatch Log dan deteksi anomali CloudWatch Metrik untuk secara otomatis mengidentifikasi aktivitas yang tidak biasa dalam operasi aplikasi Anda. Alat-alat ini menggunakan algoritma machine learning untuk mendeteksi dan memberikan peringatan tentang anomali yang ditemukan, yang dapat meningkatkan kemampuan pemantauan Anda dan mempercepat waktu respons terhadap adanya potensi gangguan atau ancaman keamanan. Siapkan fitur-fitur ini untuk mengelola kesehatan dan keamanan aplikasi secara proaktif.
4. Mengamankan data log sensitif: Gunakan perlindungan data Amazon CloudWatch Logs untuk menutupi informasi sensitif di dalam log Anda. Fitur ini akan membantu Anda menjaga privasi dan

kepatuhan dengan melakukan deteksi otomatis dan pengaburan data sensitif sebelum diakses. Menerapkan penyembunyian data untuk menangani dan melindungi detail sensitif dengan aman seperti informasi yang dapat diidentifikasi secara pribadi (). PII

5. Tentukan dan pantau bisnisKPIs: Tetapkan metrik khusus yang selaras dengan hasil bisnis Anda.
6. Instrumentasikan aplikasi Anda dengan AWS X-Ray: Selain menggunakan CloudWatch agen, penting untuk menginstruksikan aplikasi Anda untuk memancarkan data jejak. Proses ini dapat memberikan wawasan lebih lanjut tentang perilaku dan performa beban kerja Anda.
7. Standardisasi pengumpulan data di seluruh aplikasi Anda: Lakukan standardisasi terhadap praktik-praktik pengumpulan data di seluruh aplikasi Anda. Keseragaman bermanfaat dalam mengorelasikan dan menganalisis data, sehingga itu akan memberikan pandangan yang komprehensif tentang perilaku aplikasi Anda.
8. Menerapkan observabilitas lintas akun: Tingkatkan efisiensi pemantauan di beberapa akun dengan observabilitas CloudWatch lintas akun Akun AWS Amazon. Dengan fitur ini, Anda dapat menggabungkan metrik, log, dan alarm dari akun yang berbeda ke dalam satu tampilan, yang menyederhanakan manajemen dan meningkatkan waktu respons untuk masalah yang diidentifikasi di seluruh lingkungan organisasi Anda. AWS
9. Menganalisis dan bertindak berdasarkan data: Setelah pengumpulan dan normalisasi data dilakukan, gunakan Amazon CloudWatch untuk analisis metrik dan log, dan AWS X-Rayuntuk analisis jejak. Analisis tersebut dapat menghasilkan wawasan penting tentang kondisi, performa, dan perilaku beban kerja Anda, sehingga dapat memandu Anda dalam proses pengambilan keputusan.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Tentukan beban kerja KPIs
- OPS04-BP03 Melaksanakan telemetri aktivitas pengguna
- OPS04-BP04 Menerapkan telemetri ketergantungan
- OPS04-BP05 Menerapkan ketertelusuran transaksi

Dokumen terkait:

- Praktik Terbaik Observabilitas AWS

- Panduan Pengguna CloudWatch
- AWS X-Ray Panduan Pengembang
- Menginstrumentasikan sistem terdistribusi untuk visibilitas operasional
- Kursus Skill Builder Observabilitas AWS
- Apa yang Baru dengan Amazon CloudWatch
- Apa yang baru dengan AWS X-Ray

Video terkait:

- AWS re:invent 2022 - Praktik terbaik observabilitas di Amazon
- AWS re:invent 2022 - Mengembangkan strategi observabilitas

Contoh terkait:

- Lokakarya Satu Observabilitas
- AWS Perpustakaan Solusi: Pemantauan Aplikasi dengan Amazon CloudWatch

OPS04-BP03 Menerapkan telemetri pengalaman pengguna

Memperoleh wawasan yang mendalam tentang pengalaman dan interaksi pelanggan dengan aplikasi Anda adalah hal yang krusial. Pemantauan pengguna nyata (RUM) dan transaksi sintetis berfungsi sebagai alat yang ampuh untuk tujuan ini. RUMmenyediakan data tentang interaksi pengguna nyata yang memberikan perspektif kepuasan pengguna tanpa filter, sementara transaksi sintetis mensimulasikan interaksi pengguna, membantu mendeteksi potensi masalah bahkan sebelum berdampak pada pengguna nyata.

Hasil yang diinginkan: Pandangan yang menyeluruh tentang pengalaman pelanggan, deteksi masalah yang proaktif, dan optimalisasi interaksi pengguna untuk memberikan pengalaman digital yang mulus (seamless).

Anti-pola umum:

- Aplikasi tanpa pemantauan pengguna nyata (RUM):
  - Deteksi masalah yang tertunda: TanpaRUM, Anda mungkin tidak menyadari kemacetan atau masalah kinerja hingga pengguna mengeluh. Pendekatan reaktif ini dapat menyebabkan pelanggan menjadi tidak puas.

  - Kurangnya wawasan pengalaman pengguna: Tidak menggunakan RUM berarti Anda kehilangan data penting yang menunjukkan bagaimana pengguna nyata berinteraksi dengan aplikasi Anda, sehingga membatasi kemampuan Anda untuk mengoptimalkan pengalaman pengguna.
- Aplikasi tanpa transaksi sintetis:
  - Kasus edge yang terlewatkan: Transaksi-transaksi sintetis akan membantu Anda untuk menguji jalur dan fungsi yang mungkin jarang digunakan oleh pengguna biasa, tetapi sangat penting untuk fungsi bisnis tertentu. Tanpanya, jalur-jalur tersebut bisa mengalami kesalahan fungsi dan luput dari perhatian.
  - Memeriksa masalah saat aplikasi tidak digunakan: Pengujian sintetis yang dilakukan secara rutin dapat memberikan simulasi saat-saat ketika pengguna nyata tidak berinteraksi secara aktif dengan aplikasi Anda, sehingga hal itu akan memastikan sistem selalu berfungsi dengan benar.

Manfaat menjalankan praktik terbaik ini:

- Deteksi masalah proaktif: Identifikasi dan atasi potensi masalah sebelum berdampak pada pengguna nyata.
- Pengalaman pengguna yang dioptimalkan: Umpan balik berkelanjutan dari RUM bantuan dalam menyempurnakan dan meningkatkan pengalaman pengguna secara keseluruhan.
- Wawasan tentang performa perangkat dan browser: Memahami performa aplikasi Anda di berbagai perangkat dan browser, sehingga memungkinkan optimalisasi lebih lanjut.
- Alur kerja bisnis yang divalidasi: Transaksi-transaksi sintetis rutin akan memastikan fungsionalitas inti dan jalur-jalur penting tetap berjalan dan efisien.
- Performa aplikasi yang ditingkatkan: Manfaatkan wawasan yang dikumpulkan dari data pengguna nyata untuk meningkatkan responsivitas dan keandalan aplikasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk memanfaatkan RUM dan transaksi sintetis untuk telemetri aktivitas pengguna, AWS menawarkan layanan seperti Amazon dan CloudWatch RUMAmazon CloudWatch Synthetics. Metrik, log, dan jejak, ditambah dengan data aktivitas pengguna, memberikan sebuah pandangan yang komprehensif tentang status operasional aplikasi dan pengalaman pengguna.

Langkah-langkah implementasi

1. Menerapkan Amazon CloudWatch RUM: Integrasikan aplikasi Anda CloudWatch RUM untuk mengumpulkan, menganalisis, dan menyajikan data pengguna nyata.
   a. Gunakan CloudWatch RUM JavaScript perpustakaan untuk berintegrasi RUM dengan aplikasi Anda.
   b. Siapkan dasbor untuk memvisualisasikan dan memantau data pengguna nyata.
2. Configure CloudWatch Synthetics: Buat kenari, atau rutinitas skrip, yang mensimulasikan interaksi pengguna dengan aplikasi Anda.
   a. Tentukan alur kerja dan jalur aplikasi kritis.
   b. Desain kenari menggunakan skrip CloudWatch Synthetics untuk mensimulasikan interaksi pengguna untuk jalur ini.
   c. Jadwalkan dan pantau canary agar berjalan pada interval-interval tertentu, sehingga memastikan pemeriksaan performa yang konsisten.
3. Menganalisis dan bertindak berdasarkan data: Memanfaatkan data dari RUM dan transaksi sintetis untuk mendapatkan wawasan dan mengambil tindakan korektif ketika anomali terdeteksi. Gunakan CloudWatch dasbor dan alarm untuk tetap mendapat informasi.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS04-BP04 Menerapkan telemetri ketergantungan
- OPS04-BP05 Melaksanakan penelusuran terdistribusi

Dokumen terkait:

- CloudWatch RUMPanduan Amazon
- Panduan Amazon CloudWatch Synthetics

Video terkait:

- Optimalkan aplikasi melalui wawasan pengguna akhir dengan Amazon CloudWatch RUM
- AWS di Air ft. Pemantauan Pengguna Nyata untuk Amazon CloudWatch

Contoh terkait:

- Lokakarya Satu Observabilitas
- Git Repository untuk Amazon CloudWatch RUM Web Client
- Menggunakan Amazon CloudWatch Synthetics untuk mengukur waktu buka halaman

OPS04-BP04 Menerapkan telemetri ketergantungan

Telemetri dependensi sangat penting untuk memantau kondisi dan performa layanan dan komponen eksternal yang diandalkan oleh beban kerja Anda. Ini memberikan wawasan berharga tentang jangkauan, batas waktu, dan peristiwa penting lainnya yang terkait dengan dependensi sepertiDNS, database, atau pihak ketiga. APIs Ketika Anda menginstrumentasi aplikasi Anda agar menghasilkan metrik, log, dan jejak tentang dependensi ini, Anda mendapatkan pemahaman yang lebih jelas tentang potensi kemacetan, masalah performa, atau kegagalan yang dapat memengaruhi beban kerja Anda.

Hasil yang diinginkan: Pastikan bahwa dependensi yang diandalkan beban kerja Anda menunjukkan performa yang sesuai harapan, sehingga Anda dapat secara proaktif mengatasi masalah-masalah dan memastikan performa beban kerja yang optimal.

Anti-pola umum:

- Mengabaikan dependensi eksternal: Hanya berfokus pada metrik aplikasi internal sambil mengabaikan metrik yang berkaitan dengan dependensi eksternal.
- Kurangnya pemantauan proaktif: Menunggu masalah muncul alih-alih terus memantau kondisi dan performa dependensi.
- Pemantauan model silo: Menggunakan beberapa alat pemantauan yang berbeda-beda sehingga wawasan tentang kondisi dependensi menjadi terfragmentasi dan tidak konsisten.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan keandalan beban kerja: Dengan memastikan bahwa dependensi eksternal terus-menerus tersedia dan berkinerja optimal.

- Deteksi dan penyelesaian masalah yang lebih cepat: Secara proaktif mengidentifikasi dan menangani masalah pada dependensi sebelum berdampak pada beban kerja.
- Pandangan menyeluruh: Mendapatkan pandangan yang menyeluruh tentang komponen internal dan eksternal yang memengaruhi kondisi beban kerja.
- Peningkatan skalabilitas beban kerja: Dengan memahami batas skalabilitas dan karakteristik performa dependensi eksternal.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Implementasikan telemetri dependensi dengan memulai melakukan identifikasi layanan, infrastruktur, dan proses yang digunakan oleh beban kerja Anda. Ukur seperti apa kondisi yang baik ketika dependensi berfungsi sesuai harapan, kemudian tentukan data apa yang akan diperlukan untuk mengukur kondisi-kondisi itu. Dengan informasi tersebut, Anda dapat membuat dasbor dan peringatan yang memberikan wawasan kepada tim operasi Anda tentang status dependensi tersebut. Gunakan AWS alat untuk menemukan dan mengukur dampak ketika dependensi tidak dapat memberikan sesuai kebutuhan. Selalu tinjau ulang strategi Anda agar memperhitungkan perubahan-perubahan dalam prioritas, sasaran, dan wawasan yang diperoleh.

Langkah-langkah implementasi

Cara mengimplementasikan telemetri dependensi secara efektif:

1. Identifikasi dependensi eksternal: Lakukan kolaborasi dengan pemangku kepentingan untuk menentukan dependensi eksternal yang diandalkan oleh beban kerja Anda. Dependensi eksternal dapat mencakup layanan seperti database eksternal, pihak ketiga, rute konektivitas jaringan ke lingkungan lainAPIs, dan layanan. DNS Langkah pertama menuju telemetri dependensi yang efektif adalah memiliki pemahaman yang menyeluruh tentang apa saja dependensi tersebut.
2. Kembangkan strategi pemantauan: Setelah Anda memiliki gambaran yang jelas tentang dependensi eksternal Anda, rancanglah strategi pemantauan yang disesuaikan dengan dependensi tersebut. Ini melibatkan pemahaman kekritisan setiap ketergantungan, perilaku yang diharapkan, dan setiap perjanjian atau target tingkat layanan terkait (atau). SLA SLTs Siapkan peringatan proaktif untuk memberi tahu Anda tentang perubahan status atau penyimpangan-penyimpangan performa.
3. Gunakan pemantauan jaringan: Gunakan Monitor Internet dan Monitor Jaringan, yang memberikan wawasan komprehensif mengenai kondisi internet dan jaringan global. Alat-alat ini akan membantu

Anda untuk memahami dan merespons pemadaman (outage), gangguan, atau penurunan kinerja yang memengaruhi dependensi eksternal Anda.

4. Tetap terinformasi dengan AWS Health Dashboard: Ini memberikan peringatan dan panduan remediasi ketika AWS mengalami peristiwa yang dapat memengaruhi layanan Anda.
   a. Pantau AWS Health peristiwa dengan EventBridge aturan Amazon, atau integrasikan secara terprogram AWS Health API untuk mengotomatiskan tindakan saat Anda menerima AWS Health acara. Ini bisa berupa tindakan-tindakan umum, seperti mengirimkan semua pesan peristiwa siklus hidup yang direncanakan ke antarmuka obrolan, atau tindakan tertentu, seperti inisiasi alur kerja di alat manajemen layanan IT.
   b. Jika Anda menggunakan AWS Organizations, agregat AWS Health peristiwa di seluruh akun.
5. Instrumentasikan aplikasi Anda dengan AWS X-Ray: AWS X-Ray memberikan wawasan tentang kinerja aplikasi dan dependensi yang mendasarinya. Dengan melacak permintaan dari awal hingga akhir, Anda dapat mengidentifikasi kemacetan atau kegagalan yang terjadi dalam layanan eksternal atau komponen yang diandalkan oleh aplikasi Anda.
6. Gunakan Amazon DevOps Guru: Layanan berbasis pembelajaran mesin ini mengidentifikasi masalah operasional, memprediksi kapan masalah kritis mungkin terjadi, dan merekomendasikan tindakan spesifik yang harus diambil. Layanan ini sangat bermanfaat untuk mendapatkan wawasan tentang dependensi dan memastikan bahwa dependensi bukan merupakan sumber masalah operasional.
7. Pantau secara rutin: Terus pantau metrik dan log yang berkaitan dengan dependensi eksternal. Siapkan peringatan untuk perilaku tak terduga atau performa yang menurun.
8. Lakukan validasi setelah perubahan: Setiap kali ada pembaruan atau perubahan yang dilakukan pada salah satu dependensi eksternal, Anda harus melakukan validasi terhadap performa dan memeriksa keselarasannya dengan persyaratan-persyaratan aplikasi Anda.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Tentukan beban kerja KPIs
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS04-BP03 Melaksanakan telemetri aktivitas pengguna
- OPS04-BP05 Menerapkan ketertelusuran transaksi

- OPS08-BP04 Membuat peringatan yang dapat ditindaklanjuti

Dokumen terkait:

- Panduan AWS Health Dashboard Pengguna Pribadi Amazon
- Panduan Pengguna Monitor Internet AWS
- AWS X-Ray Panduan Pengembang
- AWS DevOpsPanduan Pengguna Guru

Video terkait:

- Visibilitas tentang bagaimana masalah internet memengaruhi performa aplikasi
- Pengantar Amazon DevOps Guru
- Mengelola peristiwa siklus hidup sumber daya dalam skala besar dengan AWS Health

Contoh terkait:

- Mendapatkan wawasan operasional dengan AIOps menggunakan Amazon Guru DevOps
- AWS Health Sadar
- Menggunakan Penyaringan Berbasis Tag untuk Mengelola AWS Health Pemantauan dan Peringatan pada Skala

OPS04-BP05 Melaksanakan penelusuran terdistribusi

Penelusuran terdistribusi menawarkan cara untuk memantau dan memvisualisasikan permintaan yang melintasi berbagai komponen sistem terdistribusi. Dengan menangkap data jejak dari berbagai sumber dan menganalisisnya dalam tampilan terpadu, tim dapat lebih memahami bagaimana permintaan mengalir, di mana kemacetan terjadi, dan di mana upaya pengoptimalan harus difokuskan.

Hasil yang diinginkan: Dapatkan tampilan menyeluruh permintaan-permintaan yang mengalir melewati sistem terdistribusi Anda, sehingga akan memungkinkan Anda untuk melakukan debugging yang presisi, performa yang dioptimalkan, dan pengalaman pengguna yang lebih baik.

Anti-pola umum:

- Instrumentasi yang tidak konsisten: Tidak semua layanan yang ada dalam sebuah sistem terdistribusi diinstrumentasi untuk penelusuran.
- Mengabaikan latensi: Hanya berfokus pada kesalahan dan tidak mempertimbangkan latensi atau penurunan performa yang terjadi secara bertahap.

Manfaat menjalankan praktik terbaik ini:

- Gambaran umum sistem yang komprehensif: Memberikan visualisasi dari seluruh jalur permintaan, dari masuk hingga keluar.
- Debugging yang disempurnakan: Mengidentifikasi dengan cepat di mana kegagalan atau masalah performa terjadi.
- Pengalaman pengguna yang ditingkatkan: Melakukan pemantauan dan optimalisasi berdasarkan data pengguna aktual, yang akan memastikan bahwa sistem memenuhi tuntutan dunia nyata.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Mulailah dengan mengidentifikasi semua elemen beban kerja Anda yang memerlukan instrumentasi. Setelah semua komponen diperhitungkan, manfaatkan alat seperti AWS X-Ray dan OpenTelemetry untuk mengumpulkan data jejak untuk analisis dengan alat seperti X-Ray dan Amazon CloudWatch ServiceLens Map. Terlibat dalam ulasan reguler dengan pengembang, dan lengkapi diskusi ini dengan alat seperti Amazon DevOps Guru, X-Ray Analytics, dan X-Ray Insights untuk membantu mengungkap temuan yang lebih dalam. Buatlah peringatan dari data jejak untuk memberikan notifikasi kapan hasil, sebagaimana didefinisikan dalam rencana pemantauan beban kerja, mengandung risiko.

Langkah-langkah implementasi

Cara mengimplementasikan penelusuran terdistribusi secara efektif:

1. Adopsi AWS X-Ray: Integrasikan X-Ray ke dalam aplikasi Anda untuk mendapatkan wawasan tentang perilakunya, memahami performanya, dan mengenali kemacetan. Manfaatkan Wawasan X-Ray untuk analisis jejak otomatis.
2. Instrumen layanan Anda: Verifikasi bahwa setiap layanan, dari AWS Lambdafungsi hingga EC2instance, mengirimkan data jejak. Semakin banyak layanan yang Anda instrumen, semakin jelas end-to-end tampilan.

3. Menggabungkan Pemantauan Pengguna CloudWatch Nyata dan pemantauan sintetis: Integrasikan Pemantauan Pengguna Nyata (RUM) dan pemantauan sintetis dengan X-Ray. Hal ini akan memungkinkan perekaman pengalaman pengguna dunia nyata dan simulasi interaksi pengguna untuk mengidentifikasi masalah-masalah potensial yang mungkin terjadi.
4. Gunakan CloudWatch agen: Agen dapat mengirim jejak dari X-Ray atau OpenTelemetry, meningkatkan kedalaman wawasan yang diperoleh.
5. Gunakan Amazon DevOps Guru: DevOps Guru menggunakan data dari X-Ray,, CloudWatch AWS Config, dan AWS CloudTrail untuk memberikan rekomendasi yang dapat ditindaklanjuti.
6. Lakukan analisis jejak: Tinjau data jejak secara rutin untuk membedakan pola, anomali, atau kemacetan yang dapat memengaruhi performa aplikasi Anda.
7. Siapkan peringatan: Konfigurasikan alarm CloudWatchuntuk pola yang tidak biasa atau latensi yang diperpanjang, memungkinkan pengalamatan masalah proaktif.
8. Peningkatan terus-menerus: Tinjau ulang strategi penelusuran Anda saat layanan ditambahkan atau dimodifikasi untuk menangkap semua titik data yang relevan.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS04-BP03 Menerapkan telemetri pengalaman pengguna
- OPS04-BP04 Menerapkan telemetri ketergantungan

Dokumen terkait:

- AWS X-Ray Panduan Pengembang
- Panduan Pengguna CloudWatch agen Amazon
- Panduan Pengguna Amazon DevOps Guru

Video terkait:

- Gunakan AWS X-Ray Wawasan

- AWS di Air ft. Observabilitas: Amazon CloudWatch dan AWS X-Ray

Contoh terkait:

- Instrumentasi aplikasi Anda untuk AWS X-Ray

## OPS5. Bagaimana cara mengurangi kecacatan, mempermudah perbaikan, dan meningkatkan aliran ke dalam produksi?

Adopsi pendekatan yang meningkatkan aliran perubahan ke dalam produksi, yang memampukan pemfaktoran ulang, umpan balik cepat atas kualitas, dan perbaikan bug. Ini mempercepat perubahan yang bermanfaat memasuki produksi, membatasi masalah yang di-deploy, dan mencapai identifikasi cepat serta perbaikan masalah akibat aktivitas deployment.

Praktik terbaik

- OPS05-BP01 Gunakan kontrol versi
- OPS05-BP02 Uji dan validasi perubahan
- OPS05-BP03 Gunakan sistem manajemen konfigurasi
- OPS05-BP04 Gunakan sistem manajemen build dan deployment
- OPS05-BP05 Lakukan manajemen tambalan
- OPS05-BP06 Bagikan standar desain
- OPS05-BP07 Menerapkan praktik untuk meningkatkan kualitas kode
- OPS05-BP08 Gunakan beberapa lingkungan
- OPS05-BP09 Lakukan perubahan yang sering, kecil, dan reversibel
- OPS05- BP1 0 Mengotomatiskan integrasi dan penyebaran sepenuhnya

### OPS05-BP01 Gunakan kontrol versi

Gunakan kontrol versi untuk memungkinkan pelacakan perubahan dan rilis.

Banyak AWS layanan menawarkan kemampuan kontrol versi. Gunakan revisi atau sistem kontrol sumber seperti AWS CodeCommit untuk mengelola kode dan artefak lainnya, seperti templat AWS CloudFormation yang dikendalikan versi infrastruktur Anda.

Hasil yang diinginkan: Tim Anda berkolaborasi dalam kode. Saat digabungkan, kode tersebut konsisten dan tidak ada perubahan yang hilang. Kesalahan mudah dibatalkan melalui penentuan versi yang benar.

Anti-pola umum:

- Anda telah mengembangkan dan menyimpan kode di stasiun kerja Anda. Anda mengalami kegagalan penyimpanan yang tidak dapat dipulihkan di stasiun kerja dan kemudian kode Anda hilang.
- Setelah menimpa kode yang ada dengan perubahan Anda, Anda dapat memulai ulang aplikasi namun aplikasi sudah tidak dapat beroperasi lagi. Anda tidak bisa membatalkan perubahan.
- Anda memiliki write lock pada file laporan yang perlu diedit orang lain. Mereka meminta Anda untuk berhenti mengerjakannya agar mereka bisa menyelesaikan tugas-tugas mereka.
- Tim penelitian Anda telah mengerjakan sebuah analisis mendetail yang membentuk pekerjaan mendatang Anda. Seseorang secara tidak sengaja menyimpan daftar belanjanya dan menimpa laporan akhir. Anda tidak bisa membatalkan perubahan dan harus membuat ulang laporan tersebut.

Manfaat menerapkan praktik terbaik ini: Dengan menggunakan kemampuan kontrol versi, Anda dapat dengan mudah kembali ke versi sebelumnya dengan status yang baik, dan membatasi risiko kehilangan aset.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pelihara aset di repositori dengan kontrol versi. Tindakan ini mendukung pelacakan perubahan, deployment versi baru, deteksi perubahan pada versi yang ada, dan pengembalian ke versi sebelumnya (misalnya, kembali ke versi dengan status yang baik apabila terjadi kegagalan). Integrasikan kemampuan kontrol versi dari sistem manajemen konfigurasi Anda ke dalam prosedur Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP04 Gunakan sistem manajemen build dan deployment

Dokumen terkait:

- Apa itu AWS CodeCommit?

Video terkait:

- Pengantar AWS CodeCommit

OPS05-BP02 Uji dan validasi perubahan

Setiap perubahan yang di-deploy harus diuji untuk menghindari kesalahan dalam lingkungan produksi. Praktik terbaik ini difokuskan untuk menguji perubahan-perubahan dari kontrol versi hingga build artefak. Di samping perubahan kode aplikasi, pengujian juga harus menyertakan infrastruktur, konfigurasi, kontrol keamanan, dan prosedur-prosedur operasi. Pengujian mengambil banyak bentuk, dari pengujian unit hingga analisis komponen perangkat lunak (SCA). Makin ke kiri pengujian dalam proses integrasi dan pengiriman perangkat lunak akan menghasilkan tingkat kepastian kualitas artefak yang lebih tinggi.

Organisasi Anda harus mengembangkan standar-standar pengujian untuk semua artefak perangkat lunak. Pengujian otomatis dapat mengurangi kerja yang melelahkan dan mencegah terjadinya kesalahan dalam pengujian manual. Uji manual mungkin diperlukan dalam beberapa kasus. Pengembang harus memiliki akses ke hasil uji otomatis untuk menciptakan loop umpan balik yang meningkatkan kualitas perangkat lunak.

Hasil yang diinginkan: Perubahan perangkat lunak Anda diuji sebelum dikirim. Pengembang memiliki akses ke hasil pengujian dan validasi. Organisasi Anda memiliki standar pengujian yang berlaku untuk semua perubahan perangkat lunak.

Anti-pola umum:

- Anda men-deploy perubahan perangkat lunak baru tanpa melakukan pengujian apa pun. Perangkat lunak gagal berjalan dalam lingkungan produksi, dan mengakibatkan matinya sistem.
- Grup keamanan baru dikerahkan dengan AWS CloudFormation tanpa diuji di lingkungan pra-produksi. Grup keamanan tersebut menjadikan aplikasi Anda tidak terjangkau oleh para pelanggan Anda.
- Sebuah metode diubah tanpa pengujian unit. Perangkat lunak gagal saat di-deploy ke lingkungan produksi.

Manfaat menerapkan praktik terbaik ini: Perubahan tingkat kegagalan deployment perangkat lunak berkurang. Kualitas perangkat lunak meningkat. Pengembang memiliki kesadaran yang lebih tinggi

tentang kelayakan kode mereka. Kebijakan keamanan dapat diluncurkan dengan penuh keyakinan untuk mendukung kepatuhan organisasi. Perubahan infrastruktur, misalnya pembaruan kebijakan penskalaan otomatis, diuji di awal untuk memenuhi kebutuhan lalu lintas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pengujian dilakukan pada semua perubahan, dari kode aplikasi hingga infrastruktur, sebagai bagian dari praktik integrasi berkelanjutan yang Anda lakukan. Hasil-hasil pengujian dipublikasikan sehingga pengembang memiliki umpan balik dengan cepat. Organisasi memiliki standar pengujian bahwa semua perubahan harus lulus.

Gunakan kekuatan AI generatif dengan Pengembang Amazon Q untuk meningkatkan produktivitas pengembang dan kualitas kode Anda. Pengembang Amazon Q menyertakan pembuatan saran kode (berdasarkan model bahasa besar), produksi pengujian unit (termasuk kondisi batas), dan peningkatan keamanan kode melalui deteksi dan perbaikan kerentanan keamanan.

Contoh pelanggan

Sebagai bagian dari jalur integrasi berkelanjutan mereka, AnyCompany Retail melakukan beberapa jenis pengujian pada semua artefak perangkat lunak. Mereka mempraktikkan pengembangan yang didorong pengujian sehingga semua perangkat lunak memiliki pengujian-pengujian unit. Setelah artefak dibangun, mereka menjalankan end-to-end tes. Setelah pengujian putaran pertama selesai, mereka menjalankan pemindaian keamanan aplikasi statis, yang mencari kerentanan yang dikenali. Pengembang menerima pesan setelah setiap gerbang pengujian dilalui. Setelah semua pengujian selesai, artefak perangkat lunak kemudian disimpan di dalam sebuah repositori artefak.

Langkah-langkah implementasi

1. Bekerjalah dengan para pemangku kepentingan yang ada di organisasi Anda untuk mengembangkan sebuah standar pengujian untuk artefak perangkat lunak. Pengujian standar apa yang harus dilalui oleh semua artefak? Apakah ada persyaratan kepatuhan atau tata kelola yang harus disertakan di dalam cakupan pengujian? Apakah Anda perlu melakukan pengujian kualitas kode? Setelah pengujian selesai dilakukan, siapa yang perlu mengetahuinya?
    1. Arsitektur Referensi Pipeline Deployment AWS berisi daftar tepercaya untuk jenis-jenis pengujian yang dapat dilakukan pada artefak perangkat lunak sebagai bagian dari pipeline integrasi.

2. Instrumentasikan aplikasi Anda dengan pengujian-pengujian yang diperlukan berdasarkan standar pengujian perangkat lunak Anda. Setiap set pengujian harus diselesaikan dalam waktu kurang dari sepuluh menit. Pengujian harus berjalan sebagai bagian dari pipeline integrasi.
   a. Gunakan Pengembang Amazon Q, sebuah alat AI generatif yang dapat membantu Anda membuat kasus pengujian unit (termasuk ketentuan batas), yang dapat menghasilkan fungsi dengan menggunakan kode dan komentar, dan menerapkan algoritme terkenal.
   b. Gunakan Amazon CodeGuru Reviewer untuk menguji kode aplikasi Anda untuk cacat.
   c. Anda dapat menggunakan AWS CodeBuild untuk melakukan pengujian pada artefak perangkat lunak.
   d. AWS CodePipeline dapat mengorkestrasi pengujian perangkat lunak Anda ke dalam pipeline.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP01 Gunakan kontrol versi
- OPS05-BP06 Bagikan standar desain
- OPS05-BP07 Menerapkan praktik untuk meningkatkan kualitas kode
- OPS05- BP1 0 Mengotomatiskan integrasi dan penyebaran sepenuhnya

Dokumen terkait:

- Adopsi pendekatan pengembangan berbasis pengujian
- Mengakselerasi Siklus Hidup Pengembangan Perangkat Lunak Anda dengan Amazon Q
- Pengembang Amazon Q, sekarang tersedia secara umum, menyertakan pratinjau kemampuan-kemampuan baru untuk menata kembali pengalaman pengembang
- Lembar Cheat Utama untuk Menggunakan Pengembang Amazon Q di Anda IDE
- Beban kerja Shift-Left, memanfaatkan AI untuk Pembuatan Uji
- Pusat Pengembang Amazon Q
- 10 cara untuk membangun aplikasi lebih cepat dengan Amazon CodeWhisperer
- Melihat melampaui cakupan kode dengan Amazon CodeWhisperer
- Praktik Terbaik untuk Rekayasa Cepat dengan Amazon CodeWhisperer
- Pipa AWS CloudFormation Pengujian Otomatis dengan TaskCat dan CodePipeline

- Membangun pipa end-to-end AWS DevSecOps CI/CD dengan open sourceSCA,SAST, dan alat DAST
- Memulai pengujian aplikasi nirserver
- Pipeline CI/CD adalah pemandu utama rilisan saya
- Laporan Resmi tentang Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan di AWS

Video terkait:

- Menerapkan API dengan Agen Pengembang Amazon Q untuk Pengembangan Perangkat Lunak
- Menginstal, Mengkonfigurasi, & Menggunakan Amazon Q Developer dengan JetBrains IDEs (How-to)
- Menguasai seni Amazon CodeWhisperer - daftar putar YouTube
- AWS re: invent 2020: Infrastruktur yang dapat diuji: Pengujian integrasi pada AWS
- AWS Summit ANZ 2021 - Mendorong strategi uji-pertama dengan CDK dan pengembangan berbasis pengujian
- Menguji Infrastruktur Anda sebagai Kode dengan AWS CDK

Sumber daya terkait:

- Membangun aplikasi menggunakan AI generatif dengan Amazon CodeWhisperer
- CodeWhisperer Lokakarya Amazon
- Arsitektur Referensi Pipeline Deployment AWS - Aplikasi
- AWS Pipa Kubernetes DevSecOps
- Lokakarya Kebijakan sebagai Kode – Pengembangan yang Didorong Pengujian
- Jalankan pengujian unit untuk aplikasi Node.js dari GitHub dengan menggunakan AWS CodeBuild
- Menggunakan Serverspec untuk pengembangan kode infrastruktur yang didorong pengujian

Layanan terkait:

- Pengembang Amazon Q
- CodeGuru Peninjau Amazon

- AWS CodeBuild
- AWS CodePipeline

OPS05-BP03 Gunakan sistem manajemen konfigurasi

Gunakan sistem manajemen konfigurasi untuk membuat dan melacak perubahan konfigurasi. Sistem ini mengurangi kesalahan yang disebabkan oleh proses manual dan meminimalkan tingkat upaya untuk melakukan deployment perubahan.

Manajemen konfigurasi statis menetapkan nilai saat melakukan inisialisasi atas sebuah sumber daya yang diharapkan tetap konsisten selama masa pakai sumber daya tersebut. Manajemen konfigurasi dinamis menetapkan nilai saat inisialisasi. Nilai ini dapat atau diharapkan berubah selama masa pakai sumber daya. Misalnya, Anda dapat mengatur sebuah pengalih fitur untuk mengaktifkan fungsionalitas dalam kode Anda melalui sebuah perubahan konfigurasi, atau mengubah tingkat kerincian log selama insiden.

Konfigurasi harus di-deploy dalam status yang diketahui dan konsisten. Anda harus menggunakan inspeksi otomatis untuk selalu memantau konfigurasi sumber daya di seluruh lingkungan dan wilayah. Kontrol ini harus didefinisikan sebagai kode dan manajemen otomatis untuk memastikan aturan diterapkan secara konsisten di seluruh lingkungan. Perubahan konfigurasi harus diperbarui melalui prosedur kontrol perubahan yang disepakati dan diterapkan secara konsisten dan menghormati kontrol versi. Konfigurasi aplikasi harus dikelola secara independen dan tidak bergantung pada kode aplikasi dan infrastruktur. Hal ini memungkinkan deployment yang konsisten di banyak lingkungan. Perubahan konfigurasi tidak akan mengakibatkan pembangunan kembali atau pemindahan aplikasi.

Hasil yang diinginkan: Anda mengonfigurasi, memvalidasi, dan melakukan deployment sebagai bagian dari pipeline integrasi berkelanjutan, pengiriman berkelanjutan (CI/CD) Anda. Anda memantau untuk memvalidasi bahwa konfigurasi sudah benar. Hal ini akan meminimalkan dampak apa pun yang terjadi terhadap pelanggan dan pengguna akhir.

Anti-pola umum:

- Anda memperbarui konfigurasi server web secara manual di seluruh armada dan beberapa server menjadi tidak responsif karena terjadinya kesalahan pembaruan.
- Anda secara manual memperbarui armada server aplikasi Anda selama berjam-jam. Ketidaksesuaian dalam konfigurasi selama terjadi perubahan dapat menyebabkan perilaku yang tak terduga.

- Seseorang telah memperbarui grup keamanan Anda dan server web Anda tidak dapat diakses lagi. Tanpa mengetahui apa yang telah diubah, Anda menghabiskan banyak waktu untuk menyelidiki masalah tersebut sehingga waktu pemulihan akan semakin panjang.
- Anda mendorong konfigurasi pra-produksi ke dalam lingkungan produksi melalui CI/CD tanpa melakukan validasi. Anda mengekspos pengguna dan pelanggan ke data dan layanan yang salah.

Manfaat menerapkan praktik terbaik ini: Mengadopsi sistem manajemen konfigurasi akan meminimalkan tingkat upaya untuk membuat dan melacak perubahan, serta akan mengurangi frekuensi kesalahan yang disebabkan oleh penggunaan prosedur manual. Sistem manajemen konfigurasi dapat memberikan jaminan sehubungan dengan persyaratan tata kelola, kepatuhan, dan peraturan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sistem manajemen konfigurasi digunakan untuk melacak dan mengimplementasikan perubahan-perubahan yang dibuat pada konfigurasi aplikasi dan lingkungan. Sistem manajemen konfigurasi juga digunakan untuk mengurangi terjadinya kesalahan yang disebabkan oleh proses-proses yang diselesaikan secara manual, membuat perubahan konfigurasi menjadi dapat diulang dan dapat diaudit, serta mengurangi tingkat upaya.

AWS Aktif, Anda dapat menggunakan AWS Configuntuk terus memantau konfigurasi AWS sumber daya Anda di seluruh akun dan Wilayah. Dengan demikian, Anda dapat melacak riwayat konfigurasi mereka, memahami bagaimana perubahan konfigurasi akan memengaruhi sumber daya lainnya, dan mengauditnya terhadap konfigurasi yang diharapkan atau diinginkan dengan menggunakan Aturan AWS Config dan AWS Config Conformance Packs.

Untuk konfigurasi dinamis dalam aplikasi yang berjalan di EC2 instans Amazon, container, aplikasi seluler AWS Lambda, atau perangkat IoT, Anda dapat menggunakannya AWS AppConfiguntuk mengonfigurasi, memvalidasi, menerapkan, dan memantaunya di seluruh lingkungan.

Langkah-langkah implementasi

1. Lakukan identifikasi pemilik konfigurasi.
   a. Buat agar para pemilik konfigurasi menyadari tentang kepatuhan, tata kelola, atau peraturan apa pun.
2. Lakukan identifikasi terhadap item-item konfigurasi dan hasil kerja.

   a. Item-item konfigurasi adalah semua konfigurasi aplikasi dan lingkungan yang dipengaruhi oleh sebuah deployment yang dilakukan di dalam pipeline CI/CD Anda.
   b. Hasil kerja antara lain kriteria keberhasilan, validasi, dan hal-hal yang harus dipantau.
3. Pilihlah alat-alat yang bisa digunakan untuk melakukan manajemen konfigurasi berdasarkan kebutuhan bisnis dan pipeline pengiriman Anda.
4. Pertimbangkan deployment tertimbang seperti deployment canary untuk perubahan-perubahan konfigurasi yang signifikan guna meminimalkan dampak konfigurasi yang salah.
5. Integrasikan manajemen konfigurasi Anda ke dalam pipeline CI/CD Anda.
6. Validasikan semua perubahan yang didorong.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS06-BP01 Rencana untuk perubahan yang gagal
- OPS06-BP02 Uji penerapan
- OPS06-BP03 Menggunakan strategi penyebaran yang aman
- OPS06-BP04 Mengotomatiskan pengujian dan rollback

Dokumen terkait:

- AWS Control Tower
- AWS Akselerator Zona Pendaratan
- AWS Config
- Apa itu AWS Config?
- AWS AppConfig
- Apa itu AWS CloudFormation?
- Alat Pengembang AWS

Video terkait:

- AWS re:invent 2022 - Tata kelola proaktif dan kepatuhan untuk beban kerja AWS
- AWS re:invent 2020: Mencapai kepatuhan sebagai menggunakan kode AWS Config
- Mengelola dan Menyebarkan Konfigurasi Aplikasi dengan AWS AppConfig

OPS05-BP04 Gunakan sistem manajemen build dan deployment

Gunakan sistem manajemen build dan deployment. Sistem ini mengurangi kesalahan yang disebabkan oleh proses manual dan meminimalkan tingkat upaya untuk melakukan deployment perubahan.

Di AWS, Anda dapat membangun pipeline continuous integration/continuous deployment (CI/CD) menggunakan layanan seperti AWS Developer Tools (misalnya,,,, AWS CodeCommit, AWS CodeBuilddan). AWS CodePipelineAWS CodeDeployAWS CodeStar

Hasil yang diinginkan: Sistem manajemen build dan deployment Anda mendukung sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) yang ada di organisasi Anda yang menyediakan kemampuan-kemampuan untuk mengotomatisasi peluncuran yang aman dengan konfigurasi yang benar.

Anti-pola umum:

- Setelah menyusun kode Anda pada sistem pengembangan, Anda menyalin file yang dapat dieksekusi ke sistem produksi namun file tersebut gagal memulai. File log lokal mengindikasikan bahwa kegagalan tersebut dikarenakan terjadinya kehilangan dependensi.
- Anda berhasil membangun aplikasi Anda dengan fitur-fitur baru pada lingkungan pengembangan Anda dan memberikan kodenya ke tim jaminan kualitas (QA). Kode tersebut gagal dalam pengujian QA karena ada aset statis yang hilang.
- Pada hari Jumat, setelah melakukan upaya keras, Anda berhasil membangun aplikasi Anda secara manual di lingkungan pengembangan Anda termasuk fitur-fitur yang baru Anda kodekan. Pada hari Senin, Anda tidak dapat mengulangi langkah-langkah yang membuat Anda berhasil membangun aplikasi tersebut.
- Anda melakukan pengujian yang telah Anda siapkan untuk rilisan baru Anda. Kemudian Anda menghabiskan minggu selanjutnya untuk mempersiapkan lingkungan pengujian dan melakukan seluruh pengujian integrasi yang ada sekarang disusul dengan pengujian kinerja. Kode baru tersebut ternyata memiliki dampak kinerja yang tidak dapat diterima dan harus dikembangkan ulang dan kemudian diuji ulang.

Manfaat menerapkan praktik terbaik ini: Dengan menyediakan mekanisme untuk mengatasi aktivitas build dan deployment, Anda mengurangi upaya yang diperlukan untuk melakukan tugas-tugas berulang, membebaskan anggota tim Anda untuk fokus pada tugas-tugas kreatif mereka yang berharga, serta mengurangi terjadinya kesalahan akibat pelaksanaan prosedur yang dilakukan secara manual.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sistem manajemen build dan deployment digunakan untuk melacak dan mengimplementasikan perubahan, mengurangi kesalahan yang disebabkan oleh proses yang dilakukan secara manual, dan mengurangi upaya yang diperlukan untuk melakukan deployment dengan aman. Lakukan otomatisasi sepenuhnya terhadap pipeline integrasi dan deployment dari check-in kode hingga build, pengujian, deployment, dan validasi. Hal ini akan mempersingkat waktu tunggu (lead time), mengurangi biaya, mendorong peningkatan frekuensi perubahan, mengurangi tingkat upaya, dan meningkatkan kolaborasi.

Langkah-langkah implementasi

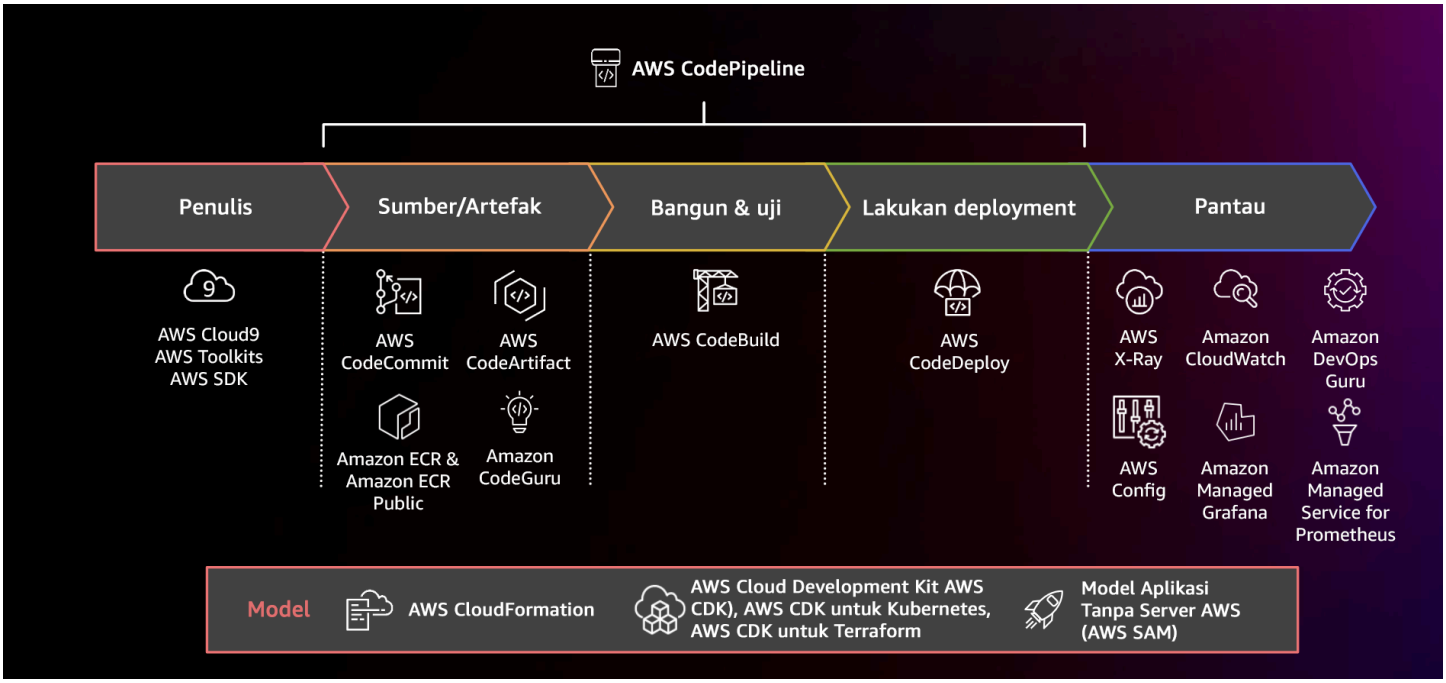



Diagram yang menunjukkan pipa CI/CD menggunakan AWS CodePipeline dan layanan terkait

1. Gunakan AWS CodeCommit untuk mengontrol versi, menyimpan, dan mengelola aset (seperti dokumen, kode sumber, dan file biner).
2. Gunakan CodeBuild untuk mengkompilasi kode sumber Anda, menjalankan pengujian unit, dan menghasilkan artefak yang siap digunakan.
3. Gunakan CodeDeploy sebagai layanan penerapan yang mengotomatiskan penerapan aplikasi ke instans EC2Amazon, instans lokal, fungsi tanpa server, atau Amazon. AWS Lambda ECS
4. Pantau deployment Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS06-BP04 Mengotomatiskan pengujian dan rollback

Dokumen terkait:

- Alat Pengembang AWS
- Apa itu AWS CodeCommit?
- Apa itu AWS CodeBuild?
- AWS CodeBuild
- Apa itu AWS CodeDeploy?

Video terkait:

- AWS Re:invent 2022 - Praktik terbaik Well-Architected AWS untuk di DevOps AWS

OPS05-BP05 Lakukan manajemen tambalan

Lakukan manajemen patch untuk mendapatkan fitur, menangani permasalahan, dan menjaga kepatuhan terhadap tata kelola. Otomatiskan manajemen patch untuk mengurangi kesalahan yang disebabkan oleh proses manual, menskalakan, dan mengurangi upaya untuk melakukan patch.

Manajemen patch dan kerentanan adalah bagian dari aktivitas manajemen manfaat dan risiko Anda. Lebih baik Anda memiliki infrastruktur tetap dan melakukan deployment beban kerja pada status yang diketahui baik dan terverifikasi. Jika tidak memungkinkan, opsi yang tersisa ialah menerapkan patching.

Amazon EC2 Image Builder menyediakan saluran pipa untuk memperbarui gambar mesin. Sebagai bagian dari manajemen tambalan, pertimbangkan Amazon Machine Images (AMIs) menggunakan pipeline AMI gambar atau image container dengan pipeline image Docker, sekaligus AWS Lambda menyediakan pola untuk runtime kustom dan library tambahan untuk menghapus kerentanan.

Anda harus mengelola pembaruan untuk Amazon Machine Images untuk gambar Linux atau Windows Server menggunakan Amazon EC2 Image Builder. Anda dapat menggunakan Amazon Elastic Container Registry (AmazonECR) dengan pipeline yang ada untuk mengelola ECS gambar Amazon dan mengelola EKS gambar Amazon. Lambda menyertakan fitur manajemen versi.

Patching tidak boleh dilakukan pada sistem produksi tanpa mengujinya terlebih dahulu di sebuah lingkungan yang aman. Patch hanya bisa diterapkan jika mendukung hasil operasi atau bisnis. Pada AWS, Anda dapat menggunakan AWS Systems Manager Patch Manager untuk mengotomatiskan proses menambal sistem terkelola dan menjadwalkan aktivitas menggunakan Systems Manager Maintenance Windows.

Hasil yang diinginkan: Gambar Anda AMI dan kontainer ditambal up-to-date,, dan siap diluncurkan. Anda dapat melacak status dari semua citra yang di-deploy dan mengetahui kepatuhan patch. Anda dapat melaporkan status saat ini dan memiliki proses untuk memenuhi kebutuhan-kebutuhan kepatuhan Anda.

Anti-pola umum:

- Anda diberi tugas untuk menerapkan semua patch keamanan baru dalam waktu dua jam yang menyebabkan terjadinya beberapa kali pemadaman akibat ketidaksesuaian aplikasi dengan patch.
- Pustaka yang tidak di-patch dapat menimbulkan konsekuensi yang tidak dinginkan karena pihak yang tidak diketahui memanfaatkan kerentanan di dalamnya untuk mengakses beban kerja Anda.
- Anda melakukan patching pada lingkungan pengembangan secara otomatis tanpa memberikan notifikasi kepada pengembang. Anda menerima beberapa keluhan dari pengembang bahwa lingkungan mereka tidak lagi beroperasi sesuai dengan yang diharapkan.
- Anda belum menambal off-the-shelf perangkat lunak komersial pada instance persisten. Ketika Anda mengalami masalah pada perangkat lunak dan menghubungi vendornya, Anda diberi tahu bahwa versi tersebut tidak didukung dan Anda harus melakukan patching pada tingkat tertentu untuk menerima bantuan.
- Patch yang baru-baru ini dirilis untuk perangkat lunak enkripsi yang Anda gunakan memiliki peningkatan kinerja yang signifikan. Sistem Anda yang tidak di-patching tetap memiliki masalah kinerja akibat tidak dilakukannya patching.
- Anda mendapatkan notifikasi tentang kerentanan zero-day yang memerlukan perbaikan darurat dan Anda harus menerapkan patching pada semua lingkungan Anda secara manual.

Manfaat menerapkan praktik terbaik ini: Dengan menjalankan proses manajemen patching, termasuk kriteria Anda untuk patching dan metodologi untuk distribusi ke seluruh lingkungan Anda, Anda dapat menskalakan dan melaporkan tingkat patching Anda. Ini memberikan jaminan seputar patching keamanan dan memastikan visibilitas yang jelas tentang status perbaikan yang diketahui yang sekarang sedang dilakukan. Hal ini mendorong adopsi fitur dan kemampuan yang diinginkan, penyingkiran masalah secara cepat, dan kepatuhan yang berkelanjutan terhadap tata kelola.

Implementasikan sistem manajemen dan otomatisasi patching untuk mengurangi tingkat upaya untuk men-deploy patch dan mengurangi kesalahan yang disebabkan oleh proses yang dilakukan secara manual.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Lakukan patching pada sistem untuk menyelesaikan masalah, untuk mendapatkan fitur atau kemampuan yang diinginkan, dan untuk tetap patuh terhadap kebijakan tata kelola serta persyaratan dukungan vendor. Pada sistem tetap, lakukan deployment dengan rangkaian patching yang sesuai untuk mencapai hasil yang diinginkan. Lakukan otomatisasi mekanisme manajemen patching untuk mengurangi waktu yang telah berlalu untuk melakukan patching, untuk mencegah kesalahan yang disebabkan oleh proses manual, dan mengurangi upaya dalam melakukan patching.

Langkah-langkah implementasi

Untuk Amazon EC2 Image Builder:

1. Menggunakan Amazon EC2 Image Builder, tentukan detail pipeline:
    a. Buatlah sebuah pipeline citra dan beri nama
    b. Tentukan jadwal pipeline dan zona waktu
    c. Konfigurasikan dependensi apa pun
2. Pilih resep:
    a. Pilih resep yang sudah ada atau buat resep baru
    b. Pilih jenis citra
    c. Beri nama dan versi resep Anda
    d. Pilih citra dasar Anda
    e. Tambahkan komponen build dan tambahkan ke registri target
3. Opsional - tentukan konfigurasi infrastruktur Anda.
4. Opsional - tentukan pengaturan konfigurasi.
5. Tinjau pengaturan.
6. Pertahankan kebersihan resep secara teratur.

Untuk Systems Manager Patch Manager:

1. Membuat acuan dasar patch.
2. Pilih metode operasi patching.
3. Aktifkan pelaporan dan pemindaian kepatuhan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS06-BP04 Mengotomatiskan pengujian dan rollback

Dokumen terkait:

- Apa itu Amazon EC2 Image Builder
- Membuat pipeline gambar menggunakan Amazon EC2 Image Builder
- Buat pipeline citra kontainer
- AWS Manajer Patch Systems Manager
- Menggunakan Patch Manager
- Menggunakan laporan kepatuhan patch
- AWS Alat Pengembang

Video terkait:

- CI/CD untuk Aplikasi Tanpa Server di AWS
- Mendesain dengan Mempertimbangkan Operasional

Contoh terkait:

- Lab Well-Architected - Manajemen Inventaris dan Patch
- AWS Systems Manager Tutorial Patch Manager

OPS05-BP06 Bagikan standar desain

Bagikan praktik terbaik kepada seluruh tim untuk meningkatkan kesadaran dan memaksimalkan manfaat dari upaya-upaya pengembangan. Dokumentasikan dan jaga agar hal ini selalu mutakhir seiring perkembangan yang terjadi pada arsitektur Anda. Jika standar bersama telah diterapkan di

dalam organisasi Anda, tersedianya mekanisme merupakan hal yang sangat penting untuk meminta penambahan, perubahan, dan pengecualian terhadap standar. Tanpa opsi ini, standar akan menjadi penghambat inovasi.

Hasil yang diinginkan: Standar desain dibagikan ke semua tim yang ada dalam organisasi Anda. Mereka didokumentasikan dan disimpan up-to-date sebagai praktik terbaik berkembang.

Anti-pola umum:

- Dua tim pengembangan masing-masing telah membuat sebuah layanan autentikasi pengguna. Pengguna Anda harus mempertahankan rangkaian kredensial terpisah untuk masing-masing bagian sistem yang ingin diakses.
- Setiap tim mengelola infrastruktur mereka sendiri. Persyaratan kepatuhan baru memaksakan penerapan sebuah perubahan pada infrastruktur Anda dan setiap tim mengimplementasikannya dengan cara yang berbeda.

Manfaat menerapkan praktik terbaik ini: Penggunaan standar bersama akan mendukung adopsi praktik terbaik dan memaksimalkan manfaat dari upaya-upaya pengembangan yang dilakukan. Mendokumentasikan dan memperbarui standar desain membuat organisasi Anda tetap up-to-date dengan praktik terbaik serta persyaratan keamanan dan kepatuhan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Berbagi praktik terbaik, standar desain, daftar periksa, prosedur operasi, panduan, dan persyaratan tata kelola yang ada sekarang dengan semua tim. Buatlah prosedur-prosedur untuk meminta perubahan, penambahan, dan pengecualian standar desain untuk mendukung peningkatan dan inovasi. Buat tim mengetahui tentang konten yang dipublikasikan. Memiliki mekanisme untuk menjaga standar desain up-to-date saat praktik terbaik baru muncul.

Contoh pelanggan

AnyCompany Retail memiliki tim arsitektur lintas fungsi yang menciptakan pola arsitektur perangkat lunak. Tim ini membangun arsitektur dengan kepatuhan dan tata kelola bawaan. Tim yang mengadopsi standar bersama ini mendapatkan manfaat dari memiliki kepatuhan dan tata kelola bawaan. Mereka dapat membangun di atas standar desain dengan cepat. Tim arsitektur mengadakan pertemuan setiap tiga bulan untuk mengevaluasi pola-pola arsitektur dan melakukan pembaruan, jika perlu.

Langkah-langkah implementasi

1. Identifikasikan sebuah tim lintas fungsi yang memegang kepemilikan atas pengembangan dan pembaruan standar desain. Tim ini harus bekerja sama dengan para pemangku kepentingan yang ada di seluruh organisasi Anda untuk mengembangkan standar desain, standar operasi, daftar periksa, panduan, dan persyaratan tata kelola. Dokumentasikan standar desain dan bagikan dalam organisasi Anda.
   a. AWS Service Catalog dapat digunakan untuk membuat portofolio yang mewakili standar desain dengan menggunakan infrastruktur sebagai kode. Anda dapat berbagi portofolio dengan semua akun.
2. Miliki mekanisme untuk menjaga standar desain up-to-date karena praktik terbaik baru diidentifikasi.
3. Jika standar desain diterapkan secara terpusat, Anda harus memiliki proses untuk meminta perubahan, pembaruan, dan pengecualian.

Tingkat upaya untuk rencana implementasi: Sedang. Untuk mengembangkan sebuah proses untuk membuat dan berbagi standar desain mungkin diperlukan kerja sama dan koordinasi dengan para pemangku kepentingan yang ada di seluruh organisasi Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP03 Mengevaluasi persyaratan tata kelola - Persyaratan tata kelola memengaruhi standar desain.
- OPS01-BP04 Mengevaluasi persyaratan kepatuhan - Kepatuhan adalah input penting dalam membuat standar desain.
- OPS07-BP02 Memastikan tinjauan kesiapan operasional yang konsisten - Daftar periksa kesiapan operasional merupakan sebuah mekanisme untuk mengimplementasikan standar desain ketika Anda merancang desain beban kerja Anda.
- OPS11-BP01 Memiliki proses untuk perbaikan berkelanjutan - Memperbarui standar desain merupakan bagian dari peningkatan berkelanjutan.
- OPS11-BP04 Melakukan manajemen pengetahuan - Sebagai bagian dari praktik manajemen pengetahuan Anda, dokumentasikan dan bagikan standar desain.

Dokumen terkait:

- Otomatiskan AWS Backup s dengan AWS Service Catalog
- AWS Service Catalog Akun Factory-Enhanced
- Bagaimana Expedia Group membangun Database as a Service (DBaaS) menawarkan menggunakan AWS Service Catalog
- Mempertahankan visibilitas tentang penggunaan pola-pola arsitektur cloud
- Sederhanakan berbagi AWS Service Catalog portofolio Anda dalam pengaturan AWS Organizations

Video terkait:

- AWS Service Catalog — Memulai
- AWS re:invent 2020: Kelola AWS Service Catalog portofolio Anda seperti seorang ahli

Contoh terkait:

- AWS Service Catalog Arsitektur Referensi
- AWS Service Catalog Lokakarya

Layanan terkait:

- AWS Service Catalog

OPS05-BP07 Menerapkan praktik untuk meningkatkan kualitas kode

Implementasikan praktik untuk meningkatkan kualitas kode dan meminimalkan kecacatan. Beberapa contohnya termasuk, pengembangan yang didorong pengujian, peninjauan kode, pengadopsian standar, dan pemrograman berpasangan. Sertakan praktik-praktik ini ke dalam integrasi berkelanjutan dan proses penyampaian hasil Anda.

Hasil yang diinginkan: Organisasi Anda menggunakan praktik-praktik terbaik seperti peninjauan kode atau pemrograman berpasangan untuk meningkatkan kualitas kode. Pengembang dan operator mengadopsi praktik-praktik terbaik dalam kualitas kode sebagai bagian dari siklus hidup pengembangan perangkat lunak.

Anti-pola umum:

- Anda mempercayakan kode ke cabang utama aplikasi tanpa peninjauan kode. Perubahan otomatis melakukan deployment ke lingkungan produksi dan menyebabkan penghentian produksi.
- Aplikasi baru dikembangkan tanpa unit, end-to-end, atau tes integrasi apa pun. Tidak ada cara untuk menguji aplikasi sebelum deployment.
- Tim Anda membuat perubahan-perubahan manual pada lingkungan produksi untuk mengatasi kecacatan. Perubahan tidak melalui proses pengujian atau peninjauan kode dan tidak direkam atau dicatat log-nya melalui proses penyampaian hasil dan integrasi berkelanjutan.

Manfaat menerapkan praktik terbaik ini: Dengan mengadopsi praktik-praktik untuk meningkatkan kualitas kode, Anda dapat membantu meminimalkan masalah-masalah yang terjadi di lingkungan produksi. Kualitas kode memudahkan penggunaan praktik-praktik terbaik seperti pemrograman berpasangan, tinjauan kode, dan implementasi alat-alat produktivitas AI.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Implementasikan praktik-praktik untuk meningkatkan kualitas kode guna meminimalkan terjadinya kecacatan sebelum dilakukan deployment terhadapnya. Gunakan praktik-praktik, misalnya pengembangan berbasis pengujian, peninjauan kode, dan pemrograman berpasangan, untuk meningkatkan kualitas pengembangan Anda.

Gunakan kekuatan AI generatif dengan Pengembang Amazon Q untuk meningkatkan produktivitas pengembang dan kualitas kode Anda. Pengembang Amazon Q menyertakan pembuatan saran kode (berdasarkan model bahasa besar), produksi pengujian unit (termasuk kondisi batas), dan peningkatan keamanan kode melalui deteksi dan perbaikan kerentanan keamanan.

Contoh pelanggan

AnyCompany Retail mengadopsi beberapa praktik untuk meningkatkan kualitas kode. Mereka telah mengadopsi pengembangan berbasis pengujian sebagai standar untuk menulis aplikasi. Untuk beberapa fitur baru, pengembang mereka akan memasangkan program menjadi satu saat proses sprint. Setiap permintaan penarikan (pull request) akan melewati peninjauan kode oleh pengembang senior sebelum diintegrasikan dan dilakukan deployment.

Langkah-langkah implementasi

1. Adopsi praktik-praktik kualitas kode, misalnya praktik pengembangan berbasis pengujian, peninjauan kode, dan pemrograman berpasangan, ke dalam proses penyampaian hasil dan

integrasi berkelanjutan Anda. Gunakan teknik-teknik ini untuk meningkatkan kualitas perangkat lunak.

a. Gunakan Pengembang Amazon Q, sebuah alat AI generatif yang dapat membantu Anda membuat kasus pengujian unit (termasuk ketentuan batas), yang dapat menghasilkan fungsi menggunakan kode dan komentar, menerapkan algoritme terkenal, mendeteksi pelanggaran kebijakan keamanan dan kerentanan dalam kode Anda, mendeteksi rahasia, memindai infrastruktur sebagai kode (IaC), kode dokumen, dan mempelajari pustaka kode pihak ketiga dengan lebih cepat.
b. Amazon CodeGuru Reviewer dapat memberikan rekomendasi pemrograman untuk kode Java dan Python menggunakan pembelajaran mesin.
c. Anda dapat membuat lingkungan pengembangan bersama dengan AWS Cloud9 tempat Anda dapat berkolaborasi dalam mengembangkan kode.

Tingkat upaya untuk rencana implementasi: Sedang. Ada banyak cara untuk mengimplementasikan praktik-praktik terbaik ini, tetapi membuat organisasi mau mengadopsinya mungkin akan menjadi hal yang sulit.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP02 Uji dan validasi perubahan
- OPS05-BP06 Bagikan standar desain

Dokumen terkait:

- Adopsi pendekatan pengembangan berbasis pengujian
- Mengakselerasi Siklus Hidup Pengembangan Perangkat Lunak Anda dengan Amazon Q
- Pengembang Amazon Q, sekarang tersedia secara umum, menyertakan pratinjau kemampuan-kemampuan baru untuk menata kembali pengalaman pengembang
- Lembar Cheat Utama untuk Menggunakan Pengembang Amazon Q di Anda IDE
- Beban kerja Shift-Left, memanfaatkan AI untuk Pembuatan Uji
- Pusat Pengembang Amazon Q
- 10 cara untuk membangun aplikasi lebih cepat dengan Amazon CodeWhisperer
- Melihat melampaui cakupan kode dengan Amazon CodeWhisperer

- Praktik Terbaik untuk Rekayasa Cepat dengan Amazon CodeWhisperer
- Panduan Perangkat Lunak Tangkas
- Pipeline CI/CD adalah pemandu utama rilisan saya
- Otomatiskan ulasan kode dengan Amazon CodeGuru Reviewer
- Adopsi pendekatan pengembangan berbasis pengujian
- Bagaimana DevFactory membangun aplikasi yang lebih baik dengan Amazon CodeGuru
- Tentang Pemrograman Berpasangan
- RENGAInc. mengotomatiskan ulasan kode dengan Amazon CodeGuru
- Seni Pengembangan Tangkas: Pengembangan yang Didorong Pengujian
- Mengapa peninjauan kode itu penting (dan sesungguhnya menghemat waktu!)

Video terkait:

- Menerapkan API dengan Agen Pengembang Amazon Q untuk Pengembangan Perangkat Lunak
- Menginstal, Mengkonfigurasi, & Menggunakan Amazon Q Developer dengan JetBrains IDEs (How-to)
- Menguasai seni Amazon CodeWhisperer - daftar putar YouTube
- AWS Re: invent 2020: Peningkatan kualitas kode yang berkelanjutan dengan Amazon CodeGuru
- AWS Summit ANZ 2021 - Mendorong strategi uji-pertama dengan CDK dan pengembangan berbasis pengujian

Layanan terkait:

- Pengembang Amazon Q
- CodeGuru Peninjau Amazon
- Amazon CodeGuru Profiler
- AWS Cloud9

OPS05-BP08 Gunakan beberapa lingkungan

Gunakan beberapa lingkungan untuk bereksperimen, mengembangkan, dan menguji beban kerja Anda. Gunakan tingkat kontrol berjenjang seiring lingkungan mendekati tahap produksi untuk mendapatkan keyakinan bahwa beban kerja Anda beroperasi sesuai keinginan ketika di-deploy.

Hasil yang diinginkan: Anda memiliki beberapa lingkungan yang mencerminkan kebutuhan-kebutuhan kepatuhan dan tata kelola Anda. Anda menguji dan mempromosikan kode melalui lingkungan-lingkungan yang ada di jalur Anda menuju produksi.

Anti-pola umum:

- Anda sedang melakukan pengembangan di sebuah lingkungan pengembangan bersama dan pengembang lain menimpa perubahan kode Anda.
- Kontrol keamanan terbatas di lingkungan pengembangan bersama Anda melarang Anda untuk melakukan eksperimen dengan layanan dan fitur baru.
- Anda melakukan pengujian beban pada sistem produksi Anda dan menyebabkan terjadinya pemadaman (outage) terhadap pengguna Anda.
- Kesalahan fatal yang menyebabkan hilangnya data terjadi di lingkungan produksi. Di lingkungan produksi, Anda mencoba membuat ulang kondisi yang menyebabkan data hilang tersebut sehingga Anda dapat mengidentifikasi bagaimana hal tersebut bisa terjadi dan mencegahnya agar tidak terjadi kembali. Untuk mencegah kejadian hilang data lainnya selama proses pengujian, Anda terpaksa menjadikan aplikasi tidak tersedia untuk pengguna.
- Anda mengoperasikan layanan multi-tenant dan tidak dapat mendukung permintaan lingkungan khusus yang diajukan oleh pelanggan.
- Anda mungkin tidak selalu melakukan pengujian, tetapi ketika Anda melakukannya, Anda melakukan pengujian tersebut di lingkungan produksi.
- Anda percaya bahwa dengan satu lingkungan tunggal, cakupan dampak perubahannya hanya terjadi di dalam lingkungan tersebut.

Manfaat menerapkan praktik terbaik ini: Anda dapat mendukung beberapa lingkungan pengembangan, pengujian, dan produksi secara serentak tanpa menimbulkan konflik antar pengembang atau komunitas pengguna.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Gunakan beberapa lingkungan dan sediakan lingkungan sandbox pengembang yang memiliki kontrol minimum untuk membantu eksperimen. Sediakan lingkungan pengembangan individu untuk membantu melakukan pekerjaan secara paralel, sehingga ketangkasan pengembangan akan meningkat. Implementasikan kontrol yang lebih kuat di lingkungan tersebut ketika mendekati produksi agar pengembang dapat membuat inovasi. Gunakan infrastruktur sebagai kode dan sistem

manajemen konfigurasi untuk men-deploy lingkungan yang dikonfigurasi sesuai dengan kontrol yang ada di dalam lingkungan produksi guna memastikan sistem beroperasi sesuai keinginan saat di-deploy. Saat lingkungan tidak digunakan, nonaktifkan lingkungan tersebut untuk menghindari timbulnya biaya-biaya terkait sumber daya tidak terpakai (misalnya sistem pengembangan di malam hari dan di akhir pekan). Deploy lingkungan setara produksi saat melakukan pengujian beban untuk meningkatkan hasil yang valid.

Sumber daya

Dokumen terkait:

- Penjadwal Instance aktif AWS
- Apa itu AWS CloudFormation?

OPS05-BP09 Lakukan perubahan yang sering, kecil, dan reversibel

Gunakan perubahan yang sering, kecil, dan dapat dikembalikan untuk mengurangi cakupan perubahan. Ketika digunakan bersamaan dengan sistem manajemen perubahan, sistem manajemen konfigurasi, dan sistem build serta pengiriman, perubahan yang sering, kecil, dan dapat dikembalikan dapat mengurangi cakupan dan dampak perubahan. Hal ini menghasilkan pemecahan masalah yang lebih efektif dan remediasi yang lebih cepat dengan opsi untuk membatalkan perubahan.

Anti-pola umum:

- Anda melakukan deployment versi baru aplikasi Anda setiap tiga bulan sekali dengan periode perubahan yang mengharuskan layanan inti dinonaktifkan.
- Anda sering kali membuat perubahan pada skema basis data Anda tanpa melacak perubahan dalam sistem manajemen Anda terlebih dahulu.
- Anda melakukan pembaruan secara manual di tempat, menimpa instalasi dan konfigurasi yang ada, dan tidak memiliki rencana roll-back yang jelas.

Manfaat menerapkan praktik terbaik ini: Upaya pengembangan akan menjadi lebih cepat dengan menerapkan perubahan kecil dalam frekuensi yang rapat. Ketika perubahan itu berukuran kecil, perubahan tersebut akan jauh lebih mudah diidentifikasi jika terdapat konsekuensi yang tidak diinginkan, serta lebih mudah untuk dikembalikan. Ketika perubahan dapat dikembalikan, risiko implementasi perubahan menjadi lebih kecil karena pemulihannya lebih mudah diterapkan. Proses perubahan memiliki risiko yang lebih kecil dan dampak kegagalan perubahan menjadi berkurang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Gunakan perubahan yang sering, kecil, dan dapat dikembalikan untuk mengurangi cakupan dan dampak yang ditimbulkan perubahan tersebut. Hal ini akan memudahkan Anda dalam melakukan pemecahan masalah, membantu proses remediasi yang lebih cepat, dan menyediakan opsi untuk membatalkan perubahan. Hal ini juga akan meningkatkan rasio nilai yang dapat Anda berikan ke bisnis.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP03 Gunakan sistem manajemen konfigurasi
- OPS05-BP04 Gunakan sistem manajemen build dan deployment
- OPS06-BP04 Mengotomatiskan pengujian dan rollback

Dokumen terkait:

- Menerapkan Layanan Mikro pada AWS
- Layanan Mikro - Observabilitas

OPS05- BP1 0 Mengotomatiskan integrasi dan penyebaran sepenuhnya

Otomatiskan build, deployment, dan pengujian beban kerja. Hal ini mengurangi kesalahan yang disebabkan oleh proses manual, dan mengurangi upaya untuk melakukan deployment perubahan.

Terapkan metadata menggunakan Tag Sumber Daya dan AWS Resource Groups sesuai strategi penandaan yang konsisten untuk mencapai identifikasi sumber daya Anda. Berikan tag pada sumber daya Anda untuk organisasi, akuntansi biaya, kontrol akses, dan penargetan pelaksanaan aktivitas operasi yang diotomatiskan.

Hasil yang diinginkan: Pengembang menggunakan alat untuk mengirimkan kode dan mencapai produksi. Pengembang tidak harus masuk ke AWS Management Console untuk memberikan pembaruan. Terdapat jejak audit penuh untuk perubahan dan konfigurasi, sehingga hal itu cukup untuk memenuhi kebutuhan tata kelola dan kepatuhan. Proses dapat diulang dan distandardisasi di seluruh tim. Pengembang bebas untuk memusatkan perhatian pada pengembangan dan pendorongan kode, sehingga akan meningkatkan produktivitas.

Anti-pola umum:

- Pada hari Jumat, Anda selesai menulis kode baru untuk cabang fitur Anda. Pada hari Senin, setelah menjalankan skrip pengujian kualitas kode dan setiap skrip pengujian unit, Anda mendaftarkan kode tersebut untuk rilisan terjadwal berikutnya.
- Anda ditugaskan untuk membuat kode perbaikan untuk sebuah masalah besar yang dapat memengaruhi banyak pelanggan di lingkungan produksi. Setelah menguji perbaikan tersebut, Anda melakukan commit terhadap kode Anda dan mengirimkan manajemen perubahan melalui email untuk meminta persetujuan deployment ke lingkungan produksi.
- Sebagai pengembang, Anda masuk ke AWS Management Console untuk membuat lingkungan pengembangan baru menggunakan metode dan sistem non-standar.

Manfaat menerapkan praktik terbaik ini: Dengan mengimplementasikan sistem manajemen build dan deployment otomatis, Anda dapat mengurangi kesalahan yang disebabkan proses yang diselesaikan secara manual dan mengurangi upaya yang diperlukan untuk melakukan deployment perubahan, sehingga akan membantu anggota tim Anda berkonsentrasi untuk menghadirkan nilai bisnis. Anda meningkatkan kecepatan pengiriman selama proses menuju lingkungan produksi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Anda menggunakan sistem manajemen build dan deployment untuk melacak dan mengimplementasikan perubahan, mengurangi kesalahan yang disebabkan oleh proses yang diselesaikan secara manual, dan mengurangi upaya yang diperlukan. Lakukan otomatisasi sepenuhnya terhadap pipeline integrasi dan deployment dari check-in kode hingga build, pengujian, deployment, dan validasi. Hal ini dapat mengurangi waktu tunggu, mendorong peningkatan frekuensi perubahan, mengurangi tingkat upaya, meningkatkan kecepatan masuk pasar, menghasilkan peningkatan produktivitas, dan meningkatkan keamanan kode Anda selama proses Anda menuju lingkungan produksi.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP03 Gunakan sistem manajemen konfigurasi
- OPS05-BP04 Gunakan sistem manajemen build dan deployment

Dokumen terkait:

- Apa itu AWS CodeBuild?
- Apa itu AWS CodeDeploy?

Video terkait:

- AWS re\ :Invent 2022 - Praktik terbaik AWS Well-Architected untuk di DevOps AWS

## OPS6. Bagaimana cara memitigasi risiko deployment?

Adopsi pendekatan yang memberikan umpan balik cepat atas kualitas dan mencapai pemulihan cepat dari perubahan yang tidak memiliki hasil yang tidak diinginkan. Menggunakan praktik tersebut akan memitigasi dampak masalah akibat deployment perubahan.

Praktik terbaik

- OPS06-BP01 Rencana untuk perubahan yang gagal
- OPS06-BP02 Uji penerapan
- OPS06-BP03 Menggunakan strategi penyebaran yang aman
- OPS06-BP04 Mengotomatiskan pengujian dan rollback

### OPS06-BP01 Rencana untuk perubahan yang gagal

Rencanakan untuk kembali ke keadaan yang diketahui pasti baik, atau perbaiki di lingkungan produksi jika deployment menyebabkan hasil yang tidak diinginkan. Adanya kebijakan untuk menetapkan rencana semacam ini bermanfaat bagi semua tim dalam mengembangkan strategi untuk pulih dari perubahan yang gagal. Beberapa contoh strategi adalah langkah deployment dan rollback, kebijakan perubahan, penanda fitur, pemisahan lalu lintas, dan pergeseran lalu lintas. Rilis tunggal dapat mencakup beberapa perubahan komponen yang terkait. Strategi harus memberikan kemampuan untuk bertahan atau pulih dari kegagalan perubahan komponen apa pun.

Hasil yang diinginkan: Anda telah menyiapkan sebuah rencana pemulihan yang mendetail untuk perubahan Anda apabila perubahan tersebut tidak berhasil. Selain itu, Anda juga telah mengurangi ukuran rilis untuk meminimalkan dampak-dampak potensial yang mungkin ditimbulkan terhadap komponen beban kerja lainnya. Hasilnya, Anda telah mengurangi dampak bisnis Anda dengan mempersingkat potensi waktu henti yang mungkin diakibatkan oleh kegagalan perubahan dan meningkatkan fleksibilitas serta efisiensi waktu pemulihan.

Anti-pola umum:

- Anda melakukan deployment dan aplikasi Anda menjadi tidak stabil, namun sepertinya masih ada pengguna yang aktif di sistem. Anda harus memutuskan apakah akan melakukan roll back terhadap perubahan yang akan berdampak pada pengguna aktif atau menunggu untuk melakukan roll back perubahan tersebut karena tahu bagaimana pun juga pengguna dapat terkena dampaknya.
- Setelah Anda membuat perubahan rutin, lingkungan baru Anda dapat diakses tetapi salah satu subnet Anda menjadi tidak dapat dijangkau. Anda harus memutuskan apakah akan melakukan roll back terhadap semuanya atau mencoba memperbaiki subnet yang tidak dapat diakses tersebut. Sementara Anda sedang memutuskan hal ini, subnet tersebut tetap tidak dapat dijangkau.
- Sistem Anda tidak dirancang dapat diperbarui dengan rilis-rilis yang lebih kecil. Akibatnya, Anda mengalami kesulitan dalam membatalkan perubahan massal tersebut selama deployment yang gagal.
- Anda tidak menggunakan infrastruktur sebagai kode (IaC) dan Anda melakukan pembaruan secara manual pada infrastruktur Anda sehingga mengakibatkan terjadinya konfigurasi yang tidak diinginkan. Anda tidak dapat melacak dan membatalkan perubahan manual secara efektif.
- Karena Anda belum mengukur peningkatan frekuensi deployment Anda, tim Anda kemudian mengalami kesulitan untuk mengurangi ukuran perubahan mereka dan meningkatkan rencana rollback mereka untuk setiap perubahan, yang berimbas pada risiko yang lebih besar dan tingkat kegagalan yang meningkat.
- Anda tidak mengukur total durasi pemadaman (outage) yang disebabkan oleh perubahan yang tidak berhasil. Tim Anda tidak dapat memprioritaskan dan meningkatkan proses deployment serta efektivitas rencana pemulihannya.

Manfaat membangun praktik terbaik ini: Memiliki rencana untuk pulih dari perubahan yang gagal meminimalkan waktu rata-rata untuk memulihkan (MTTR) dan mengurangi dampak bisnis Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kebijakan dan praktik yang konsisten serta terdokumentasi yang diadopsi oleh tim rilis akan memungkinkan organisasi untuk merencanakan apa yang seharusnya terjadi apabila terjadi kegagalan perubahan. Kebijakan tersebut harus memungkinkan perbaikan ke depan (fixing forward) dalam keadaan tertentu. Dalam situasi apa pun, rencana perbaikan ke depan atau rollback harus

didokumentasikan dan diuji dengan baik sebelum melakukan deployment ke lingkungan produksi langsung sehingga waktu yang diperlukan untuk mengembalikan perubahan dapat diminimalkan.

Langkah-langkah implementasi

1. Buatlah dokumentasi kebijakan yang mengharuskan tim memiliki rencana efektif untuk mengembalikan perubahan dalam periode tertentu.
   a. Kebijakan harus menentukan kapan situasi perbaikan ke depan diperbolehkan.
   b. Rencana rollback yang terdokumentasi harus dapat diakses oleh semua pihak yang terlibat.
   c. Tentukan persyaratan-persyaratan untuk rollback (misalnya, ketika ternyata ada deployment perubahan tidak sah).
2. Lakukan analisis terhadap tingkat dampak yang ditimbulkan oleh semua perubahan yang berkaitan dengan setiap komponen dari sebuah beban kerja.
   a. Buatlah perubahan-perubahan berulang memungkinkan untuk distandardisasi, dijadikan templat, dan diotorisasi di awal jika perubahan-perubahan tersebut mengikuti alur kerja yang konsisten yang memberlakukan kebijakan perubahan.
   b. Kurangi potensi dampak yang mungkin ditimbulkan oleh setiap perubahan dengan menjadikan ukuran perubahan lebih kecil sehingga waktu pemulihan yang dibutuhkan menjadi lebih singkat dan menyebabkan lebih sedikit dampak bisnis.
   c. Pastikan prosedur rollback akan mengembalikan kode ke keadaan yang pasti baik untuk menghindari terjadinya insiden, jika memungkinkan.
3. Integrasikan alat-alat dan alur kerja untuk menegakkan kebijakan Anda secara terprogram.
4. Buat agar data tentang perubahan dapat dilihat oleh para pemilik beban kerja lain untuk meningkatkan kecepatan diagnosis perubahan yang gagal yang tidak dapat dibatalkan.
   a. Ukur keberhasilan praktik ini dengan menggunakan data perubahan yang terlihat dan identifikasi setiap peningkatan iteratif yang mungkin dilakukan.
5. Gunakan alat-alat pemantauan untuk memverifikasi keberhasilan atau kegagalan sebuah deployment untuk mempercepat pengambilan keputusan saat melakukan rollback.
6. Ukur durasi pemadaman (outage) Anda selama terjadi kegagalan perubahan untuk terus meningkatkan kualitas rencana pemulihan Anda.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS06-BP04 Mengotomatiskan pengujian dan rollback

Dokumen terkait:

- AWS Builders Library | Memastikan Keamanan Rollback Selama Penerapan
- AWS Whitepaper | Ubah Manajemen di Cloud

Video terkait:

- re:Invent 2019 | Pendekatan Amazon untuk deployment ketersediaan tinggi

OPS06-BP02 Uji penerapan

Uji prosedur rilis dalam tahap praproduksi dengan menggunakan konfigurasi deployment, kontrol keamanan, langkah, dan prosedur yang sama seperti dalam tahap produksi. Lakukan validasi bahwa semua langkah yang di-deploy selesai sesuai harapan, seperti dengan memeriksa file, konfigurasi, dan layanan. Uji lebih lanjut semua perubahan dengan pengujian fungsional, integrasi, dan beban, beserta pemantauan apa pun seperti pemeriksaan kondisi. Dengan melakukan pengujian ini, Anda dapat mengidentifikasi masalah deployment lebih awal dengan peluang untuk merencanakan dan menanggulanginya sebelum produksi.

Anda dapat membuat lingkungan paralel sementara untuk menguji setiap perubahan. Lakukan otomatisasi deployment lingkungan pengujian dengan menggunakan infrastruktur sebagai kode (IaC) untuk membantu mengurangi jumlah pekerjaan yang terlibat dan memastikan stabilitas, konsistensi, dan pengiriman fitur yang lebih cepat.

Hasil yang diinginkan: Organisasi Anda mengadopsi budaya pengembangan berbasis pengujian yang mencakup pengujian deployment. Ini akan memastikan bahwa tim akan berkonsentrasi untuk menghadirkan nilai bisnis, bukan mengelola rilis. Tim terlibat sejak dini setelah identifikasi risiko deployment untuk menentukan arah mitigasi yang sesuai.

Anti-pola umum:

- Selama rilis produksi, deployment yang belum teruji sering kali akan menyebabkan masalah-masalah yang memerlukan penyelesaian dan eskalasi.

- Rilis Anda berisi infrastruktur sebagai kode (IaC) yang memperbarui sumber daya yang ada sekarang. Anda tidak yakin apakah IaC berjalan dengan sukses atau akan menyebabkan dampak pada sumber daya.
- Anda men-deploy sebuah fitur baru ke aplikasi Anda. Fitur tersebut tidak berfungsi sesuai keinginan dan masalah ini baru dapat diketahui setelah dilaporkan oleh para pengguna yang terdampak.
- Anda memperbarui sertifikat Anda. Anda secara tidak sengaja menginstal sertifikat ke komponen-komponen yang salah, yang akhirnya tidak terdeteksi dan berdampak pada pengunjung situs web karena koneksi yang aman ke situs web tidak dapat dibuat.

Manfaat menerapkan praktik terbaik ini: Pengujian ekstensif selama tahap pra-produksi dalam prosedur deployment serta perubahan-perubahan yang dimunculkannya dapat meminimalkan potensi dampak yang mungkin dapat ditimbulkan terhadap lingkungan produksi yang disebabkan oleh langkah-langkah deployment. Hal ini akan meningkatkan kepercayaan diri selama rilis produksi dan meminimalkan dukungan operasional tanpa memperlambat kecepatan penyampaian perubahan yang hendak dilakukan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Menguji proses deployment Anda sama pentingnya dengan menguji perubahan yang dihasilkan dari deployment Anda. Hal ini dapat dicapai dengan menguji langkah-langkah deployment Anda di lingkungan pra-produksi yang semaksimal mungkin mencerminkan produksi. Masalah-masalah umum, seperti langkah-langkah deployment yang tidak lengkap atau salah, atau kesalahan konfigurasi, dapat terdeteksi sebelum masuk ke lingkungan produksi. Selain itu, Anda dapat menguji langkah-langkah pemulihan Anda.

Contoh pelanggan

Sebagai bagian dari pipeline continuous integration and continuous delivery (CI/CD), AnyCompany Retail melakukan langkah-langkah yang ditentukan yang diperlukan untuk merilis pembaruan infrastruktur dan perangkat lunak bagi pelanggannya dalam lingkungan seperti produksi. Pipeline tersebut terdiri dari langkah pra-pemeriksaan untuk mendeteksi penyimpangan (mendeteksi perubahan pada sumber daya yang dilakukan di luar IaC Anda) di dalam sumber daya sebelum deployment, serta melakukan validasi terhadap tindakan-tindakan yang dilakukan IaC setelah inisiasi. Tahap ini memvalidasi langkah-langkah deployment, seperti memverifikasi bahwa file dan konfigurasi tertentu sudah siap dan layanan-layanan sudah berada dalam status berjalan serta merespons

dengan benar pemeriksaan kondisi pada host lokal sebelum didaftarkan ulang dengan penyeimbang beban. Selain itu, semua perubahan menandai sejumlah pengujian otomatis, misalnya pengujian fungsional, keamanan, regresi, integrasi, dan beban.

Langkah-langkah implementasi

1. Lakukan pemeriksaan pra-instalasi untuk mencerminkan lingkungan pra-produksi ke lingkungan produksi.
   a. Gunakan deteksi drift untuk mendeteksi kapan sumber daya telah diubah di luar. AWS CloudFormation
   b. Gunakan set perubahan untuk memvalidasi bahwa maksud pembaruan tumpukan cocok dengan tindakan yang AWS CloudFormation dilakukan saat set perubahan dimulai.
2. Ini akan memicu langkah persetujuan manual di AWS CodePipeline untuk mengotorisasi deployment ke lingkungan pra-produksi.
3. Gunakan konfigurasi penerapan seperti AWS CodeDeploy AppSpecfile untuk menentukan langkah penerapan dan validasi.
4. Jika berlaku, berintegrasi AWS CodeDeploy dengan AWS layanan lain atau berintegrasi AWS CodeDeploy dengan produk dan layanan mitra.
5. Pantau penerapan menggunakan Amazon CloudWatch, AWS CloudTrail, dan pemberitahuan SNS acara Amazon.
6. Lakukan pengujian otomatis pasca-deployment, termasuk pengujian fungsional, keamanan, regresi, integrasi, dan beban.
7. Memecahkan masalah deployment.
8. Validasi yang berhasil terhadap langkah-langkah sebelumnya seharusnya menginisiasi alur kerja persetujuan manual untuk memberikan otorisasi deployment ke produksi.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP02 Uji dan validasi perubahan

Dokumen terkait:

- AWS Perpustakaan Pembangun | Mengotomatiskan penerapan yang aman dan lepas tangan | Uji Penerapan
- AWS Whitepaper | Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan AWS
- Kisah Apollo - Mesin Deployment Amazon
- Cara menguji dan men-debug AWS CodeDeploy secara lokal sebelum Anda mengirimkan kode Anda
- Mengintegrasikan Pengujian Konektivitas Jaringan dengan Deployment Infrastruktur

Video terkait:

- re:Invent 2020 | Menguji perangkat lunak dan sistem di Amazon

Contoh terkait:

- Tutorial | Menyebarkan dan ECS layanan Amazon dengan tes validasi

OPS06-BP03 Menggunakan strategi penyebaran yang aman

Peluncuran produksi yang aman mengontrol aliran perubahan yang bermanfaat dengan tujuan untuk meminimalkan dampak yang dirasakan oleh pelanggan dari perubahan tersebut. Kontrol keselamatan menyediakan mekanisme-mekanisme inspeksi untuk memvalidasi hasil yang diinginkan dan membatasi ruang lingkup dampak yang ditimbulkan oleh cacat apa pun yang disebabkan oleh perubahan atau kegagalan deployment. Peluncuran yang aman dapat mencakup strategi seperti feature-flag, one-box, rolling (rilis canary), immutable, pemisahan lalu lintas, dan deployment blue/green.

Hasil yang diinginkan: Organisasi Anda menggunakan sebuah sistem integrasi berkelanjutan pengiriman berkelanjutan (CI/CD) yang menyediakan kemampuan-kemampuan untuk mengotomatiskan peluncuran (rollout) dengan aman. Tim diharuskan menggunakan strategi peluncuran aman yang sesuai.

Anti-pola umum:

- Anda melakukan deployment perubahan yang tidak berhasil ke seluruh lingkungan produksi secara sekaligus. Akibatnya, semua pelanggan merasakan dampaknya secara bersamaan.
- Cacat akibat deployment serentak yang dilakukan ke semua sistem memerlukan rilis darurat. Diperlukan waktu beberapa hari untuk memperbaikinya untuk semua pelanggan.

- Untuk mengelola rilis produksi diperlukan perencanaan dan partisipasi dari beberapa tim. Hal ini akan menghambat kemampuan Anda untuk melakukan pembaruan fitur bagi pelanggan Anda dalam rentang waktu yang berdekatan (frequent).
- Anda melakukan deployment yang dapat diubah dengan melakukan modifikasi terhadap sistem yang sudah ada. Setelah mengetahui bahwa perubahan yang di-deploy tidak berhasil, Anda terpaksa melakukan modifikasi terhadap sistem sekali lagi untuk memulihkan versi yang sebelumnya, dan hal ini memperpanjang waktu pemulihan Anda.

Manfaat menerapkan praktik terbaik ini: Deployment otomatis menyeimbangkan kecepatan peluncuran (roll-out) dengan menghadirkan perubahan yang bermanfaat secara konsisten kepada para pelanggan. Pembatasan dampak dapat mencegah kegagalan deployment yang mahal dan memaksimalkan kemampuan tim untuk merespons kegagalan tersebut dengan efisien.

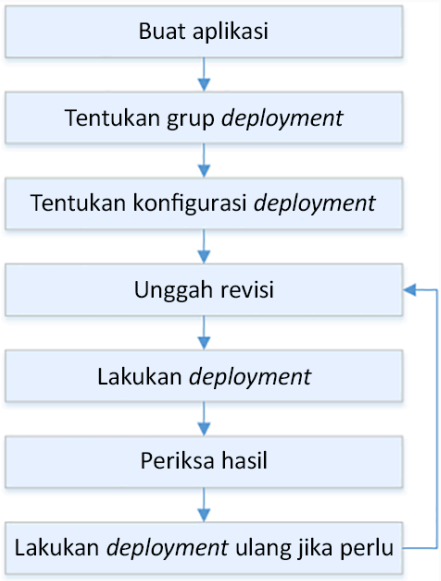
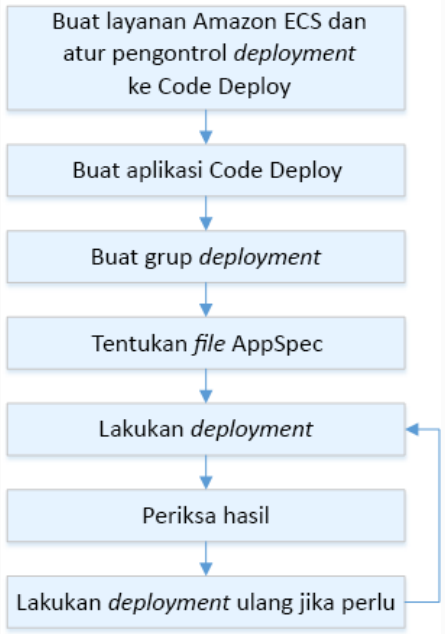
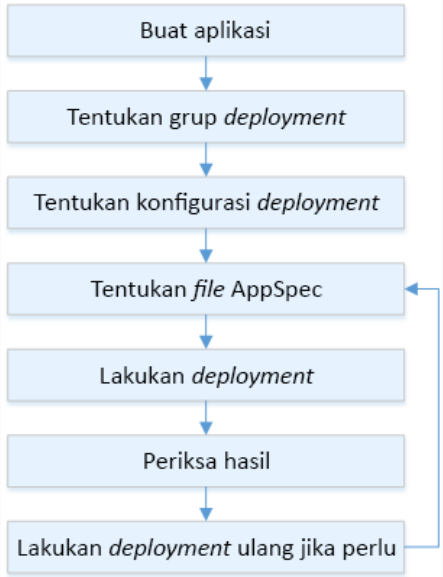
Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Kegagalan pengiriman yang terjadi secara berkelanjutan dapat menyebabkan berkurangnya ketersediaan layanan dan buruknya pengalaman pelanggan. Untuk memaksimalkan tingkat penerapan yang berhasil, terapkan kontrol keamanan dalam proses end-to-end rilis untuk meminimalkan kesalahan penerapan, dengan tujuan mencapai nol kegagalan penerapan.

Contoh pelanggan

AnyCompany Retail memiliki misi untuk mencapai penerapan downtime minimal hingga nol, yang berarti bahwa tidak ada dampak yang dapat dirasakan bagi penggunanya selama penerapan. Untuk mencapai tujuan tersebut, perusahaan telah membuat pola-pola deployment (lihat diagram alur kerja berikut), seperti deployment blue/green dan deployment bergulir (rolling). Semua tim mengadopsi satu atau beberapa pola tersebut di dalam pipeline CI/CD mereka.

| CodeDeploy alur kerja untuk Amazon EC2 | CodeDeploy alur kerja untuk Amazon ECS | CodeDeploy alur kerja untuk Lambda |
| --- | --- | --- |
|  |  |  |







Langkah-langkah implementasi

1. Gunakan alur kerja persetujuan untuk memulai urutan langkah-langkah peluncuran (roll-out) produksi setelah promosi ke produksi.
2. Gunakan sistem penyebaran otomatis seperti AWS CodeDeploy. AWS CodeDeploy Opsi penerapan mencakup penerapan di tempat untuk penerapan /Lokal dan EC2 biru/hijau untuk / LokalEC2, dan Amazon (lihat diagram alur kerja sebelumnya). AWS Lambda ECS
   a. Jika berlaku, berintegrasi AWS CodeDeploy dengan AWS layanan lain atau berintegrasi AWS CodeDeploy dengan produk dan layanan mitra.
3. Gunakan penerapan biru/hijau untuk database seperti Amazon Aurora dan Amazon. RDS
4. Pantau penerapan menggunakan Amazon CloudWatch, AWS CloudTrail, dan pemberitahuan acara Amazon Simple Notification Service SNS (Amazon).
5. Lakukan pengujian otomatis pasca-deployment, antara lain pengujian fungsional, keamanan, regresi, integrasi, dan uji beban.
6. Memecahkan masalah deployment.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP02 Uji dan validasi perubahan
- OPS05-BP09 Lakukan perubahan yang sering, kecil, dan reversibel
- OPS05- BP1 0 Mengotomatiskan integrasi dan penyebaran sepenuhnya

Dokumen terkait:

- AWS Perpustakaan Pembangun | Mengotomatiskan penerapan yang aman dan lepas tangan | Penerapan produksi
- AWS Perpustakaan Pembangun | Pipa CI/CD saya adalah kapten rilis saya | Rilis produksi otomatis yang aman
- AWS Whitepaper | Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan AWS | Metode penyebaran
- AWS CodeDeploy Panduan Pengguna
- Bekerja dengan konfigurasi penerapan di AWS CodeDeploy
- Siapkan penerapan rilis kenari API Gateway
- Jenis ECS Penerapan Amazon
- Penerapan Biru/Hijau yang Dikelola Sepenuhnya di Amazon Aurora dan Amazon RDS
- Penerapan Biru/Hijau dengan AWS Elastic Beanstalk

Video terkait:

- re:Invent 2020 | Hands-off: Mengotomatiskan pipeline pengiriman berkelanjutan di Amazon
- re:Invent 2019 | Pendekatan deployment ketersediaan tinggi Amazon

Contoh terkait:

- Coba Contoh Penerapan Biru/Hijau di AWS CodeDeploy
- Worlshop | Membentuk pipa CI/CD untuk penggunaan Lambda canary AWS CDK
- Workshop | Penerapan Biru/Hijau dan Canary untuk dan EKS ECS

- Lokakarya | Membangun Pipeline CI/CD Lintas Akun

OPS06-BP04 Mengotomatiskan pengujian dan rollback

Untuk meningkatkan kecepatan, keandalan, dan keyakinan pada proses deployment Anda, miliki strategi untuk kemampuan pengujian dan rollback otomatis di lingkungan praproduksi dan produksi. Otomatiskan pengujian saat melakukan deployment ke produksi untuk menyimulasikan interaksi manusia dan sistem yang memverifikasi perubahan yang sedang di-deploy. Otomatiskan rollback untuk kembali ke keadaan pasti baik sebelumnya dengan cepat. Rollback harus dimulai secara otomatis pada kondisi yang telah ditentukan di awal seperti ketika hasil perubahan yang Anda inginkan tidak tercapai atau ketika pengujian otomatis mengalami kegagalan. Mengotomatiskan kedua aktivitas ini dapat memperbaiki tingkat keberhasilan untuk deployment Anda, meminimalkan waktu pemulihan, dan mengurangi potensi dampak terhadap bisnis.

Hasil yang diinginkan: Strategi-strategi pengujian dan rollback otomatis Anda diintegrasikan ke dalam pipeline integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) Anda. Pemantauan Anda dapat melakukan validasi berdasarkan kriteria keberhasilan Anda dan memulai rollback otomatis setelah terjadi kegagalan. Hal ini akan meminimalkan dampak apa pun yang terjadi terhadap pelanggan dan pengguna akhir. Misalnya, ketika semua hasil pengujian telah terpenuhi, Anda meneruskan kode Anda ke lingkungan produksi tempat pengujian regresi otomatis dimulai, dengan memanfaatkan kasus-kasus pengujian yang sama. Jika hasil pengujian regresi yang didapatkan tidak sesuai dengan harapan, maka rollback otomatis akan dimulai dalam alur kerja pipeline.

Anti-pola umum:

- Sistem Anda tidak dirancang dapat diperbarui dengan rilis-rilis yang lebih kecil. Akibatnya, Anda mengalami kesulitan dalam membatalkan perubahan massal tersebut selama deployment yang gagal.
- Proses deployment Anda terdiri dari serangkaian langkah-langkah manual. Setelah melakukan deployment perubahan ke beban kerja, Anda mulai melakukan pengujian pasca-deployment. Setelah pengujian selesai, Anda menyadari bahwa beban kerja Anda tidak dapat dioperasikan dan koneksi pelanggan terputus. Kemudian Anda mulai melakukan rollback ke versi sebelumnya. Semua langkah manual ini dapat menghambat pemulihan sistem secara keseluruhan dan akan menyebabkan dampak yang berkepanjangan terhadap pelanggan Anda.
- Anda menghabiskan waktu mengembangkan kasus-kasus pengujian otomatis untuk fungsionalitas yang jarang digunakan dalam aplikasi Anda, sehingga memperkecil laba atas investasi (roi) dalam kemampuan pengujian otomatis Anda.

- Rilis Anda terdiri dari aplikasi, infrastruktur, patch, dan pembaruan konfigurasi yang tidak bergantung satu sama lain. Namun demikian, Anda memiliki satu pipeline CI/CD yang mengirimkan semua perubahan dalam satu waktu sekaligus. Kegagalan yang terjadi pada satu komponen memaksa Anda untuk mengembalikan semua perubahan, dan membuat rollback Anda menjadi kompleks dan tidak efisien.
- Tim Anda menyelesaikan tugas-tugas coding dalam sprint one dan memulai tugas sprint two, tetapi rencana Anda tidak menyertakan pengujian sampai sprint three. Akibatnya, pengujian-pengujian otomatis mengungkap cacat dari sprint one yang harus diselesaikan sebelum pengujian hasil kerja sprint two dapat dimulai dan seluruh rilis menjadi tertunda, sehingga menurunkan nilai pengujian otomatis Anda.
- Kasus pengujian regresi otomatis Anda untuk rilis produksi sudah selesai, tetapi Anda tidak memantau kondisi beban kerja. Karena Anda tidak memiliki visibilitas mengenai apakah layanan telah dimulai ulang atau belum, Anda menjadi tidak yakin apakah rollback diperlukan atau rollback sudah terjadi.

Manfaat menerapkan praktik terbaik ini: Pengujian otomatis meningkatkan transparansi proses pengujian Anda dan kemampuan Anda untuk mencakup lebih banyak fitur dalam satu periode waktu yang lebih singkat. Dengan melakukan pengujian dan validasi terhadap perubahan-perubahan dalam produksi, Anda dapat mengidentifikasi masalah dengan cepat. Peningkatan konsistensi dengan alat-alat pengujian otomatis memungkinkan deteksi kecacatan yang lebih baik. Dengan melakukan rollback otomatis ke versi sebelumnya, dampak-dampak yang ditimbulkan terhadap para pelanggan diminimalkan. Rollback otomatis pada akhirnya akan memunculkan keyakinan yang lebih tinggi pada kemampuan deployment Anda dengan mengurangi dampak bisnis. Secara keseluruhan, kemampuan ini berkurang time-to-delivery sambil memastikan kualitas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Lakukan otomatisasi terhadap pengujian lingkungan yang di-deploy untuk mengonfirmasi hasil-hasil yang diinginkan dengan lebih cepat. Lakukan otomatisasi terhadap rollback ke keadaan yang diketahui baik sebelumnya ketika hasil yang ditetapkan di awal tidak tercapai, untuk mempersingkat waktu pemulihan dan mengurangi kesalahan yang disebabkan oleh proses-proses yang dilakukan secara manual. Integrasikan alat-alat pengujian dengan alur kerja pipeline Anda untuk menguji dan meminimalkan input manual secara konsisten. Prioritaskan otomatisasi kasus pengujian, seperti kasus pengujian yang memitigasi risiko terbesar dan kasus pengujian yang harus sering diuji dengan

setiap perubahan. Selain itu, otomatiskan rollback berdasarkan kondisi tertentu yang telah ditentukan sebelumnya di awal dalam rencana pengujian Anda.

Langkah-langkah implementasi

1. Bangun sebuah siklus hidup pengujian untuk siklus hidup pengembangan Anda yang menentukan setiap tahap proses pengujian mulai dari perencanaan persyaratan hingga pengembangan kasus pengujian, konfigurasi alat, pengujian otomatis, dan penutupan kasus pengujian.
   a. Buatlah sebuah pendekatan pengujian khusus beban kerja dari strategi pengujian Anda secara keseluruhan.
   b. Pertimbangkan strategi pengujian berkelanjutan jika diperlukan di seluruh siklus hidup pengembangan.
2. Pilih alat-alat otomatis untuk pengujian dan rollback berdasarkan kebutuhan bisnis dan investasi pipeline Anda.
3. Tentukan kasus pengujian mana yang ingin Anda otomatisasi dan mana yang harus dilakukan secara manual. Anda dapat menentukannya berdasarkan prioritas nilai bisnis dari fitur yang sedang diuji. Selaraskan semua anggota tim dengan rencana ini dan pastikan pertanggungjawabannya untuk melakukan pengujian manual.
   a. Terapkan kemampuan-kemampuan pengujian otomatis ke kasus pengujian tertentu yang cocok untuk otomatisasi, seperti kasus berulang atau yang sering dijalankan, kasus yang memerlukan tugas berulang, atau kasus yang diperlukan di beberapa konfigurasi.
   b. Tentukan skrip otomatisasi pengujian serta kriteria keberhasilan di dalam alat otomatisasi sehingga otomatisasi alur kerja yang berkelanjutan dapat dimulai ketika ada kasus tertentu yang mengalami kegagalan.
   c. Tentukan kriteria kegagalan khusus untuk melakukan rollback otomatis.
4. Prioritaskan otomatisasi pengujian untuk mendorong hasil yang konsisten dengan pengembangan kasus pengujian menyeluruh di mana kompleksitas dan interaksi manusia memiliki risiko kegagalan yang lebih tinggi.
5. Integrasikan alat-alat pengujian otomatis dan rollback Anda ke dalam pipeline CI/CD Anda.
   a. Kembangkan kriteria keberhasilan yang jelas untuk perubahan Anda.
   b. Lakukan pemantauan dan pengamatan untuk mendeteksi kriteria-kriteria ini dan secara otomatis membatalkan perubahan ketika kriteria rollback tertentu terpenuhi.
6. Lakukan berbagai jenis pengujian produksi otomatis, seperti:
   a. Pengujian A/B untuk menunjukkan hasil yang dibandingkan dengan versi saat ini antara dua kelompok pengujian pengguna.

   b. Pengujian canary yang memungkinkan Anda untuk meluncurkan perubahan Anda pada subset pengguna sebelum merilisnya ke semua pengguna.
   c. Pengujian penandaan fitur (feature-flag testing) yang memungkinkan satu per satu fitur dari versi baru untuk ditandai atau dihapus tandanya dari luar aplikasi sehingga setiap fitur baru dapat divalidasi satu per satu.
   d. Pengujian regresi untuk memverifikasi fungsionalitas baru dengan komponen-komponen yang saling terkait.
7. Lakukan pemantauan pada aspek operasional aplikasi, transaksi, dan interaksi dengan aplikasi dan komponen lain. Kembangkan laporan untuk menunjukkan keberhasilan perubahan berdasarkan beban kerja sehingga Anda dapat mengidentifikasi bagian otomatisasi dan alur kerja apa yang dapat dioptimalkan lebih lanjut.
   a. Kembangkan laporan hasil pengujian yang membantu Anda mengambil keputusan cepat terkait apakah prosedur rollback perlu diinvokasi.
   b. Terapkan sebuah strategi yang dapat memungkinkan rollback otomatis berdasarkan kondisi kegagalan yang telah ditentukan di awal yang dihasilkan dari satu atau beberapa metode pengujian Anda.
8. Kembangkan kasus pengujian otomatis untuk memungkinkan penggunaan ulang di seluruh perubahan berulang di masa mendatang.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS06-BP01 Rencana untuk perubahan yang gagal
- OPS06-BP02 Uji penerapan

Dokumen terkait:

- AWS Builders Library | Memastikan keamanan rollback selama penerapan
- Menerapkan ulang dan mengembalikan penerapan dengan AWS CodeDeploy
- 8 praktik terbaik saat mengotomatiskan penerapan Anda dengan AWS CloudFormation

Contoh terkait:

- Pengujian UI tanpa server menggunakan Selenium,, AWS Lambda, dan Alat AWS Fargate Pengembang AWS

Video terkait:

- re:Invent 2020 | Hands-off: Mengotomatiskan pipeline pengiriman berkelanjutan di Amazon
- re:Invent 2019 | Pendekatan deployment ketersediaan tinggi Amazon

## OPS7. Bagaimana cara mengetahui bahwa Anda siap untuk mendukung beban kerja?

Evaluasi kesiapan operasional beban kerja, proses, dan prosedur, serta personel Anda untuk memahami risiko operasional terkait beban kerja Anda.

Praktik terbaik

- OPS07-BP01 Memastikan kemampuan personel
- OPS07-BP02 Memastikan tinjauan kesiapan operasional yang konsisten
- OPS07-BP03 Gunakan runbook untuk melakukan prosedur
- OPS07-BP04 Gunakan pedoman untuk menyelidiki masalah
- OPS07-BP05 Membuat keputusan berdasarkan informasi untuk menyebarkan sistem dan perubahan
- OPS07-BP06 Buat rencana dukungan untuk beban kerja produksi

### OPS07-BP01 Memastikan kemampuan personel

Miliki mekanisme untuk memvalidasi bahwa Anda memiliki jumlah personel terlatih yang sesuai untuk mendukung beban kerja. Mereka harus diberi pelatihan tentang platform dan layanan yang membentuk beban kerja Anda. Berikan kepada mereka pengetahuan yang diperlukan untuk mengoperasikan beban kerja. Anda harus memiliki cukup banyak personel terlatih untuk mendukung pengoperasian normal beban kerja dan menyelesaikan masalah-masalah terkait insiden yang terjadi. Anda harus memiliki cukup banyak personel sehingga Anda dapat melakukan rotasi untuk personel yang siap tugas mendadak dan personel yang liburan guna menghindari lelah ekstrim pada personel.

Hasil yang diinginkan:

- Ada cukup banyak personel terlatih untuk mendukung beban kerja pada saat beban kerja tersedia.

- Anda memberikan pelatihan tentang perangkat lunak dan layanan yang membentuk beban kerja Anda kepada personel.

Anti-pola umum:

- Melakukan deployment beban kerja tanpa anggota tim yang terlatih untuk mengoperasikan platform dan layanan yang digunakan.
- Tidak memiliki cukup banyak personel untuk mendukung pelaksanaan rotasi personel yang siap tugas mendadak atau personel yang sedang libur.

Manfaat menjalankan praktik terbaik ini:

- Memiliki anggota tim yang terampil membantu dukungan yang efektif untuk beban kerja.
- Dengan cukup banyak anggota tim, Anda dapat mendukung beban kerja dan pelaksanaan rotasi personel yang siap tugas mendadak sekaligus mengurangi risiko personel yang terlalu lelah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pastikan bahwa terdapat personel yang terlatih dengan memadai untuk mendukung beban kerja. Pastikan Anda memiliki jumlah anggota tim yang cukup untuk menangani aktivitas-aktivitas operasional dalam kondisi normal, termasuk pelaksanaan rotasi personel yang siap bertugas mendadak.

Contoh pelanggan

AnyCompany Retail memastikan bahwa tim yang mendukung beban kerja dikelola dan dilatih dengan baik. Mereka memiliki cukup banyak rekayasawan untuk mendukung pelaksanaan rotasi personel yang siap tugas mendadak. Personel mendapatkan pelatihan tentang perangkat lunak dan platform yang merupakan dasar pembangunan beban kerja dan mereka didorong untuk mendapatkan sertifikasi. Ada cukup banyak personel sehingga orang dapat mengambil cuti sambil tetap ada dukungan untuk beban kerja dan rotasi personel yang siap tugas mendadak.

Langkah-langkah implementasi

1. Tetapkan jumlah personel yang memadai untuk mengoperasikan dan mendukung beban kerja Anda, termasuk untuk tugas-tugas mendadak.

2. Latih personel Anda tentang perangkat lunak dan platform yang membentuk beban kerja Anda.
   a. AWS Pelatihan dan Sertifikasi memiliki perpustakaan kursus tentang AWS. Kursus-kursus ini disediakan gratis dan berbayar, baik secara online maupun tatap muka.
   b. AWS menyelenggarakan acara dan webinar tempat Anda belajar dari AWS para ahli.
3. Lakukan evaluasi terhadap ukuran dan keterampilan tim secara teratur seiring perubahan kondisi pengoperasian dan beban kerja. Sesuaikan ukuran dan keterampilan tim agar memenuhi persyaratan-persyaratan operasional.

Tingkat upaya untuk rencana implementasi: Tinggi. Mempekerjakan dan melatih tim untuk mendukung beban kerja mengharuskan Anda melakukan upaya yang cukup besar, tetapi hal itu akan memberikan manfaat besar dalam jangka panjang.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP04 Melakukan manajemen pengetahuan - Anggota tim harus memiliki informasi yang diperlukan untuk mengoperasikan dan mendukung beban kerja. Manajemen pengetahuan merupakan kunci untuk menyediakan informasi tersebut.

Dokumen terkait:

- AWS Acara dan Webinar
- AWS Pelatihan dan Sertifikasi

OPS07-BP02 Memastikan tinjauan kesiapan operasional yang konsisten

Gunakan Ulasan Kesiapan Operasional (ORRs) untuk memvalidasi bahwa Anda dapat mengoperasikan beban kerja Anda. ORRadalah mekanisme yang dikembangkan di Amazon untuk memvalidasi bahwa tim dapat mengoperasikan beban kerja mereka dengan aman. An ORR adalah proses peninjauan dan inspeksi menggunakan daftar persyaratan. An ORR adalah pengalaman swalayan yang digunakan tim untuk mengesahkan beban kerja mereka. ORRstermasuk praktik terbaik dari pelajaran yang dipetik dari tahun-tahun membangun perangkat lunak kami.

ORRDaftar periksa terdiri dari rekomendasi arsitektur, proses operasional, manajemen acara, dan kualitas rilis. Proses Koreksi Kesalahan (CoE) kami merupakan pendorong utama item-item ini. Analisis pasca-insiden Anda sendiri harus mendorong evolusi Anda sendiriORR. An ORR tidak hanya

tentang mengikuti praktik terbaik tetapi mencegah terulangnya peristiwa yang pernah Anda lihat sebelumnya. Terakhir, persyaratan keamanan, tata kelola, dan kepatuhan juga dapat dimasukkan dalam persyaratan. ORR

Jalankan ORRs sebelum beban kerja diluncurkan ke ketersediaan umum dan kemudian di seluruh siklus hidup pengembangan perangkat lunak. Menjalankan ORR sebelum peluncuran meningkatkan kemampuan Anda untuk mengoperasikan beban kerja dengan aman. Jalankan kembali beban kerja Anda ORR secara berkala untuk menangkap penyimpangan dari praktik terbaik. Anda dapat memiliki ORR daftar periksa untuk peluncuran layanan baru dan ORRs untuk tinjauan berkala. Ini membantu Anda untuk tetap up to date dengan praktik terbaik yang muncul dan menggabungkan pelajaran yang didapatkan dari analisis pasca-insiden. Saat penggunaan cloud Anda matang, Anda dapat membangun ORR persyaratan ke dalam arsitektur Anda sebagai default.

Hasil yang diinginkan: Anda memiliki ORR daftar periksa dengan praktik terbaik untuk organisasi Anda. ORRsdilakukan sebelum peluncuran beban kerja. ORRsdijalankan secara berkala selama siklus hidup beban kerja.

Anti-pola umum:

- Anda meluncurkan beban kerja tanpa mengetahui apakah Anda dapat mengoperasikannya atau tidak.
- Persyaratan pengelolaan dan keamanan tidak diikutsertakan ketika menjamin peluncuran beban kerja.
- Beban kerja tidak dievaluasi kembali secara berkala.
- Beban kerja diluncurkan tanpa menerapkan prosedur-prosedur yang diperlukan.
- Anda melihat berulangnya kegagalan yang disebabkan akar masalah yang sama di beberapa beban kerja.

Manfaat menjalankan praktik terbaik ini:

- Beban kerja Anda mencakup praktik terbaik dalam hal arsitektur, proses, dan manajemen.
- Pelajaran yang dipetik dimasukkan ke dalam ORR proses Anda.
- Prosedur yang diperlukan tersedia ketika beban kerja diluncurkan.
- ORRsdijalankan di seluruh siklus hidup perangkat lunak beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

An ORR adalah dua hal: proses dan daftar periksa. ORRProses Anda harus diadopsi oleh organisasi Anda dan didukung oleh sponsor eksekutif. Minimal, ORRs harus dilakukan sebelum beban kerja diluncurkan ke ketersediaan umum. Jalankan ORR seluruh siklus hidup pengembangan perangkat lunak untuk tetap up to date dengan praktik terbaik atau persyaratan baru. ORRDaftar periksa harus mencakup item konfigurasi, persyaratan keamanan dan tata kelola, dan praktik terbaik dari organisasi Anda. Seiring waktu, Anda dapat menggunakan layanan, seperti AWS Config, AWS Security Hub, dan AWS Control Tower Pagar Pembatas, untuk membangun praktik terbaik dari pagar pembatas ORR hingga deteksi otomatis praktik terbaik.

Contoh pelanggan

Setelah beberapa insiden produksi, AnyCompany Retail memutuskan untuk menerapkan suatu ORR proses. Mereka membangun daftar periksa yang terdiri dari praktik terbaik, persyaratan pengelolaan dan kepatuhan, serta pelajaran yang didapatkan dari pemadaman (outage). Beban kerja baru dilakukan ORRs sebelum diluncurkan. Setiap beban kerja dilakukan setiap tahun ORR dengan subset praktik terbaik untuk menggabungkan praktik dan persyaratan terbaik baru yang ditambahkan ke daftar periksa. ORR Seiring waktu, AnyCompany Retail digunakan AWS Configuntuk mendeteksi beberapa praktik terbaik, mempercepat ORR proses.

Langkah-langkah implementasi

Untuk mempelajari selengkapnyaORRs, baca whitepaper Ulasan Kesiapan Operasional (ORR). Ini memberikan informasi rinci tentang sejarah ORR proses, bagaimana membangun ORR praktik Anda sendiri, dan bagaimana mengembangkan ORR daftar periksa Anda. Langkah-langkah berikut ini merupakan versi singkat dari dokumen tersebut. Untuk pemahaman mendalam tentang apa itu ORRs dan bagaimana membangunnya sendiri, kami sarankan membaca whitepaper itu.

1. Kumpulkan para pemangku kepentingan utama, termasuk para perwakilan dari bagian keamanan, operasi, dan pengembangan.
2. Minta setiap pemangku kepentingan untuk menyediakan setidaknya satu persyaratan. Untuk iterasi pertama, coba batasi jumlah item menjadi tiga puluh atau kurang.
   - Lampiran B: Contoh ORR pertanyaan dari whitepaper Ulasan Kesiapan Operasional (ORR) berisi contoh pertanyaan yang dapat Anda gunakan untuk memulai.
3. Kumpulkan persyaratan Anda ke dalam lembar kerja.
   - Anda dapat menggunakan lensa khusus AWS Well-Architected Tooluntuk mengembangkan ORR dan membagikannya di seluruh akun dan AWS Organisasi Anda.

4. Identifikasi satu beban kerja untuk melakukan ORR on. Idealnya adalah beban kerja sebelum peluncuran atau beban kerja internal.
5. Jalankan ORR daftar periksa dan catat setiap penemuan yang dibuat. Penemuannya mungkin akan buruk jika terdapat mitigasi. Untuk penemuan yang tidak memiliki mitigasi, tambahkan beban kerja ke backlog item Anda dan implementasikan sebelum peluncuran.
6. Terus tambahkan praktik dan persyaratan terbaik ke ORR daftar periksa Anda dari waktu ke waktu.

AWS Support Pelanggan dengan Enterprise Support dapat meminta Lokakarya Tinjauan Kesiapan Operasional dari Manajer Akun Teknis mereka. Lokakarya ini adalah sesi kerja mundur interaktif untuk mengembangkan ORR daftar periksa Anda sendiri.

Tingkat upaya untuk rencana implementasi: Tinggi. Mengadopsi ORR praktik di organisasi Anda membutuhkan sponsor eksekutif dan pembelian pemangku kepentingan. Buat dan perbarui daftar periksa dengan masukan dari seluruh organisasi Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP03 Mengevaluasi persyaratan tata kelola— Persyaratan tata kelola sangat cocok untuk ORR daftar periksa.
- OPS01-BP04 Mengevaluasi persyaratan kepatuhan— Persyaratan kepatuhan terkadang disertakan dalam ORR daftar periksa. Terkadang persyaratan kepatuhan adalah proses yang terpisah.
- OPS03-BP07 Tim sumber daya dengan tepat— Kemampuan tim adalah kandidat yang baik untuk suatu ORR persyaratan.
- OPS06-BP01 Rencana untuk perubahan yang gagal – Rencana rollback atau rollforward harus dibuat sebelum Anda meluncurkan beban kerja Anda.
- OPS07-BP01 Memastikan kemampuan personel – Untuk mendukung beban kerja, Anda harus memiliki personel yang diperlukan.
- SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan kontrol — Tujuan pengendalian keamanan membuat persyaratan yang sangat baik. ORR
- REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data — Rencana pemulihan bencana adalah persyaratan yang baik. ORR

- COST02-BP01 Kembangkan kebijakan berdasarkan persyaratan organisasi Anda — Kebijakan manajemen biaya baik untuk dimasukkan dalam daftar periksa Anda. ORR

Dokumen terkait:

- AWS Control Tower - Pagar pembatas di AWS Control Tower
- AWS Well-Architected Tool - Lensa Kustom
- Templat Peninjauan Kesiapan Operasional oleh Adrian Hornsby
- Ulasan Kesiapan Operasional (ORR) Whitepaper

Video terkait:

- AWS Support s You | Membangun Tinjauan Kesiapan Operasional yang Efektif () ORR

Contoh terkait:

- Contoh Tinjauan Kesiapan Operasional (ORR) Lensa

Layanan terkait:

- AWS Config
- AWS Control Tower
- AWS Security Hub
- AWS Well-Architected Tool

OPS07-BP03 Gunakan runbook untuk melakukan prosedur

Runbook adalah sebuah proses terdokumentasi untuk meraih hasil tertentu. Runbook terdiri dari serangkaian langkah yang diikuti seseorang untuk menyelesaikan sesuatu. Runbook telah digunakan dalam operasi sejak masa-masa awal industri penerbangan. Dalam operasi cloud, kita menggunakan runbook untuk mengurangi risiko dan meraih hasil-hasil yang diinginkan. Dalam bentuk paling sederhananya, runbook adalah daftar periksa untuk menyelesaikan tugas.

Runbook adalah bagian penting dari operasi beban kerja Anda. Mulai dari pelaksanaan orientasi untuk anggota tim baru hingga melakukan deployment rilis besar, runbook adalah proses terkodifikasi yang dapat memberikan hasil-hasil yang konsisten, siapa pun yang menggunakannya. Runbook

harus dipublikasikan di lokasi sentral dan diperbarui seiring berkembangnya proses karena memperbarui runbook adalah komponen utama dari proses manajemen perubahan. Runbook juga harus menyertakan panduan tentang cara menangani kesalahan, alat, izin, pengecualian, dan eskalasi jika terjadi masalah.

Saat organisasi Anda matang, mulailah mengotomatiskan runbook. Mulailah dengan runbook yang singkat dan sering kali digunakan. Gunakan bahasa skrip untuk mengotomatiskan langkah-langkah atau mempermudah pelaksanaan langkah-langkah. Seiring Anda mengotomatiskan beberapa runbook pertama, Anda harus mendedikasikan waktu untuk mengotomatiskan runbook yang lebih kompleks. Seiring waktu, sebagian besar runbook Anda harus diotomatiskan dalam cara tertentu.

Hasil yang diinginkan: Tim Anda memiliki kumpulan step-by-step panduan untuk melakukan tugas beban kerja. Runbook berisi hasil yang diinginkan, alat dan izin yang diperlukan, serta petunjuk untuk penanganan kesalahan. Runbook disimpan di sebuah lokasi sentral (sistem kontrol versi) dan sering diperbarui. Misalnya, runbook Anda menyediakan kemampuan bagi tim Anda untuk memantau, berkomunikasi, dan merespons AWS Health peristiwa untuk akun penting selama alarm aplikasi, masalah operasional, dan peristiwa siklus hidup yang direncanakan.

Anti-pola umum:

- Mengandalkan ingatan untuk menyelesaikan setiap langkah dari suatu proses.
- Menerapkan perubahan secara manual tanpa menggunakan daftar periksa.
- Anggota tim yang berbeda-beda melakukan proses yang sama, tetapi dengan langkah atau hasil yang berbeda.
- Membiarkan runbook tidak selaras dengan perubahan sistem dan otomatisasi.

Manfaat menjalankan praktik terbaik ini:

- Mengurangi tingkat kesalahan untuk tugas-tugas yang dilakukan manual.
- Operasi dilakukan secara konsisten.
- Anggota tim baru dapat mulai melakukan tugas dengan lebih cepat.
- Runbook dapat diotomatiskan untuk mengurangi upaya yang diperlukan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Runbook dapat memiliki beberapa bentuk, bergantung pada tingkat kematangan organisasi Anda. Minimal, mereka harus terdiri dari dokumen step-by-step teks. Hasil yang diinginkan harus ditunjukkan dengan jelas. Buatlah dokumentasi yang jelas mengenai izin atau alat khusus yang diperlukan. Berikan panduan mendetail tentang cara menangani kesalahan dan cara melakukan eskalasi jika terjadi kesalahan. Cantumkan pemilik runbook dan publikasikan di sebuah lokasi sentral. Setelah runbook Anda didokumentasikan, kemudian validasi dengan meminta orang lain di tim Anda untuk menjalankannya. Seiring berkembangnya prosedur, perbarui runbook Anda sesuai dengan proses manajemen perubahan Anda.

Runbook teks Anda harus diotomatiskan seiring semakin matangnya organisasi Anda. Dengan menggunakan layanan-layanan seperti AWS Systems Manager Automation, Anda dapat mentransformasikan teks biasa menjadi otomatisasi yang dapat dijalankan dengan beban kerja Anda. Otomatisasi ini dapat dijalankan sebagai respons terhadap peristiwa, mengurangi beban operasional untuk mempertahankan beban kerja Anda. AWS Systems Manager Automation juga menyediakan pengalaman desain visual kode rendah untuk membuat runbook otomatisasi dengan lebih mudah.

Contoh pelanggan

AnyCompany Ritel harus melakukan pembaruan skema basis data selama penerapan perangkat lunak. Tim Operasi Cloud bekerja sama dengan Tim Administrasi Basis Data untuk membuat sebuah runbook guna menerapkan perubahan ini secara manual. Runbook ini mencantumkan setiap langkah yang ada dalam prosesnya dalam bentuk daftar periksa. Runbook ini berisi sebuah bagian yang menjelaskan cara menangani kesalahan saat terjadi kesalahan. Mereka memublikasikan runbook di wiki internal mereka bersama dengan runbook mereka yang lain. Tim Operasi Cloud berencana untuk mengotomatiskan runbook dalam sprint mendatang.

Langkah-langkah implementasi

Jika Anda belum memiliki repositori dokumen, repositori kontrol versi bisa menjadi tempat yang tepat untuk mulai membangun pustaka runbook Anda. Anda dapat membangun runbook Anda dengan menggunakan Markdown. Kami telah menyediakan contoh templat runbook yang dapat Anda gunakan untuk mulai membangun runbook.

```
# Runbook Title
## Runbook Info
| Runbook ID | Description | Tools Used | Special Permissions | Runbook Author | Last
 Updated | Escalation POC |
|-------|-------|-------|-------|-------|-------|-------|
```

```
| RUN001 | What is this runbook for? What is the desired outcome? | Tools | Permissions
 | Your Name | 2022-09-21 | Escalation Name |
## Steps
1. Step one
2. Step two
```

1. Jika Anda belum memiliki repositori atau wiki dokumentasi, buatlah repositori kontrol versi baru di sistem kontrol versi Anda.
2. Identifikasi proses yang tidak memiliki runbook. Proses yang ideal adalah proses yang dilakukan secara semi-reguler, sedikit jumlah langkahnya, dan memiliki kegagalan yang dampaknya rendah.
3. Di repositori dokumen Anda, buat draf dokumen Markdown baru dengan menggunakan templat tersebut. Isi Judul Runbook dan bidang-bidang yang wajib diisi di bawah Info Runbook.
4. Dimulai dengan langkah pertama, isi bagian Langkah-langkah dalam runbook.
5. Berikan runbook kepada anggota tim. Minta mereka menggunakan runbook ini untuk memvalidasi langkah-langkahnya. Jika ada sesuatu yang belum dimasukkan atau memerlukan kejelasan, perbarui runbook ini.
6. Publikasikan runbook ini ke bagian penyimpanan dokumentasi internal Anda. Setelah dipublikasikan, beri tahu tim Anda dan para pemangku kepentingan lainnya.
7. Seiring waktu, Anda akan membangun sebuah pustaka runbook. Saat pustaka tersebut bertambah besar, mulailah bekerja untuk mengotomatiskan runbook.

Tingkat upaya untuk rencana implementasi: Rendah. Standar minimum untuk runbook adalah panduan step-by-step teks. Mengotomatiskan runbook dapat meningkatkan upaya penerapan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik
- OPS07-BP04 Gunakan pedoman untuk menyelidiki masalah
- OPS10-BP01 Gunakan proses untuk manajemen peristiwa, insiden, dan masalah
- OPS10-BP02 Memiliki proses per peringatan
- OPS11-BP04 Melakukan manajemen pengetahuan

Dokumen terkait:

- Kerangka Kerja AWS Well-Architected: Konsep: Pengembangan runbook
- Mencapai Keunggulan Operasional menggunakan playbook dan runbook otomatis
- AWS Systems Manager: Bekerja dengan runbook
- Buku pedoman migrasi untuk migrasi AWS besar - Tugas 4: Meningkatkan runbook migrasi Anda
- Gunakan runbook AWS System Manager Automation untuk menyelesaikan tugas-tugas operasional

Video terkait:

- AWS Re: invent 2019: DIY panduan untuk runbook, laporan insiden, dan respons insiden
- Cara mengotomatiskan Operasi TI di AWS | Amazon Web Services
- Integrasikan Skrip ke dalam AWS Systems Manager

Contoh terkait:

- Lab Well-Architected: Melakukan otomatisasi operasi dengan Playbook dan Runbook
- AWS Posting Blog: Membangun Praktik Otomasi Cloud untuk Keunggulan Operasional: Praktik Terbaik dari AWS Managed Services
- AWS Systems Manager: Panduan otomatisasi
- AWS Systems Manager: Kembalikan volume root dari runbook snapshot terbaru
- Membangun runbook respons AWS insiden menggunakan notebook Jupyter dan Danau CloudTrail
- Gitlab - Runbook
- Rubix – Pustaka Python untuk membuat runbook di Notebook Jupyter
- Menggunakan Document Builder untuk membuat runbook kustom

Layanan terkait:

- AWS Otomatisasi Systems Manager

OPS07-BP04 Gunakan pedoman untuk menyelidiki masalah

Playbook adalah step-by-step panduan yang digunakan untuk menyelidiki suatu insiden. Ketika terjadi sebuah insiden, playbook digunakan untuk menyelidiki, membuat cakupan dampak, dan mengidentifikasi akar masalah penyebabnya. Playbook digunakan untuk berbagai skenario, dari

deployment yang gagal hingga insiden keamanan. Dalam banyak kasus, playbook mengidentifikasi akar masalah yang dimitigasi dengan menggunakan runbook. Playbook adalah komponen pokok dalam rencana respons insiden organisasi Anda.

Playbook yang baik memiliki sejumlah fitur utama. Playbook memberikan panduan secara mendetail bagi pengguna, dalam proses penemuan. Dengan berpikir secara menyeluruh, langkah apa saja yang sebaiknya diikuti seseorang untuk mendiagnosis sebuah insiden? Tetapkan secara jelas di dalam playbook apakah alat-alat khusus atau izin yang lebih tinggi diperlukan di dalam playbook. Membuat sebuah rencana komunikasi untuk memberikan informasi terbaru kepada para pemangku kepentingan mengenai status penyelidikan adalah komponen utama. Dalam situasi ketika akar penyebab masalah tidak dapat diidentifikasi, playbook harus memiliki rencana eskalasi. Jika akar penyebab masalah sudah diidentifikasi, playbook harus mengarah ke sebuah runbook yang menjelaskan cara menyelesaikannya. Playbook harus disimpan secara terpusat dan dipelihara secara rutin. Jika playbook digunakan untuk pemberitahuan khusus, bekali tim Anda dengan penunjuk ke playbook yang ada di dalam pemberitahuan tersebut.

Otomatisasi playbook Anda seiring dengan kematangan organisasi. Mulailah dengan playbook yang mencakup insiden-insiden berisiko rendah. Gunakan penulisan skrip untuk mengotomatiskan langkah-langkah penemuan. Pastikan Anda memiliki runbook pendamping untuk memitigasi akar masalah umum.

Hasil yang diinginkan: Organisasi Anda memiliki playbook untuk insiden umum. Playbook disimpan di lokasi terpusat dan tersedia untuk anggota tim Anda. Playbook harus sering diperbarui. Runbook pendamping dibuat untuk akar masalah apa pun yang diketahui.

Anti-pola umum:

- Tidak ada cara standar untuk menyelidiki sebuah insiden.
- Anggota tim mengandalkan memori otot atau pengetahuan kelembagaan untuk memecahkan masalah kegagalan deployment.
- Anggota tim baru mempelajari cara menyelidiki permasalahan melalui coba-coba (trial and error).
- Praktik terbaik untuk menyelidiki permasalahan tidak dibagikan ke seluruh tim.

Manfaat menjalankan praktik terbaik ini:

- Playbook meningkatkan upaya Anda untuk memitigasi insiden.
- Anggota tim yang berbeda-beda dapat menggunakan playbook yang sama untuk mengidentifikasi akar penyebab masalah secara konsisten.

- Setelah akar penyebab masalah diketahui, kemudian bisa dikembangkan runbook untuk masalah tersebut, sehingga dapat mempercepat waktu pemulihan.
- Playbook akan membantu anggota tim untuk mulai berkontribusi lebih cepat.
- Tim dapat menskalakan proses mereka dengan playbook yang dapat diulangi langkah-langkahnya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Cara Anda membangun dan menggunakan playbook bergantung pada kematangan organisasi Anda. Jika Anda baru mengenal cloud, buatlah playbook dalam bentuk teks di dalam repositori dokumen pusat. Seiring dengan kematangan organisasi, playbook bisa dibuat menjadi semi-otomatis dengan bahasa skrip seperti Python. Skrip-skrip ini dapat dijalankan di dalam notebook Jupyter untuk mempercepat penemuan. Organisasi tingkat lanjut memiliki playbook yang sepenuhnya otomatis untuk permasalahan-permasalahan umum yang diperbaiki secara otomatis dengan runbook.

Mulailah membangun playbook Anda dengan mengidentifikasi insiden-insiden umum yang terjadi pada beban kerja Anda. Untuk mengawali, pilihlah playbook untuk insiden-insiden dengan risiko rendah dan dengan akar penyebab masalah yang telah dipersempit menjadi beberapa permasalahan. Setelah Anda memiliki playbook untuk skenario yang lebih sederhana, beralihlah ke skenario yang memiliki risiko lebih tinggi atau skenario dengan akar penyebab masalah yang tidak diketahui dengan baik.

Playbook teks Anda harus diotomatiskan seiring dengan kematangan organisasi Anda. Menggunakan layanan seperti AWS Systems Manager Automation, teks datar dapat ditransformasi menjadi otomatisasi. Otomatisasi ini dapat dijalankan terhadap beban kerja untuk mempercepat penyelidikan. Otomatisasi ini dapat diaktifkan untuk merespons peristiwa, sehingga akan mengurangi rata-rata waktu untuk menemukan dan menyelesaikan insiden.

Pelanggan dapat menggunakan AWS Systems Manager Incident Manager untuk menanggapi insiden. Layanan ini menyediakan satu antarmuka untuk memeriksa insiden, memberikan informasi kepada para pemangku kepentingan saat proses penemuan dan mitigasi, dan berkolaborasi selama berlangsungnya insiden. Menggunakan Automasi AWS Systems Manager untuk mempercepat deteksi dan pemulihan.

Contoh pelanggan

Insiden produksi berdampak pada AnyCompany Retail. Rekayasawan yang siap dipanggil kapan saja (on-call) menggunakan playbook untuk menyelidiki permasalahan. Seiring mereka mengikuti

langkah-langkahnya, mereka terus memutakhirkan para pemangku kepentingan utama yang diidentifikasi di dalam playbook. Rekayasawan mengidentifikasi akar penyebab masalah sebagai kondisi pacu di dalam layanan backend. Menggunakan runbook, insinyur meluncurkan kembali layanan, membawa AnyCompany Retail kembali online.

Langkah-langkah implementasi

Jika Anda belum memiliki repositori dokumen, kami menyarankan Anda untuk membuat repositori kontrol versi untuk pustaka playbook Anda. Anda dapat membangun playbook Anda dengan menggunakan Markdown, yang kompatibel dengan sebagian besar sistem otomatisasi playbook. Jika Anda memulai dari nol, gunakan contoh templat playbook berikut ini.

```
# Playbook Title
## Playbook Info
| Playbook ID | Description | Tools Used | Special Permissions | Playbook Author | Last
 Updated | Escalation POC | Stakeholders | Communication Plan |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| RUN001 | What is this playbook for? What incident is it used for? | Tools |
 Permissions | Your Name | 2022-09-21 | Escalation Name | Stakeholder Name | How will
 updates be communicated during the investigation? |
## Steps
1. Step one
2. Step two
```

1. Jika Anda belum memiliki repositori dokumen atau wiki, buatlah repositori kontrol versi baru untuk playbook Anda di sistem kontrol versi Anda.
2. Identifikasi permasalahan umum yang memerlukan penyelidikan. Ini sebaiknya adalah skenario dengan akar penyebab masalah yang dibatasi ke beberapa permasalahan dan penyelesaian yang memiliki risiko rendah.
3. Menggunakan templat Markdown, isilah bagian Nama Playbook dan bidang di bawah Info Playbook.
4. Lengkapi langkah-langkah pemecahan masalah. Sampaikan dengan sejelas mungkin tindakan-tindakan yang akan dilakukan atau area apa saja yang harus Anda selidiki.
5. Berikan playbook tersebut kepada anggota tim dan minta mereka mempelajari dan memvalidasinya. Jika terdapat hal yang terlewatkan atau tidak jelas, lakukan pembaruan playbook.
6. Terbitkan playbook di dalam repositori dokumen Anda dan informasikan kepada tim dan pemangku kepentingan.

7. Pustaka playbook ini akan tumbuh seiring dengan semakin banyaknya playbook yang Anda tambahkan. Setelah Anda memiliki beberapa buku pedoman, mulailah mengotomatiskannya menggunakan alat seperti AWS Systems Manager Automations untuk menjaga otomatisasi dan buku pedoman tetap sinkron.

Tingkat upaya untuk rencana implementasi: Rendah. Playbook Anda harus berupa dokumen teks yang disimpan di sebuah lokasi terpusat. Organisasi yang lebih matang akan beralih ke playbook otomatis.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik
- OPS07-BP03 Gunakan runbook untuk melakukan prosedur
- OPS10-BP01 Gunakan proses untuk manajemen peristiwa, insiden, dan masalah
- OPS10-BP02 Memiliki proses per peringatan
- OPS11-BP04 Melakukan manajemen pengetahuan

Dokumen terkait:

- Kerangka Kerja AWS Well-Architected: Konsep: Pengembangan Playbook
- Mencapai Keunggulan Operasional menggunakan playbook dan runbook otomatis
- AWS Systems Manager: Bekerja dengan runbook
- Gunakan runbook AWS System Manager Automation untuk menyelesaikan tugas-tugas operasional

Video terkait:

- AWS re: invent 2019: DIY panduan untuk runbook, laporan insiden, dan respons insiden (-R1) SEC318
- AWS Manajer Insiden Systems Manager - Lokakarya AWS Virtual
- Integrasikan Skrip ke dalam AWS Systems Manager

Contoh terkait:

- Kerangka Kerja Playbook Pelanggan AWS
- AWS Systems Manager: Panduan otomatisasi
- Membangun runbook respons AWS insiden menggunakan notebook Jupyter dan Danau CloudTrail
- Rubix – Pustaka Python untuk membuat runbook di Notebook Jupyter
- Menggunakan Document Builder untuk membuat runbook kustom
- Lab Well-Architected: Melakukan otomatisasi operasi dengan Playbook dan Runbook
- Lab Well-Architect: Playbook respons insiden dengan Jupyter

Layanan terkait:

- AWS Otomatisasi Systems Manager
- AWS Manajer Insiden Systems Manager

OPS07-BP05 Membuat keputusan berdasarkan informasi untuk menyebarkan sistem dan perubahan

Miliki proses untuk perubahan yang sukses dan tidak sukses pada beban kerja Anda. Pre-mortem adalah latihan simulasi tim terhadap terjadinya kegagalan untuk mengembangkan strategi mitigasi. Gunakan pre-mortem untuk mengantisipasi kegagalan dan menciptakan prosedur ketika diperlukan. Evaluasi manfaat dan risiko dari deployment perubahan ke beban kerja Anda. Pastikan apakah semua perubahan sudah mematuhi tata kelola atau tidak.

Hasil yang diinginkan:

- Anda mengambil keputusan yang tepat ketika melakukan deployment perubahan ke beban kerja Anda.
- Perubahan mematuhi tata kelola.

Anti-pola umum:

- Melakukan deployment perubahan ke beban kerja tanpa proses untuk menangani deployment yang gagal.
- Membuat perubahan pada lingkungan produksi Anda yang tidak mematuhi persyaratan-persyaratan tata kelola.
- Melakukan deployment versi beban kerja baru Anda tanpa menetapkan garis dasar untuk pemanfaatan sumber daya.

Manfaat menjalankan praktik terbaik ini:

- Anda siap untuk menangani perubahan-perubahan yang tidak berhasil dilakukan pada beban kerja Anda.
- Perubahan pada beban kerja Anda mematuhi kebijakan-kebijakan tata kelola.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Gunakan pre-mortem untuk mengembangkan proses untuk perubahan yang tidak berhasil. Buatlah dokumentasi dari proses-proses Anda untuk perubahan yang tidak berhasil. Pastikan semua perubahan mematuhi tata kelola. Evaluasi manfaat dan risiko melakukan deployment perubahan ke beban kerja Anda.

Contoh pelanggan

AnyCompany Ritel secara teratur melakukan pra-mortem untuk memvalidasi proses mereka untuk perubahan yang gagal. Mereka mendokumentasikan proses mereka di Wiki bersama dan sering kali memperbaruinya. Semua perubahan mematuhi persyaratan-persyaratan tata kelola.

Langkah-langkah implementasi

1. Ambil keputusan yang tepat ketika melakukan deployment perubahan ke beban kerja Anda. Tetapkan dan tinjau kriteria untuk deployment yang berhasil. Kembangkan skenario atau kriteria yang akan menginisiasi pengembalian perubahan ke versi sebelumnya. Pikirkan manfaat dari deployment perubahan dibandingkan dengan risiko perubahan yang tidak berhasil.
2. Pastikan bahwa semua perubahan mematuhi kebijakan tata kelola.
3. Gunakan pre-mortem guna membuat rencana untuk perubahan-perubahan yang tidak berhasil dan mendokumentasikan strategi mitigasi. Jalankan sesi latihan table-top untuk memperagakan perubahan yang tidak berhasil dan melakukan validasi terhadap prosedur pengembalian ke versi sebelumnya (roll-back).

Tingkat upaya untuk rencana implementasi: Sedang. Mengimplementasikan praktik pre-mortem memerlukan koordinasi dan upaya dari para pemangku kepentingan dalam seluruh organisasi Anda

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP03 Mengevaluasi persyaratan tata kelola - Persyaratan tata kelola merupakan faktor kunci dalam menentukan apakah akan melakukan deployment perubahan.
- OPS06-BP01 Rencana untuk perubahan yang gagal - Buat rencana untuk memitigasi deployment yang gagal dan gunakan pre-mortem untuk memvalidasinya.
- OPS06-BP02 Uji penerapan - Setiap perubahan perangkat lunak harus diuji dengan tepat sebelum deployment untuk mengurangi kecacatan dalam produksi.
- OPS07-BP01 Memastikan kemampuan personel - Memiliki cukup banyak personel yang terlatih untuk mendukung beban kerja sangat penting dalam mengambil keputusan yang tepat dalam hal deployment perubahan sistem.

Dokumen terkait:

- Amazon Web Services: Risiko dan Kepatuhan
- AWS Model Tanggung Jawab Bersama
- Tata Kelola dalam AWS Cloud: Keseimbangan yang Tepat Antara Kelincahan dan Keamanan

OPS07-BP06 Buat rencana dukungan untuk beban kerja produksi

Aktifkan dukungan untuk perangkat lunak dan layanan yang diandalkan beban kerja produksi Anda. Pilih tingkat dukungan yang sesuai untuk memenuhi kebutuhan tingkat layanan produksi Anda. Rencana dukungan untuk dependensi ini diperlukan untuk berjaga-jaga jika ada gangguan layanan atau masalah perangkat lunak yang terjadi. Buatlah dokumentasi dari rencana-rencana dukungan dan cara meminta dukungan untuk semua vendor perangkat lunak dan layanan. Implementasikan mekanisme yang memastikan bahwa titik kontak dukungan selalu yang terbaru.

Hasil yang diinginkan:

- Implementasikan rencana dukungan untuk perangkat lunak dan layanan yang diandalkan oleh beban kerja produksi.
- Pilih rencana dukungan yang sesuai berdasarkan kebutuhan tingkat layanan.
- Buatlah dokumentasi dari rencana dukungan, tingkat dukungan, dan cara meminta dukungan.

Anti-pola umum:

- Anda tidak memiliki rencana dukungan untuk vendor perangkat lunak yang penting. Beban kerja Anda terkena dampaknya dan Anda tidak dapat melakukan apa-apa untuk mempercepat perbaikan atau mendapatkan informasi terbaru dari vendor secara tepat waktu.
- Seorang pengembang yang merupakan titik utama kontak untuk vendor perangkat lunak tidak lagi bekerja di perusahaan. Anda tidak dapat menghubungi dukungan vendor secara langsung. Anda harus meluangkan waktu menelusuri dan mencari-cari dalam sistem kontak generik, sehingga menambah waktu yang diperlukan untuk memberikan respons ketika diperlukan.
- Penghentian (outage) produksi terjadi pada vendor perangkat lunak. Tidak ada dokumentasi tentang cara mengajukan kasus dukungan.

Manfaat menjalankan praktik terbaik ini:

- Dengan tingkat dukungan yang sesuai, Anda dapat memperoleh respons dalam kerangka waktu yang diperlukan untuk memenuhi kebutuhan-kebutuhan tingkat layanan.
- Sebagai pelanggan yang didukung, Anda dapat menyampaikan masalah, jika terjadi masalah produksi.
- Vendor layanan dan perangkat lunak dapat membantu menyelesaikan masalah saat terjadi insiden.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Aktifkan rencana dukungan untuk vendor perangkat lunak dan layanan yang diandalkan oleh beban kerja produksi Anda. Atur rencana dukungan yang sesuai untuk memenuhi kebutuhan-kebutuhan tingkat layanan Anda. Bagi AWS pelanggan, ini berarti mengaktifkan AWS Business Support atau lebih besar pada akun mana pun yang memiliki beban kerja produksi. Temui para vendor dukungan secara teratur untuk mendapatkan informasi terbaru mengenai penawaran dukungan, proses, dan kontak. Buatlah dokumentasi tentang cara meminta dukungan dari para vendor perangkat lunak dan layanan, termasuk cara menyampaikan masalah jika ada penghentian (outage). Implementasikan mekanisme-mekanisme untuk menjaga agar kontak selalu yang terbaru.

Contoh pelanggan

Di AnyCompany Retail, semua dependensi perangkat lunak dan layanan komersial memiliki rencana dukungan. Misalnya, mereka mengaktifkan AWS Enterprise Support di semua akun dengan beban kerja produksi. Semua pengembang dapat membuka kasus dukungan bila ada masalah yang terjadi.

Ada satu halaman wiki yang memuat informasi tentang cara meminta dukungan, siapa yang harus diberi tahu, dan praktik-praktik terbaik untuk mempercepat penanganan kasus.

Langkah-langkah implementasi

1. Bekerjasamalah dengan para pemangku kepentingan yang ada di organisasi Anda untuk mengidentifikasi para vendor perangkat lunak dan layanan yang diandalkan oleh beban kerja Anda. Buatlah dokumentasi mengenai dependensi ini.
2. Tentukan kebutuhan tingkat layanan untuk beban kerja Anda. Pilih rencana dukungan yang selaras dengannya.
3. Untuk layanan-layanan dan perangkat lunak komersial, tetapkan rencana dukungan dengan para vendornya.
    a. Berlangganan AWS Business Support atau yang lebih besar untuk semua akun produksi memberikan waktu respons yang lebih cepat dari AWS Support dan sangat disarankan. Jika Anda tidak memiliki dukungan premium, Anda harus memiliki rencana tindakan untuk menangani masalah, yang memerlukan bantuan dari AWS Support. AWS Support menyediakan campuran alat dan teknologi, orang, dan program yang dirancang untuk secara proaktif membantu Anda mengoptimalkan kinerja, menurunkan biaya, dan berinovasi lebih cepat. AWS Business Support memberikan manfaat tambahan, termasuk akses ke AWS Trusted Advisor dan AWS Personal Health Dashboard dan waktu respons yang lebih cepat.
4. Buatlah dokumentasi tentang rencana dukungan di alat manajemen pengetahuan Anda. Sertakan cara untuk meminta dukungan, siapa yang harus diberi tahu jika ada kasus dukungan diajukan, dan cara untuk menyampaikan masalah saat terjadi insiden. Wiki bisa menjadi sebuah mekanisme yang bagus untuk memungkinkan semua orang membuat pembaruan yang diperlukan pada dokumentasi ketika mereka mengetahui tentang adanya perubahan yang dilakukan untuk mendukung proses atau perubahan kontak.

Tingkat upaya untuk rencana implementasi: Rendah. Sebagian besar vendor perangkat lunak dan layanan menawarkan pilihan penyertaan rencana dukungan. Mendokumentasikan dan berbagi praktik terbaik terkait dukungan di sistem manajemen pengetahuan Anda akan memastikan tim Anda mengetahui tindakan-tindakan yang harus dilakukan jika ada masalah produksi.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP02 Proses dan prosedur telah mengidentifikasi pemilik

Dokumen terkait:

- AWS Support Rencana

Layanan terkait:

- AWS Support Bisnis
- AWS Support Perusahaan

## Jalankan

Pertanyaan

- OPS8. Bagaimana cara memanfaatkan observabilitas beban kerja di organisasi Anda?
- OPS9. Bagaimana cara memahami kondisi operasi Anda?
- OPS10. Bagaimana cara mengelola peristiwa operasi dan beban kerja?

### OPS8. Bagaimana cara memanfaatkan observabilitas beban kerja di organisasi Anda?

Memastikan kondisi beban kerja yang optimal dengan memanfaatkan observabilitas. Memanfaatkan metrik, log, dan jejak yang relevan untuk mendapatkan pandangan komprehensif tentang kinerja beban kerja Anda dan mengatasi masalah secara efisien.

Praktik terbaik

- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja
- OPS08-BP03 Menganalisis jejak beban kerja
- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti
- OPS08-BP05 Buat dasbor

OPS08-BP01 Menganalisis metrik beban kerja

Setelah mengimplementasikan telemetri aplikasi, lakukan analisis terhadap metrik yang dikumpulkan secara rutin. Latensi, permintaan, kesalahan, dan kapasitas (atau kuota) memang memberikan wawasan tentang performa sistem, tetapi memprioritaskan peninjauan terhadap metrik hasil bisnis

adalah hal yang sangat penting. Ini akan memastikan Anda mengambil keputusan berbasis data yang selaras dengan tujuan-tujuan bisnis Anda.

Hasil yang diharapkan: Wawasan akurat tentang performa beban kerja yang mendorong keputusan berdasarkan informasi data, sehingga memastikan keselarasan dengan tujuan bisnis.

Anti-pola umum:

- Menganalisis metrik secara terpisah tanpa mempertimbangkan dampak-dampak yang ditimbulkannya terhadap hasil bisnis.
- Ketergantungan berlebihan pada metrik teknis sekaligus mengesampingkan metrik bisnis.
- Peninjauan metrik jarang dilakukan, sehingga peluang pengambilan keputusan waktu nyata terlewatkan.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan pemahaman tentang korelasi antara performa teknis dan hasil bisnis.
- Perbaikan proses pengambilan keputusan yang berlandaskan data waktu nyata.
- Melakukan identifikasi dan mitigasi masalah secara proaktif sebelum hasil bisnis terkena dampaknya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Manfaatkan alat seperti Amazon CloudWatch untuk melakukan analisis metrik. AWS layanan seperti deteksi CloudWatch anomali dan Amazon DevOps Guru dapat digunakan untuk mendeteksi anomali, terutama ketika ambang batas statis tidak diketahui atau ketika pola perilaku lebih cocok untuk deteksi anomali.

Langkah-langkah implementasi

1. Lakukan analisis dan peninjauan: Tinjau dan tafsirkan metrik beban kerja Anda secara rutin.
   a. Memprioritaskan metrik hasil bisnis daripada metrik teknis murni.
   b. Memahami arti penting dari lonjakan, penurunan, atau pola dalam data Anda.
2. Manfaatkan Amazon CloudWatch: Gunakan Amazon CloudWatch untuk tampilan terpusat dan analisis mendalam.

   a. Konfigurasikan CloudWatch dasbor untuk memvisualisasikan metrik Anda dan membandingkannya dari waktu ke waktu.
   b. Gunakan persentil CloudWatch untuk mendapatkan pandangan yang jelas tentang distribusi metrik, yang dapat membantu dalam mendefinisikan SLAs dan memahami outlier.
   c. Siapkan deteksi CloudWatch anomali untuk mengidentifikasi pola yang tidak biasa tanpa bergantung pada ambang batas statis.
   d. Menerapkan observabilitas CloudWatch lintas akun untuk memantau dan memecahkan masalah aplikasi yang menjangkau beberapa akun dalam suatu Wilayah.
   e. Gunakan Wawasan CloudWatch Metrik untuk menanyakan dan menganalisis data metrik di seluruh akun dan Wilayah, mengidentifikasi tren dan anomali.
   f. Terapkan CloudWatch Metric Math untuk mengubah, menggabungkan, atau melakukan perhitungan pada metrik Anda untuk wawasan yang lebih dalam.
3. Mempekerjakan Amazon DevOps Guru: Gabungkan Amazon DevOps Guru untuk deteksi anomali yang ditingkatkan pembelajaran mesin untuk mengidentifikasi tanda-tanda awal masalah operasional untuk aplikasi tanpa server Anda dan memperbaikinya sebelum berdampak pada pelanggan Anda.
4. Lakukan optimalisasi berdasarkan wawasan: Ambil keputusan cerdas berdasarkan analisis metrik Anda untuk menyesuaikan dan meningkatkan beban kerja Anda.

Tingkat upaya untuk Rencana Implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi

Dokumen terkait:

- The Wheel Blog - Menekankan pentingnya peninjauan metrik secara terus-menerus
- Persentil itu penting
- Menggunakan AWS Cost Anomaly Detection
- CloudWatch observabilitas lintas akun
- Kueri metrik Anda dengan Wawasan CloudWatch Metrik

Video terkait:

- Aktifkan Observabilitas Lintas Akun di Amazon CloudWatch
- Pengantar Amazon DevOps Guru
- Terus Menganalisis Metrik menggunakan AWS Cost Anomaly Detection

Contoh terkait:

- Lokakarya Satu Observabilitas
- Mendapatkan wawasan operasi dengan AIOps menggunakan Amazon Guru DevOps

OPS08-BP02 Menganalisis log beban kerja

Melakukan analisis log beban kerja secara rutin merupakan hal yang sangatlah penting untuk mendapatkan pemahaman yang lebih mendalam tentang aspek-aspek operasional aplikasi Anda. Dengan memilah-milah, memvisualisasikan, dan menafsirkan data log secara efisien, Anda akan dapat terus mengoptimalkan performa dan keamanan aplikasi.

Hasil yang diinginkan: Wawasan yang kaya tentang perilaku dan operasi aplikasi yang berasal dari analisis log yang dilakukan secara menyeluruh, sehingga akan memastikan deteksi dan mitigasi masalah yang proaktif.

Anti-pola umum:

- Mengabaikan analisis log sampai ada masalah kritis yang muncul.
- Tidak menggunakan rangkaian alat lengkap yang tersedia untuk melakukan analisis log, sehingga ada wawasan kritis yang terlewatkan.
- Hanya mengandalkan tinjauan log manual tanpa memanfaatkan kemampuan-kemampuan otomatisasi dan kueri.

Manfaat menjalankan praktik terbaik ini:

- Lakukan identifikasi kemacetan operasional, ancaman keamanan, dan masalah-masalah potensial lain secara proaktif.
- Pemanfaatan data log yang efisien untuk optimalisasi aplikasi yang berkelanjutan.
- Peningkatan pemahaman tentang perilaku aplikasi, sehingga itu akan membantu Anda dalam melakukan upaya debugging dan pemecahan masalah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Amazon CloudWatch Logs adalah alat yang ampuh untuk analisis log. Fitur terintegrasi seperti Wawasan CloudWatch Log dan Wawasan Kontributor membuat proses memperoleh informasi yang bermakna dari log menjadi intuitif dan efisien.

Langkah-langkah implementasi

1. Siapkan CloudWatch Log: Konfigurasikan aplikasi dan layanan untuk mengirim CloudWatch log ke Log.
2. Gunakan deteksi anomali log: Manfaatkan deteksi anomali CloudWatch Amazon Logs untuk secara otomatis mengidentifikasi dan memperingatkan pola log yang tidak biasa. Alat ini akan membantu Anda secara proaktif mengelola anomali-anomali yang terjadi dalam log Anda dan mendeteksi setiap potensi masalah sejak dini.
3. Siapkan Wawasan CloudWatch Log: Gunakan Wawasan CloudWatch Log untuk mencari dan menganalisis data log Anda secara interaktif.
   a. Buat kueri untuk mengekstrak pola, memvisualisasikan data log, dan memperoleh wawasan yang dapat Anda tindaklanjuti.
   b. Gunakan analisis pola Wawasan CloudWatch Log untuk menganalisis dan memvisualisasikan pola log yang sering. Fitur ini akan membantu Anda memahami tren operasional umum dan setiap potensi penyimpangan yang ada dalam data log Anda.
   c. Gunakan perbandingan CloudWatch Log (diff) untuk melakukan analisis diferensial antara periode waktu yang berbeda atau di seluruh grup log yang berbeda. Gunakan kemampuan ini untuk mendeteksi perubahan-perubahan yang terjadi dan menilai dampaknya terhadap kinerja atau perilaku sistem Anda.
4. Pantau log secara real-time dengan Live Tail: Gunakan Amazon CloudWatch Logs Live Tail untuk melihat data log secara real-time. Anda dapat secara aktif memantau aktivitas operasional aplikasi Anda saat sedang berlangsung, yang memberikan visibilitas langsung kepada Anda mengenai kinerja sistem dan potensi masalah.
5. Manfaatkan Wawasan Kontributor: Gunakan Wawasan CloudWatch Kontributor untuk mengidentifikasi pembicara teratas dalam dimensi kardinalitas tinggi seperti alamat IP atau agen pengguna.
6. Menerapkan filter metrik CloudWatch Log: Konfigurasikan filter metrik CloudWatch Log untuk mengonversi data log menjadi metrik yang dapat ditindaklanjuti. Ini memungkinkan Anda untuk mengatur alarm atau melakukan analisis pola lebih lanjut.

7. Menerapkan observabilitas CloudWatch lintas akun: Pantau dan pecahkan masalah aplikasi yang menjangkau beberapa akun dalam suatu Wilayah.
8. Lakukan peninjauan dan penyempurnaan secara rutin: Tinjau strategi analisis log Anda secara berkala untuk menangkap semua informasi yang relevan dan terus mengoptimalkan performa aplikasi.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS08-BP01 Menganalisis metrik beban kerja

Dokumen terkait:

- Menganalisis Data Log dengan Wawasan CloudWatch Log
- Menggunakan CloudWatch Wawasan Kontributor
- Membuat dan Mengelola Filter Metrik CloudWatch Log

Video terkait:

- Menganalisis Data Log dengan Wawasan CloudWatch Log
- Gunakan Wawasan CloudWatch Kontributor untuk Menganalisis Data Kardinalitas Tinggi

Contoh terkait:

- CloudWatch Pertanyaan Contoh Log
- Lokakarya Satu Observabilitas

OPS08-BP03 Menganalisis jejak beban kerja

Menganalisis data jejak sangatlah penting untuk mencapai pandangan yang komprehensif tentang perjalanan operasional aplikasi. Dengan memvisualisasikan dan memahami interaksi antara berbagai

komponen, performa dapat disesuaikan, kemacetan dapat diidentifikasi, dan pengalaman pengguna dapat ditingkatkan.

Hasil yang dinginkan: Dapatkan visibilitas yang jelas tentang operasi terdistribusi yang dimiliki aplikasi Anda, sehingga memungkinkan penyelesaian masalah yang lebih cepat dan pengalaman pengguna yang disempurnakan.

Anti-pola umum:

- Mengabaikan data jejak, dan hanya mengandalkan log serta metrik.
- Tidak melakukan korelasi antara data jejak dengan log terkait.
- Mengabaikan metrik-metrik yang berasal dari jejak, seperti latensi dan tingkat kesalahan.

Manfaat menjalankan praktik terbaik ini:

- Tingkatkan pemecahan masalah dan kurangi waktu rata-rata ke resolusi ()MTTR.
- Mendapatkan wawasan tentang dependensi dan dampaknya.
- Identifikasi dan perbaikan masalah performa secara cepat.
- Memanfaatkan metrik-metrik yang berasal dari jejak untuk pengambilan keputusan yang tepat berdasarkan informasi.
- Pengalaman pengguna yang ditingkatkan melalui interaksi komponen yang dioptimalkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS X-Ray menawarkan serangkaian fitur komprehensif untuk melakukan analisis data jejak, yang dapat menyediakan pandangan yang menyeluruh tentang interaksi layanan, memantau aktivitas pengguna, dan mendeteksi masalah-masalah performa. Fitur seperti ServiceLens, X-Ray Insights, X-Ray Analytics, dan Amazon DevOps Guru meningkatkan kedalaman wawasan yang dapat ditindaklanjuti yang berasal dari data jejak.

Langkah-langkah implementasi

Langkah-langkah berikut menawarkan pendekatan terstruktur untuk menerapkan analisis data jejak secara efektif menggunakan AWS layanan:

1. Integrasikan AWS X-Ray: Pastikan X-Ray terintegrasi dengan aplikasi Anda untuk menangkap data jejak.
2. Analisis metrik X-Ray: Selidiki metrik yang berasal dari jejak X-Ray, seperti latensi, tingkat permintaan, tingkat kesalahan, dan distribusi waktu respons, dengan menggunakan peta layanan untuk memantau kesehatan aplikasi.
3. Gunakan ServiceLens: Manfaatkan ServiceLenspeta untuk meningkatkan observabilitas layanan dan aplikasi Anda. Fitur ini memungkinkan Anda untuk menampilkan jejak, metrik, log, alarm, dan informasi kondisi lainnya secara terpadu.
4. Aktifkan Wawasan X-Ray:
   a. Aktifkan Wawasan X-Ray untuk deteksi anomali otomatis dalam jejak.
   b. Periksa wawasan untuk menentukan pola dan memastikan akar masalah, misalnya peningkatan tingkat kesalahan atau latensi.
   c. Pelajari lini waktu wawasan untuk mendapatkan analisis kronologis dari masalah-masalah yang terdeteksi.
5. Gunakan Analitik X-Ray: Analitik X-Ray akan memungkinkan Anda menjelajahi data jejak secara menyeluruh, menentukan pola, dan mengekstrak wawasan.
6. Gunakan grup di X-Ray: Buat grup di X-Ray untuk memfilter jejak berdasarkan kriteria seperti latensi tinggi, sehingga memungkinkan analisis yang lebih tertarget.
7. Menggabungkan Amazon DevOps Guru: Libatkan Amazon DevOps Guru untuk mendapatkan manfaat dari model pembelajaran mesin yang menunjukkan dengan tepat anomali operasional dalam jejak.
8. Gunakan CloudWatch Synthetics: Gunakan CloudWatchSynthetics untuk membuat kenari untuk terus memantau titik akhir dan alur kerja Anda. Canary ini dapat terintegrasi dengan X-Ray untuk menyediakan data jejak untuk analisis aplikasi yang sedang diuji secara mendalam.
9. Gunakan Real User Monitoring (RUM): Dengan AWS X-Ray dan CloudWatch RUM, Anda dapat menganalisis dan men-debug jalur permintaan mulai dari pengguna akhir aplikasi Anda melalui layanan AWS terkelola hilir. Ini akan membantu Anda untuk mengidentifikasi tren latensi dan kesalahan yang berdampak pada pengguna akhir Anda.
10. Berkorelasi dengan log: Korelasikan data jejak dengan log terkait dalam tampilan jejak X-Ray untuk perspektif mendetail tentang perilaku aplikasi. Ini memungkinkan Anda untuk melihat peristiwa log yang terkait langsung dengan transaksi-transaksi yang dilacak.
11. Menerapkan observabilitas CloudWatch lintas akun: Pantau dan pecahkan masalah aplikasi yang menjangkau beberapa akun dalam suatu Wilayah.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja

Dokumen terkait:

- Menggunakan ServiceLens untuk Memantau Kesehatan Aplikasi
- Menjelajahi Data Jejak dengan Analitik X-Ray
- Mendeteksi Anomali di dalam Jejak dengan Wawasan X-Ray
- Pemantauan Berkelanjutan dengan CloudWatch Synthetics

Video terkait:

- Analisis dan Debug Aplikasi Menggunakan Amazon CloudWatch Synthetics & AWS X-Ray
- Gunakan Wawasan AWS X-Ray

Contoh terkait:

- Lokakarya Satu Observabilitas
- Menerapkan X-Ray dengan AWS Lambda
- CloudWatchTemplat Canary Synthetics

OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti

Sangat penting untuk mendeteksi dan merespons penyimpangan dalam perilaku aplikasi Anda segera. Terutama penting adalah mengenali kapan hasil berdasarkan indikator kinerja utama (KPIs) berisiko atau ketika anomali tak terduga muncul. Mendasarkan peringatan pada KPIs memastikan bahwa sinyal yang Anda terima terkait langsung dengan dampak bisnis atau operasional. Pendekatan terhadap peringatan yang dapat ditindaklanjuti ini mempromosikan respons proaktif dan akan membantu Anda untuk mempertahankan performa dan keandalan sistem.

Hasil yang diinginkan: Menerima peringatan yang tepat waktu, relevan, dan dapat ditindaklanjuti untuk identifikasi cepat dan mitigasi masalah potensial, terutama ketika KPI hasil berisiko.

Anti-pola umum:

- Menyiapkan terlalu banyak peringatan non-kritis, yang mengakibatkan kewalahan.
- Tidak memprioritaskan peringatan berdasarkanKPIs, sehingga sulit untuk memahami dampak bisnis dari masalah.
- Mengabaikan penanganan akar masalah, yang berimbas pada munculnya peringatan berulang untuk masalah yang sama.

Manfaat menjalankan praktik terbaik ini:

- Berkurangnya kewalahan akibat peringatan dengan memusatkan perhatian pada peringatan-peringatan yang dapat ditindaklanjuti dan relevan.
- Waktu aktif dan keandalan sistem yang lebih baik melalui deteksi dan mitigasi masalah yang proaktif.
- Kolaborasi tim yang disempurnakan dan penyelesaian masalah yang lebih cepat dengan melakukan integrasi alat-alat peringatan dan komunikasi populer.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk membuat mekanisme peringatan yang efektif, sangat penting untuk menggunakan metrik, log, dan data pelacakan yang menandai ketika hasil berdasarkan KPIs risiko atau anomali terdeteksi.

Langkah-langkah implementasi

1. Tentukan indikator kinerja utama (KPIs): Identifikasi aplikasi AndaKPIs. Peringatan harus dikaitkan dengan ini KPIs untuk mencerminkan dampak bisnis secara akurat.
2. Implementasikan deteksi anomali:
   - Gunakan deteksi CloudWatch anomali Amazon: Siapkan deteksi CloudWatch anomali Amazon untuk mendeteksi pola yang tidak biasa secara otomatis, yang membantu Anda hanya menghasilkan peringatan untuk anomali asli.
   - Gunakan AWS X-Ray Wawasan:

     a. Siapkan Wawasan X-Ray untuk mendeteksi anomali dalam data jejak.

    b. Konfigurasikan notifikasi untuk Wawasan X-Ray agar Anda mendapat notifikasi tentang masalah yang terdeteksi.

- Integrasikan dengan Amazon DevOps Guru:
    a. Manfaatkan Amazon DevOps Guru untuk kemampuan pembelajaran mesinnya dalam mendeteksi anomali operasional dengan data yang ada.
    b. Arahkan ke pengaturan notifikasi di DevOps Guru untuk mengatur peringatan anomali.

3. Implementasikan peringatan yang dapat ditindaklanjuti: Rancang peringatan yang menyediakan informasi yang memadai untuk tindakan cepat.
    1. Pantau AWS Health peristiwa dengan EventBridge aturan Amazon, atau integrasikan secara terprogram dengan tindakan AWS Health API untuk mengotomatisasi saat Anda menerima AWS Health acara. Ini bisa berupa tindakan-tindakan umum, seperti mengirimkan semua pesan peristiwa siklus hidup yang direncanakan ke antarmuka obrolan, atau tindakan tertentu, seperti inisiasi alur kerja di alat manajemen layanan IT.
4. Kurangi kelelahan karena peringatan: Minimalkan peringatan non-kritis. Ketika tim kewalahan dengan banyaknya peringatan yang tidak penting, mereka bisa jadi melewatkan masalah-masalah kritis, sehingga mengurangi efektivitas mekanisme peringatan secara keseluruhan.
5. Siapkan alarm komposit: Gunakan alarm CloudWatch komposit Amazon untuk mengkonsolidasikan beberapa alarm.
6. Integrasikan dengan alat peringatan: Gabungkan alat seperti Ops Genie dan PagerDuty.
7. Terlibat AWS Chatbot: Integrasikan AWS Chatbotuntuk menyampaikan peringatan ke Amazon Chime, Microsoft Teams, dan Slack.
8. Peringatan berdasarkan log: Gunakan filter metrik log CloudWatch untuk membuat alarm berdasarkan peristiwa log tertentu.
9. Tinjau dan ulangi: Tinjau ulang dan sempurnakan konfigurasi peringatan secara rutin.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS04-BP03 Menerapkan telemetri pengalaman pengguna
- OPS04-BP04 Menerapkan telemetri ketergantungan

- OPS04-BP05 Melaksanakan penelusuran terdistribusi
- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja
- OPS08-BP03 Menganalisis jejak beban kerja

Dokumen terkait:

- Menggunakan CloudWatch alarm Amazon
- Membuat sebuah alarm gabungan
- Buat CloudWatch alarm berdasarkan deteksi anomali
- DevOpsPemberitahuan Guru
- Notifikasi wawasan X-ray
- Memantau, mengoperasikan, dan memecahkan masalah AWS sumber daya Anda dengan interaktif ChatOps
- Panduan CloudWatch Integrasi Amazon | PagerDuty
- Integrasikan Opsgenie dengan Amazon CloudWatch

Video terkait:

- Buat Alarm Komposit di Amazon CloudWatch
- AWS Chatbot Ikhtisar
- AWS Di Air ft. Perintah Mutatif di AWS Chatbot

Contoh terkait:

- Alarm, manajemen insiden, dan remediasi di cloud dengan Amazon CloudWatch
- Tutorial: Membuat EventBridge aturan Amazon yang mengirimkan notifikasi AWS Chatbot
- Lokakarya Satu Observabilitas

### OPS08-BP05 Buat dasbor

Dasbor adalah tampilan yang berpusat pada manusia tentang data telemetri beban kerja Anda. Meskipun menyediakan antarmuka visual yang vital, dasbor tidak boleh menggantikan mekanisme

peringatan, melainkan hanya melengkapinya. Ketika dibuat dengan cermat, dasbor tidak hanya dapat menawarkan wawasan yang disajikan dengan cepat tentang kondisi dan kinerja sistem, tetapi juga dapat menyajikan informasi waktu nyata kepada para pemangku kepentingan tentang hasil bisnis dan dampak dari masalah yang ditimbulkannya.

Hasil yang diinginkan:

Wawasan yang jelas dan dapat ditindaklanjuti tentang kondisi sistem dan bisnis menggunakan representasi visual.

Anti-pola umum:

- Dasbor yang terlalu rumit yang mempunyai terlalu banyak metrik.
- Mengandalkan dasbor tanpa peringatan untuk deteksi anomali.
- Tidak memperbarui dasbor seiring perkembangan beban kerja.

Manfaat praktik terbaik ini:

- Visibilitas langsung ke metrik sistem kritis dan. KPIs
- Komunikasi dan pemahaman para pemangku kepentingan yang ditingkatkan.
- Wawasan yang disajikan dengan cepat tentang dampak masalah operasional.

Tingkat risiko yang dihadapi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Dasbor yang berpusat pada bisnis

Dasbor yang disesuaikan dengan bisnis KPIs melibatkan lebih banyak pemangku kepentingan. Meskipun orang-orang ini mungkin tidak tertarik pada metrik sistem, namun mereka tertarik untuk memahami implikasi bisnis dari angka-angka ini. Dasbor yang berpusat pada bisnis memastikan semua metrik teknis dan operasional yang dipantau dan dianalisis selaras dengan tujuan-tujuan bisnis secara keseluruhan. Penyelarasan ini memberikan kejelasan, memastikan semua orang memiliki pemahaman yang sama mengenai hal-hal yang penting dan hal-hal yang tidak penting. Selain itu, dasbor yang menyoroti bisnis KPIs cenderung lebih dapat ditindaklanjuti. Para pemangku kepentingan dapat dengan cepat memahami kondisi operasi, area yang perlu diperhatikan, dan dampak yang mungkin ditimbulkan terhadap hasil bisnis.

Dengan mengingat hal ini, saat membuat dasbor Anda, pastikan ada keseimbangan antara metrik teknis dan bisnis. KPIs Keduanya penting, tetapi melayani audiens yang berbeda. Idealnya, Anda harus memiliki dasbor yang memberikan pandangan menyeluruh tentang kondisi dan performa sistem sekaligus menekankan hasil bisnis utama serta implikasinya.

CloudWatch Dasbor Amazon adalah halaman beranda yang dapat disesuaikan di CloudWatch konsol yang dapat Anda gunakan untuk memantau sumber daya Anda dalam satu tampilan, bahkan sumber daya yang tersebar di berbagai Wilayah AWS akun dan akun.

Langkah-langkah implementasi

1. Buat dasbor dasar: Buat dasbor baru CloudWatch, berikan nama deskriptif.
2. Gunakan widget Markdown: Sebelum menggunakan metrik, gunakan widget Markdown untuk menambahkan konteks tekstual di bagian atas dasbor Anda. Widget ini akan menjelaskan cakupan dasbor, tingkat pentingnya metrik yang ditampilkan, dan juga dapat diisi dengan tautan-tautan ke dasbor serta alat-alat pemecahan masalah lainnya.
3. Buat variabel dasbor: Gabungkan variabel dasbor jika sesuai agar dasbor mempunyai tampilan yang dinamis dan fleksibel.
4. Buat widget metrik: Tambahkan widget metrik untuk memberikan visualisasi dari berbagai metrik yang dihasilkan oleh aplikasi Anda, lalu sesuaikan semua widget agar efektif menampilkan kondisi sistem dan hasil bisnis.
5. Kueri Wawasan Log: Manfaatkan Wawasan CloudWatch Log untuk mendapatkan metrik yang dapat ditindaklanjuti dari log Anda dan menampilkan wawasan ini di dasbor Anda.
6. Siapkan alarm: Integrasikan CloudWatchAlarm ke dasbor Anda untuk melihat cepat metrik apa pun yang melanggar ambang batasnya.
7. Gunakan Wawasan Kontributor: Gabungkan Wawasan CloudWatch Kontributor untuk menganalisis bidang kardinalitas tinggi dan mendapatkan pemahaman yang lebih jelas tentang kontributor utama sumber daya Anda.
8. Desain widget kustom: Untuk kebutuhan spesifik yang tidak dipenuhi oleh widget standar, sebaiknya Anda membuat widget kustom. Widget kustom ini dapat menarik dari berbagai sumber data atau menyajikan data dengan cara yang unik.
9. Gunakan AWS Health Dashboard: Gunakan AWS Health Dashboarduntuk mendapatkan wawasan yang lebih dalam tentang kesehatan akun Anda, acara, dan perubahan mendatang yang mungkin memengaruhi layanan dan sumber daya Anda. Anda juga bisa mendapatkan tampilan tersentralisasi untuk peristiwa kesehatan di dasbor AWS Organizations Anda atau membuat dasbor kustom Anda sendiri (untuk detail selengkapnya, lihat Contoh terkait).

10. Ulangi dan sempurnakan: Saat aplikasi Anda berkembang, tinjau kembali dasbor Anda secara teratur untuk memastikan relevansinya.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja
- OPS08-BP03 Menganalisis jejak beban kerja
- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti

Dokumen terkait:

- Membangun Dasbor untuk Visibilitas Operasional
- Menggunakan CloudWatch Dasbor Amazon

Video terkait:

- Buat Lintas Akun & CloudWatch Dasbor Lintas Wilayah
- AWS re:Invent 2021 - Mendapatkan visibilitas korporasi dengan dasbor operasional AWS Cloud )

Contoh terkait:

- Lokakarya Satu Observabilitas
- Pemantauan Aplikasi dengan Amazon CloudWatch
- AWS Health Dasbor dan Wawasan Intelijen Acara
- Visualisasikan peristiwa AWS Health menggunakan Amazon Managed Grafana

## OPS9. Bagaimana cara memahami kondisi operasi Anda?

Tetapkan, catat, dan analisis metrik operasi untuk mendapatkan visibilitas peristiwa operasi sehingga Anda dapat mengambil tindakan yang tepat.

Praktik terbaik

- OPS09-BP01 Mengukur tujuan operasi dan dengan metrik KPIs
- OPS09-BP02 Komunikasikan status dan tren untuk memastikan visibilitas beroperasi
- OPS09-BP03 Meninjau metrik operasi dan memprioritaskan peningkatan

OPS09-BP01 Mengukur tujuan operasi dan dengan metrik KPIs

Dapatkan tujuan dan KPIs yang menentukan keberhasilan operasi dari organisasi Anda dan tentukan bahwa metrik mencerminkan hal ini. Tetapkan garis acuan sebagai titik referensi dan lakukan evaluasi ulang secara rutin. Kembangkan mekanisme untuk mengumpulkan metrik-metrik tersebut dari tim untuk dievaluasi.

Hasil yang diinginkan:

- Tujuan dan KPIs untuk tim operasi organisasi telah dipublikasikan dan dibagikan.
- Metrik yang mencerminkan ini KPIs ditetapkan. Di antara contohnya adalah:
  - Kedalaman antrean tiket atau rata-rata umur tiket
  - Jumlah tiket yang dikelompokkan berdasarkan jenis masalah
  - Waktu yang dihabiskan untuk masalah kerja dengan atau tanpa prosedur operasi standar () SOP
  - Jumlah waktu yang dihabiskan untuk pulih dari push kode yang gagal
  - Volume panggilan

Anti-pola umum:

- Tenggat waktu deployment tidak terpenuhi karena pengembang disibukkan dengan tugas-tugas pemecahan masalah. Tim pengembangan menuntut lebih banyak personel, tetapi tidak dapat mengukur berapa orang yang mereka butuhkan karena waktu yang tersita tidak dapat diukur.
- Meja Tingkat 1 disiapkan untuk menangani panggilan pengguna. Seiring waktu, makin banyak beban kerja yang ditambahkan, tetapi tidak ada personel yang dialokasikan ke meja Tingkat 1 tersebut. Kepuasan pelanggan sangat rendah karena waktu panggilan semakin meningkat dan masalah berlarut-larut tanpa penyelesaian, tetapi manajemen tidak melihat indikator permasalahan ini, sehingga tidak ada tindakan yang dilakukan.
- Beban kerja yang bermasalah diserahkan kepada tim operasi terpisah untuk dilakukan pemeliharaan. Tidak seperti beban kerja lainnya, beban kerja tersebut tidak dilengkapi dengan dokumentasi dan runbook yang baik. Akibatnya, tim menghabiskan waktu lebih lama untuk

memecahkan masalah dan mengurusi kegagalan. Namun demikian, tidak ada metrik yang mendokumentasikan hal ini, sehingga akuntabilitas menjadi sulit.

Manfaat menerapkan praktik terbaik ini: Ketika pemantauan beban kerja menunjukkan status aplikasi dan layanan kita, tim operasi pemantauan memberi pemilik wawasan tentang perubahan yang terjadi di antara para pemakai beban kerja tersebut, misalnya perubahan-perubahan kebutuhan bisnis. Ukur efektivitas tim-tim tersebut dan evaluasi mereka berdasarkan sasaran bisnis dengan membuat metrik-metrik yang dapat mencerminkan status operasi. Metrik dapat menyoroti masalah dukungan atau mengidentifikasi penyimpangan ketika terjadi pergeseran dari target tingkat layanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jadwalkan waktu dengan para pemimpin bisnis dan pemangku kepentingan untuk menentukan tujuan-tujuan layanan secara keseluruhan. Tentukan tugas apa saja yang seharusnya dijalankan oleh berbagai tim operasi dan tantangan apa yang dapat mereka hadapi. Dengan menggunakan ini, bertukar pikiran indikator kinerja utama (KPIs) yang mungkin mencerminkan tujuan operasi ini. Indikator tersebut mungkin berupa kepuasan pelanggan, waktu dari konsepsi fitur hingga deployment, waktu penyelesaian masalah rata-rata, dan lain-lain.

Bekerja dariKPIs, mengidentifikasi metrik dan sumber data yang mungkin paling mencerminkan tujuan ini. Kepuasan pelanggan dapat berupa kombinasi dari berbagai metrik seperti waktu tunggu atau respons panggilan, skor kepuasan, dan jenis-jenis masalah yang disampaikan. Waktu deployment mungkin merupakan jumlah waktu yang diperlukan untuk pengujian dan deployment, serta perbaikan pasca-deployment yang perlu ditambahkan. Statistik yang menunjukkan waktu yang dihabiskan untuk berbagai jenis masalah (atau jumlah masalah tersebut) dapat memberikan wawasan tentang bagian-bagian yang memerlukan upaya tertarget.

Sumber daya

Dokumen terkait:

- Amazon QuickSight - Menggunakan KPIs
- Amazon CloudWatch - Menggunakan Metrik
- Membangun Dasbor
- Cara melacak pengoptimalan biaya Anda KPIs dengan KPI Dashboard

OPS09-BP02 Komunikasikan status dan tren untuk memastikan visibilitas beroperasi

Anda perlu mengetahui keadaan operasi Anda dan arah trennya untuk mengidentifikasi kapan hasil mungkin berisiko, apakah pekerjaan tambahan dapat didukung, atau mengidentifikasi efek perubahan terhadap tim Anda. Selama peristiwa operasi, halaman status yang dapat dijadikan acuan oleh para pengguna dan tim operasi untuk mendapatkan informasi dapat mengurangi tekanan pada saluran komunikasi dan menyebarkan informasi secara proaktif.

Hasil yang diinginkan:

- Pimpinan operasi memiliki wawasan sekilas untuk melihat volume panggilan seperti apa yang sedang dioperasikan oleh tim mereka dan upaya apa yang mungkin sedang dilakukan, seperti deployment.
- Peringatan disebarkan kepada para pemangku kepentingan dan komunitas pengguna ketika terjadi dampak terhadap operasi normal.
- Pimpinan dan pemangku kepentingan organisasi dapat memeriksa halaman status sebagai respons terhadap peringatan atau dampak, dan memperoleh informasi seputar peristiwa operasional yang terjadi, seperti titik kontak, informasi tiket, dan perkiraan waktu pemulihan.
- Laporan tersedia bagi para pimpinan dan pemangku kepentingan lainnya untuk menunjukkan statistik operasi seperti volume panggilan selama periode waktu tertentu, skor kepuasan pengguna, jumlah tiket tertunda, dan usia mereka.

Anti-pola umum:

- Terdapat beban kerja yang tidak aktif, sehingga sebuah layanan menjadi tidak tersedia. Volume panggilan melonjak karena para pengguna ingin mengetahui apa yang terjadi. Manajer menambah volume tersebut dengan permintaan informasinya tentang siapa yang mengurusi masalah. Berbagai tim operasi melipatgandakan upaya untuk melakukan penyelidikan.
- Keinginan untuk kemampuan baru menyebabkan beberapa personel dialihkan ke upaya rekayasa. Tidak ada pengisian ulang (backfill) yang disediakan, dan waktu penyelesaian masalah semakin lama. Informasi ini tidak ditangkap, dan pimpinan baru menyadari hal ini setelah beberapa minggu dan pengguna menyampaikan ketidakpuasan.

Manfaat menerapkan praktik terbaik ini: Selama peristiwa operasional yang berdampak pada bisnis, banyak waktu dan tenaga yang bisa terbuang untuk meminta informasi dari berbagai tim yang sedang berusaha memahami situasinya. Dengan membuat halaman status dan dasbor yang disebarluaskan, para pemangku kepentingan dapat dengan cepat memperoleh informasi mengenai

hal-hal seperti apakah ada masalah yang sudah terdeteksi, siapa yang memimpin penanganan masalah tersebut, atau kapan operasi diperkirakan akan kembali normal. Dengan begitu, anggota tim terhindar dari membuang-buang waktu untuk mengomunikasikan status kepada orang lain dan lebih bisa berkonsentrasi untuk menangani masalah.

Selain itu, dasbor dan laporan dapat memberikan wawasan kepada para pembuat keputusan dan pemangku kepentingan untuk melihat bagaimana tim operasi dapat menanggapi kebutuhan bisnis dan bagaimana sumber daya mereka dialokasikan. Hal ini sangat penting untuk menentukan apakah sumber daya yang memadai tersedia untuk mendukung bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Buatlah dasbor yang menunjukkan metrik-metrik utama saat ini untuk tim operasi Anda, dan buat dasbor tersebut mudah diakses oleh para pemimpin operasi serta manajemen.

Buat halaman status yang dapat diperbarui dengan cepat untuk menunjukkan apabila insiden atau peristiwa sedang berlangsung, yang mencantumkan siapa yang bertanggung jawab, dan siapa yang mengoordinasikan respons. Bagikan langkah atau solusi apa pun yang harus dipertimbangkan oleh para pengguna di halaman ini, dan sebarkan luaskan lokasinya. Imbau para pengguna untuk memeriksa lokasi ini terlebih dahulu ketika mereka dihadapkan dengan masalah yang tidak diketahui.

Kumpulkan dan sediakan laporan yang menunjukkan kondisi operasi dari waktu ke waktu, dan distribusikan hal ini kepada para pimpinan dan pengambil keputusan untuk menggambarkan pekerjaan operasi beserta tantangan dan kebutuhan.

Bagikan metrik dan laporan ini di antara tim yang paling mencerminkan tujuan KPIs dan dan di mana mereka berpengaruh dalam mendorong perubahan. Luangkan waktu khusus untuk aktivitas ini untuk meningkatkan pentingnya operasi di dalam tim dan antar-tim.

Sumber daya

Dokumen terkait:

- Mengukur Kemajuan
- Membangun dasbor untuk visibilitas operasi

Solusi terkait:

- Operasi Data

OPS09-BP03 Meninjau metrik operasi dan memprioritaskan peningkatan

Menyisihkan waktu dan sumber daya khusus untuk meninjau keadaan operasi memastikan bahwa melayani day-to-day lini bisnis tetap menjadi prioritas. Kumpulkan para pemimpin operasi dan pemangku kepentingan untuk secara rutin meninjau metrik, menegaskan kembali atau memodifikasi sasaran dan tujuan, dan memprioritaskan perbaikan.

Hasil yang diinginkan:

- Para pemimpin operasi dan staf secara rutin bertemu untuk meninjau metrik selama periode pelaporan tertentu. Tantangan dikomunikasikan, keberhasilan dirayakan, dan pelajaran yang dipetik dibagikan.
- Pemangku kepentingan dan pemimpin bisnis secara teratur diberi pengarahan tentang keadaan operasi dan diminta untuk masukan mengenai tujuan,, KPIs dan inisiatif masa depan. Kompromi antara pemberian layanan, operasi, dan pemeliharaan dibahas dan dimasukkan ke dalam konteks.

Anti-pola umum:

- Sebuah produk baru diluncurkan, tetapi tim operasi Tingkat 1 dan Tingkat 2 tidak mendapatkan pelatihan yang memadai untuk mendukung atau tidak mendapatkan staf tambahan. Metrik-metrik yang menunjukkan penurunan waktu resolusi tiket dan peningkatan volume insiden tidak terlihat oleh para pimpinan. Tindakan diambil beberapa minggu kemudian ketika jumlah langganan mulai turun karena para pengguna yang tidak puas dan beralih ke platform lain.
- Proses manual untuk melakukan pemeliharaan pada beban kerja telah berlangsung sejak lama. Meskipun sudah ada keinginan untuk melakukan otomatisasi, prioritas yang diberikan rendah mengingat rendahnya nilai penting sistem. Namun seiring waktu, sistem menjadi semakin penting dan sekarang proses manual ini menyita sebagian besar waktu operasional. Tidak ada sumber daya yang dijadwalkan untuk menyediakan peningkatan peralatan untuk operasi, sehingga menyebabkan kelelahan pada staf saat terjadi peningkatan beban kerja. Para pimpinan menyadari hal ini setelah ada laporan bahwa para staf beralih ke kompetitor.

Manfaat menerapkan praktik terbaik ini: Beberapa organisasi mengalami kesulitan untuk mengalokasikan waktu dan perhatian yang sama untuk pemberian layanan dan produk atau penawaran baru. Ketika masalah ini terjadi, lini bisnis dapat mengalami hal buruk karena tingkat layanan yang diharapkan perlahan-lahan memburuk. Alasannya adalah karena operasi tidak berubah dan berkembang sesuai dengan perkembangan bisnis, dan bisa segera tertinggal. Tanpa melakukan peninjauan rutin terhadap wawasan yang dikumpulkan oleh operasi, risiko terhadap bisnis mungkin

baru terlihat ketika semua sudah terlambat. Dengan pengalokasian waktu untuk meninjau metrik dan prosedur, baik di antara staf operasi maupun dengan pimpinan, peran penting yang dimiliki oleh operasi akan terus dapat dilihat, dan risiko dapat diidentifikasi jauh sebelum mencapai tingkat kritis. Tim operasi mendapatkan wawasan yang lebih baik tentang perubahan dan inisiatif bisnis yang akan datang, sehingga upaya-upaya proaktif dapat dilakukan. Visibilitas para pimpinan ke dalam metrik-metrik operasi menunjukkan peran penting yang dimiliki oleh tim operasional dalam hal kepuasan pelanggan, baik internal maupun eksternal, dan memungkinkan mereka mempertimbangkan pilihan prioritas dengan lebih baik, atau memastikan bahwa operasional memiliki waktu dan sumber daya untuk berubah dan berkembang seiring munculnya inisiatif bisnis dan beban kerja baru.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Luangkan waktu khusus untuk meninjau metrik-metrik operasi antara para pemangku kepentingan dan tim operasional dan meninjau data laporan. Pertimbangkan laporan-laporan berdasarkan tujuan dan sasaran organisasi untuk menentukan apakah semuanya terpenuhi. Identifikasi sumber-sumber ambiguitas yang membuat sasaran menjadi tidak jelas, atau di mana mungkin ada ketidaksesuaian antara apa yang diminta dan apa yang diberikan.

Identifikasi di mana waktu, personel, dan alat dapat membantu mencapai hasil operasi yang diharapkan. Tentukan mana yang KPIs akan berdampak dan target kesuksesan apa yang seharusnya. Lakukan peninjauan ulang secara rutin untuk memastikan operasi memiliki sumber daya yang memadai untuk mendukung lini bisnis.

Sumber daya

Dokumen terkait:

- Amazon Athena
- CloudWatch Referensi metrik dan dimensi Amazon
- Amazon QuickSight
- AWS Glue
- AWS Glue Data Catalog
- Kumpulkan metrik dan log dari EC2 instans Amazon dan server lokal dengan Agen Amazon CloudWatch
- Menggunakan CloudWatch metrik Amazon

## OPS10. Bagaimana cara mengelola peristiwa operasi dan beban kerja?

Siapkan dan validasikan prosedur untuk merespons peristiwa guna meminimalkan gangguannya pada beban kerja Anda.

Praktik terbaik

- OPS10-BP01 Gunakan proses untuk manajemen peristiwa, insiden, dan masalah
- OPS10-BP02 Memiliki proses per peringatan
- OPS10-BP03 Memprioritaskan kegiatan operasional berdasarkan dampak bisnis
- OPS10-BP04 Tentukan jalur eskalasi
- OPS10-BP05 Menentukan rencana komunikasi pelanggan untuk peristiwa yang berdampak pada layanan
- OPS10-BP06 Komunikasikan status melalui dasbor
- OPS10-BP07 Otomatiskan tanggapan terhadap acara

### OPS10-BP01 Gunakan proses untuk manajemen peristiwa, insiden, dan masalah

Kemampuan untuk mengelola peristiwa, insiden, dan masalah secara efisien adalah kunci untuk menjaga kondisi kesehatan dan kinerja beban kerja. Sangat penting untuk mengenali dan memahami perbedaan antara elemen-elemen ini untuk mengembangkan sebuah strategi respons dan resolusi yang efektif. Dengan membentuk dan mengikuti proses yang ditentukan dengan baik untuk setiap aspek, tim Anda dapat dengan cepat dan efektif menangani setiap tantangan operasional yang muncul.

Hasil yang diinginkan: Organisasi Anda mengelola peristiwa-peristiwa operasional, insiden, dan masalah secara efektif melalui proses yang terdokumentasi dengan baik dan tersimpan secara terpusat. Proses-proses tersebut diperbarui secara konsisten untuk mencerminkan setiap perubahan, merampingkan proses penanganan, dan mempertahankan keandalan layanan serta kinerja beban kerja yang tinggi.

Anti-pola umum:

- Anda merespons peristiwa secara reaktif, bukan proaktif.
- Pendekatan-pendekatan yang tidak konsisten diambil untuk berbagai jenis peristiwa atau insiden yang berbeda.
- Organisasi Anda tidak menganalisis dan belajar dari insiden-insiden yang terjadi untuk mencegah kejadian di masa mendatang.

Manfaat menjalankan praktik terbaik ini:

- Proses respons yang efisien dan terstandardisasi.
- Berkurangnya dampak insiden pada layanan dan pelanggan.
- Resolusi masalah yang lebih cepat.
- Perbaikan berkelanjutan dalam proses operasional.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Menerapkan praktik terbaik ini berarti Anda melacak peristiwa-peristiwa beban kerja. Anda memiliki proses untuk menangani insiden dan masalah. Proses ini didokumentasikan, dibagikan, dan sering diperbarui. Masalah diidentifikasi, diprioritaskan, dan diperbaiki.

Memahami peristiwa, insiden, dan masalah

- Peristiwa: Sebuah peristiwa adalah sebuah pengamatan atas suatu tindakan, kejadian, atau perubahan status. Peristiwa dapat direncanakan atau tidak direncanakan dan dapat berasal dari dalam atau luar beban kerja.
- Insiden: Insiden adalah peristiwa-peristiwa yang memerlukan respons, seperti gangguan yang tidak terencana atau penurunan kualitas layanan. Insiden-insiden tersebut mewakili gangguan yang membutuhkan perhatian cepat untuk memulihkan operasi beban kerja yang normal.
- Masalah: Masalah adalah penyebab-penyebab yang mendasari satu atau beberapa insiden. Mengidentifikasi dan menyelesaikan masalah mencakup langkah-langkah untuk menyelidiki insiden dengan lebih mendalam untuk mencegah kejadian di masa mendatang.

Langkah-langkah implementasi

Peristiwa

1. Memantau peristiwa:
   - Menerapkan observabilitas dan memanfaatkan observabilitas beban kerja.
   - Tindakan monitor yang diambil oleh pengguna, peran, atau AWS layanan dicatat sebagai peristiwa di AWS CloudTrail.
   - Menanggapi perubahan operasional dalam aplikasi Anda secara real time dengan Amazon EventBridge.

- Lakukan penilaian, pemantauan, dan pencatatan perubahan konfigurasi sumber daya secara berkelanjutan dengan AWS Config.

2. Ciptakan proses:
   - Kembangkan sebuah proses untuk menilai peristiwa mana yang signifikan dan memerlukan pemantauan. Langkah ini melibatkan pengaturan ambang batas dan parameter untuk aktivitas normal dan abnormal.
   - Tentukan kriteria eskalasi suatu peristiwa menjadi insiden. Kriteria ini dapat didasarkan pada tingkat keparahan, dampak yang ditimbulkan pada pengguna, atau penyimpangan dari perilaku yang diperkirakan.
   - Lakukan peninjauan terhadap proses pemantauan dan respons peristiwa secara rutin. Langkah ini mencakup analisis insiden masa lalu, penyesuaian ambang batas, dan penyempurnaan mekanisme pembuatan peringatan.

Insiden

1. Merespons insiden:
   - Gunakan wawasan dari alat-alat observabilitas untuk mengidentifikasi dan merespons insiden dengan cepat.
   - Implementasikan AWS Systems Manager Ops Center untuk mengagregasi, mengatur, dan memprioritaskan item dan insiden operasional.
   - Gunakan layanan seperti Amazon CloudWatch dan AWS X-Rayuntuk analisis dan pemecahan masalah yang lebih dalam.
   - Pertimbangkan AWS Managed Services (AMS) untuk meningkatkan manajemen insiden, memanfaatkan kemampuan proaktif, pencegahan, dan detektifnya. AMSmemperluas dukungan operasional dengan layanan seperti pemantauan, deteksi dan respons insiden, dan manajemen keamanan.
   - Pelanggan Dukungan Perusahaan dapat menggunakan Deteksi dan Respons Insiden AWS, yang akan menyediakan pemantauan proaktif terus-menerus dan manajemen insiden untuk beban kerja produksi.
2. Buat proses manajemen insiden:
   - Tetapkan sebuah proses manajemen insiden yang terstruktur, termasuk peran yang jelas, protokol komunikasi, dan langkah-langkah penyelesaian masalah.
   - Integrasikan manajemen insiden dengan alat seperti AWS Chatbot untuk mendapatkan respons dan koordinasi yang efisien.

- Kategorikan insiden berdasarkan tingkat keparahan, dengan rencana respons insiden yang telah ditentukan sebelumnya untuk masing-masing kategori.

3. Pelajari dan tingkatkan:
   - Lakukan analisis pasca-insiden untuk memahami akar penyebab masalah dan efektivitas penyelesaian masalah.
   - Lakukan pembaruan dan peningkatan secara berkelanjutan terhadap rencana-rencana respons berdasarkan tinjauan dan praktik yang berkembang.
   - Buatlah dokumentasi dari dan bagikan pelajaran yang diperoleh ke seluruh tim untuk meningkatkan ketahanan operasional.
   - Pelanggan Dukungan Perusahaan dapat meminta Lokakarya Manajemen Insiden dari Manajer Akun Teknis mereka. Lokakarya terpandu ini akan menguji rencana respons insiden yang ada sekarang dan akan membantu Anda mengidentifikasi area-area yang perlu ditingkatkan.

Masalah

1. Identifikasi masalah:
   - Gunakan data dari insiden-insiden sebelumnya untuk mengidentifikasi pola-pola yang berulang yang mungkin menandakan adanya masalah sistemik yang lebih mendalam.
   - Manfaatkan alat seperti AWS CloudTraildan Amazon CloudWatch untuk menganalisis tren dan mengungkap masalah mendasar.
   - Libatkan tim lintas fungsi, termasuk tim operasional, pengembangan, dan unit bisnis, untuk mendapatkan perspektif yang beragam tentang akar penyebab masalah.
2. Buat proses manajemen masalah:
   - Kembangkan sebuah proses terstruktur untuk manajemen masalah, dengan fokus pada penyelesaian masalah jangka panjang, bukan perbaikan-perbaikan cepat.
   - Menggabungkan teknik analisis akar penyebab (RCA) untuk menyelidiki dan memahami penyebab yang mendasari insiden.
   - Perbarui kebijakan operasional, prosedur, dan infrastruktur berdasarkan temuan yang didapatkan untuk mencegah terulangnya kejadian.
3. Terus lakukan perbaikan:
   - Pupuk budaya pembelajaran dan perbaikan yang konstan, dengan mendorong tim untuk mengidentifikasi dan mengatasi setiap potensi masalah secara proaktif.

- Tinjau dan revisi proses dan alat manajemen masalah agar selaras dengan lanskap bisnis dan teknologi yang berkembang.
- Bagikan wawasan dan praktik terbaik ke seluruh organisasi untuk membangun sebuah lingkungan operasional yang lebih tangguh dan efisien.

4. Terlibat AWS Support:
   - Gunakan sumber daya AWS dukungan, seperti AWS Trusted Advisor, untuk panduan proaktif dan rekomendasi pengoptimalan.
   - Pelanggan Dukungan Perusahaan dapat mengakses program khusus seperti AWS Countdown untuk mendapatkan dukungan saat terjadi peristiwa kritis.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS04-BP02 Melaksanakan telemetri aplikasi
- OPS07-BP03 Gunakan runbook untuk melakukan prosedur
- OPS07-BP04 Gunakan pedoman untuk menyelidiki masalah
- OPS08-BP01 Menganalisis metrik beban kerja
- OPS11-BP02 Lakukan analisis pasca-insiden

Dokumen terkait:

- Panduan Respons Insiden Keamanan AWS
- AWS Deteksi dan Respon Insiden
- AWS Cloud Adoption Framework: Perspektif Operasi - Insiden dan manajemen masalah
- Manajemen Insiden di Era DevOps dan SRE
- PagerDuty - Apa itu Manajemen Insiden?

Video terkait:

- Kiat respons insiden teratas dari AWS

- AWS re:invent 2022 - Amazon Builders' Library: 25 tahun keunggulan operasional Amazon
- AWS RE: invent 2022 - Deteksi dan Respons AWS Insiden (01) SUP2
- Memperkenalkan Manajer Insiden dari AWS Systems Manager

Contoh terkait:

- AWS Layanan Proaktif — Lokakarya Manajemen Insiden
- Cara Mengotomatiskan Respons Insiden dengan PagerDuty dan AWS Systems Manager Incident Manager
- Libatkan Responden Insiden dengan Jadwal Panggilan di AWS Systems Manager Incident Manager
- Meningkatkan Visibilitas dan Kolaborasi selama Penanganan Insiden AWS Systems Manager Incident Manager
- Laporan insiden dan permintaan layanan di AMS

Layanan terkait:

- Amazon EventBridge

OPS10-BP02 Memiliki proses per peringatan

Menetapkan proses yang jelas dan terdefinisi untuk setiap peringatan di dalam sistem Anda sangat penting untuk manajemen insiden yang efektif dan efisien. Praktik ini memastikan bahwa setiap peringatan menghasilkan respons spesifik yang dapat ditindaklanjuti, sehingga meningkatkan keandalan dan responsivitas operasi Anda.

Hasil yang diinginkan: Setiap peringatan memulai rencana respons spesifik dan terdefinisi dengan baik. Jika memungkinkan, respons dilakukan secara otomatis, dengan kepemilikan yang jelas dan jalur eskalasi yang sudah ditentukan. Peringatan ditautkan ke basis up-to-date pengetahuan sehingga operator mana pun dapat merespons secara konsisten dan efektif. Respons diberikan secara cepat dan seragam, sehingga meningkatkan efisiensi dan keandalan operasional.

Anti-pola umum:

- Peringatan tidak memiliki proses respons yang telah ditentukan sebelumnya, sehingga menyebabkan resolusi yang seadanya dan tertunda.

- Jumlah peringatan yang terlalu banyak dapat menyebabkan terabaikannya peringatan-peringatan penting.
- Peringatan-peringatan ditangani secara tidak konsisten karena tidak adanya kepemilikan dan tanggung jawab yang jelas.

Manfaat menjalankan praktik terbaik ini:

- Mengurangi kewalahan akibat peringatan dengan hanya memunculkan peringatan yang dapat ditindaklanjuti.
- Penurunan waktu rata-rata untuk resolusi (MTTR) untuk masalah operasional.
- Penurunan waktu rata-rata untuk menyelidiki (MTTI), yang membantu mengurangiMTTR.
- Peningkatan kemampuan untuk menskalakan respons-respons operasional.
- Peningkatan konsistensi dan keandalan dalam menangani peristiwa-peristiwa operasional.

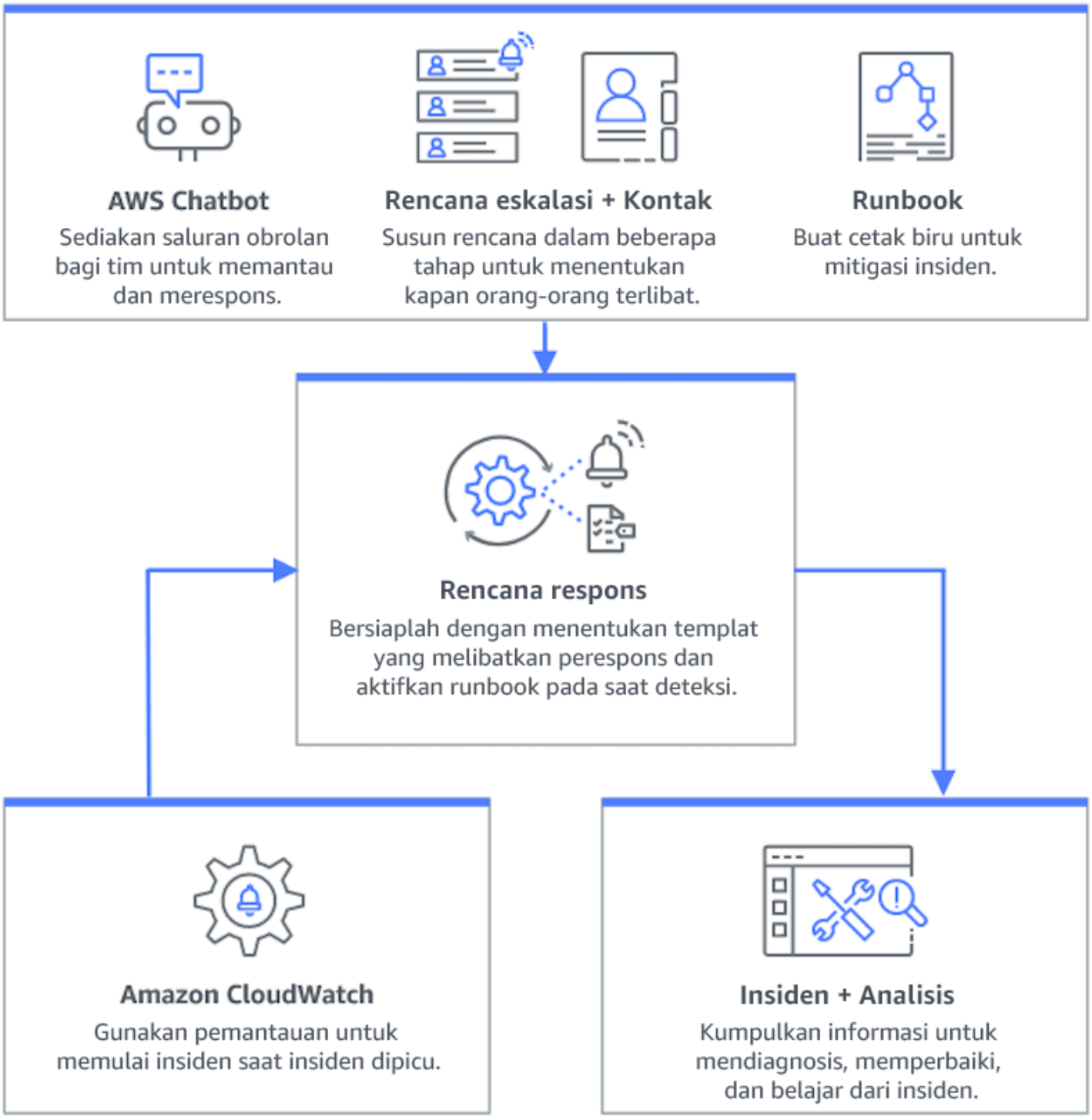
Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk membuat sebuah proses untuk setiap peringatan, diperlukan pembuatan rencana respons yang jelas untuk setiap peringatan, otomatisasi respons apabila memungkinkan, dan penyempurnaan proses-proses ini secara berkelanjutan berdasarkan umpan balik operasional dan perubahan persyaratan.

Langkah-langkah implementasi

Diagram berikut ini menggambarkan alur kerja manajemen insiden di dalam AWS Systems Manager Incident Manager. Ini dirancang untuk merespons dengan cepat masalah operasional dengan secara otomatis membuat insiden sebagai respons terhadap peristiwa tertentu dari Amazon atau CloudWatch Amazon. EventBridge Ketika insiden dibuat, baik secara otomatis atau manual, Manajer Insiden memusatkan manajemen insiden, mengatur informasi AWS sumber daya yang relevan, dan memulai rencana respons yang telah ditentukan sebelumnya. Ini termasuk menjalankan runbook Systems Manager Automation untuk tindakan segera, serta membuat item kerja operasional induk OpsCenter untuk melacak tugas dan analisis terkait. Proses yang disederhanakan ini mempercepat dan mengoordinasikan respons insiden di seluruh lingkungan Anda AWS .




1. Gunakan alarm komposit: Buat alarm komposit CloudWatch untuk mengelompokkan alarm terkait, mengurangi kebisingan dan memungkinkan respons yang lebih bermakna.
2. Integrasikan CloudWatch alarm Amazon dengan Manajer Insiden Konfigurasikan CloudWatch alarm untuk membuat insiden secara otomatis. AWS Systems Manager Incident Manager

3. Integrasikan Amazon EventBridge dengan Manajer Insiden: Buat EventBridge aturan untuk bereaksi terhadap peristiwa dan membuat insiden menggunakan rencana respons yang ditentukan.
4. Mempersiapkan insiden di Incident Manager:
   - Buat rencana respons terperinci di Incident Manager untuk setiap jenis peringatan.
   - Buat saluran obrolan melalui AWS Chatbot yang terhubung ke rencana respons di Incident Manager, yang akan memfasilitasi komunikasi waktu nyata selama insiden di seluruh platform seperti Slack, Microsoft Teams, dan Amazon Chime.
   - Menggabungkan runbook Systems Manager Automation dalam Incident Manager untuk mendorong respons otomatis terhadap insiden.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti

Dokumen terkait:

- AWS Cloud Adoption Framework: Perspektif Operasi - Insiden dan manajemen masalah
- Menggunakan CloudWatch alarm Amazon
- Menyiapkan AWS Systems Manager Incident Manager
- Mempersiapkan insiden di Incident Manager

Video terkait:

- Kiat respons insiden teratas dari AWS

Contoh terkait:

- AWS Lokakarya - AWS Systems Manager Incident Manager - Mengotomatiskan respons insiden terhadap peristiwa keamanan

OPS10-BP03 Memprioritaskan kegiatan operasional berdasarkan dampak bisnis

Merespons peristiwa operasional dengan cepat adalah hal yang sangat penting, tetapi tidak semua peristiwa sama. Ketika Anda melakukan prioritas berdasarkan dampak bisnis, Anda juga memprioritaskan penanganan peristiwa yang berpotensi menimbulkan konsekuensi signifikan, seperti keamanan, kerugian finansial, pelanggaran peraturan, atau kerusakan reputasi.

Hasil yang diinginkan: Respons terhadap peristiwa operasional diprioritaskan berdasarkan dampak yang mungkin ditimbulkannya terhadap operasi dan tujuan bisnis. Hal ini membuat respons menjadi efisien dan efektif.

Anti-pola umum:

- Setiap peristiwa diperlakukan dengan tingkat urgensi yang sama, sehingga menyebabkan kebingungan dan ketertundaan dalam menangani masalah-masalah kritis.
- Anda gagal membedakan antara peristiwa berdampak tinggi dan rendah, sehingga menyebabkan kesalahan alokasi sumber daya.
- Organisasi Anda tidak memiliki kerangka prioritas yang jelas, sehingga menghasilkan respons-respons yang tidak konsisten terhadap peristiwa-peristiwa operasional.
- Peristiwa diprioritaskan berdasarkan urutan pelaporannya, bukan dampaknya terhadap hasil bisnis.

Manfaat menjalankan praktik terbaik ini:

- Memastikan fungsi-fungsi bisnis penting mendapatkan perhatian terlebih dahulu, sehingga akan meminimalkan potensi kerugian.
- Memperbaiki alokasi sumber daya selama saat terjadi peristiwa secara serentak.
- Meningkatkan kemampuan organisasi untuk mempertahankan kepercayaan dan memenuhi persyaratan-persyaratan berdasarkan peraturan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Ketika dihadapkan dengan beberapa peristiwa operasional, sebuah pendekatan prioritas yang terstruktur berdasarkan dampak dan urgensi sangatlah penting. Pendekatan ini akan membantu Anda dalam mengambil keputusan tepat berdasarkan informasi, mengerahkan upaya pada hal-hal yang paling membutuhkan, dan mengurangi risiko terhadap kelangsungan bisnis.

Langkah-langkah implementasi

1. Lakukan penilaian dampak: Kembangkan sebuah sistem klasifikasi untuk mengevaluasi tingkat keparahan peristiwa dalam hal dampak yang mungkin ditimbulkannya terhadap operasi dan tujuan bisnis. Contoh berikut ini menunjukkan kategori-kategori dampak:

| Tingkat dampak | Deskripsi |
| --- | --- |
| Tinggi | Memengaruhi banyak staf atau pelanggan, dampak keuangan tinggi, kerusakan reputasi tinggi, atau cedera. |
| Sedang | Memengaruhi sekelompok staf atau pelanggan, dampak keuangan sedang, atau kerusakan reputasi sedang. |
| Rendah | Memengaruhi staf atau pelanggan per individu, dampak keuangan rendah, atau kerusakan reputasi rendah. |

2. Menilai urgensi: Tentukan tingkat urgensi untuk seberapa cepat suatu peristiwa membutuhkan respons, dengan mempertimbangkan faktor-faktor seperti keselamatan, implikasi keuangan, dan perjanjian tingkat layanan (). SLAs Contoh berikut ini menunjukkan kategori-kategori urgensi:

| Tingkat urgensi | Deskripsi |
| --- | --- |
| Tinggi | Kerusakan yang meningkat secara eksponens ial, dampak kerja yang sensitif terhadap waktu, eskalasi yang akan segera terjadi, atau pengguna atau kelompok terpengaruh. VIP |
| Sedang | Kerusakan meningkat dari waktu ke waktu, atau VIP pengguna tunggal atau grup terpengaruh. |
| Rendah | Kerusakan marjinal meningkat dari waktu ke waktu, atau non-time-sensitive pekerjaan terkena dampak. |

3. Buat matriks prioritas:
    - Gunakan matriks untuk melakukan referensi silang antara dampak dan urgensi, sehingga tingkat prioritas dapat ditetapkan ke berbagai kombinasi.
    - Buat agar matriks tersebut mudah diakses dan dipahami oleh semua anggota tim yang bertanggung jawab untuk memberikan respons atas peristiwa-peristiwa operasional.
    - Contoh matriks berikut ini menampilkan tingkat keparahan insiden berdasarkan urgensi dan dampak:

| Urgensi dan dampak | Tinggi | Sedang | Rendah |
|---|---|---|---|
| Tinggi | Kritis | Mendesak | Tinggi |
| Sedang | Mendesak | Tinggi | Normal |
| Rendah | Tinggi | Normal | Rendah |

4. Latih dan komunikasikan: Latih tim-tim respons tentang matriks prioritas dan pentingnya mengikuti matriks tersebut saat terjadi insiden. Komunikasikan proses penyusunan prioritas kepada semua pemangku kepentingan untuk menetapkan harapan-harapan yang jelas.
5. Integrasikan dengan respons insiden:
    - Sertakan matriks prioritas ke dalam rencana dan alat respons insiden Anda.
    - Lakukan otomatisasi terhadap klasifikasi dan penyusunan prioritas peristiwa jika memungkinkan untuk mempercepat waktu respons.
    - Pelanggan Dukungan Perusahaan dapat memanfaatkan Deteksi dan Respons Insiden AWS, yang menyediakan pemantauan proaktif dalam 24x7 dan manajemen insiden untuk beban kerja produksi.
6. Tinjau dan adaptasi: Lakukan peninjauan secara rutin terhadap efektivitas proses penyusunan prioritas dan lakukan penyesuaian berdasarkan umpan balik dan perubahan dalam lingkungan bisnis.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS03-BP03 Eskalasi didorong
- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti

- OPS09-BP01 Mengukur tujuan operasi dan dengan metrik KPIs

Dokumen terkait:

- Atlassian - Memahami tingkat keparahan insiden
- Peta Proses IT - Daftar Periksa Prioritas Insiden

OPS10-BP04 Tentukan jalur eskalasi

Tetapkan jalur eskalasi yang jelas di dalam protokol respons insiden Anda untuk memfasilitasi tindakan yang tepat waktu dan efektif. Ini termasuk menentukan petunjuk untuk eskalasi, merinci proses eskalasi, dan tindakan pra-persetujuan untuk mempercepat pengambilan keputusan dan mengurangi waktu rata-rata untuk resolusi (). MTTR

Hasil yang diinginkan: Proses terstruktur dan efisien yang meneruskan insiden ke personel yang tepat, sehingga waktu respons dan dampak menjadi minimum.

Anti-pola umum:

- Kurangnya kejelasan tentang prosedur pemulihan menyebabkan muncul respons seadanya selama insiden kritis.
- Tidak adanya penentuan izin dan kepemilikan yang mengakibatkan ketertundaan ketika diperlukan tindakan mendesak.
- Para pemangku kepentingan dan pelanggan tidak menerima informasi sesuai dengan harapan.
- Keputusan-keputusan penting tertunda.

Manfaat menjalankan praktik terbaik ini:

- Respons insiden yang efisien melalui prosedur-prosedur eskalasi yang telah ditentukan sebelumnya.
- Mengurangi waktu henti dengan tindakan-tindakan yang telah disetujui sebelumnya dan penanggung jawab yang jelas.
- Alokasi sumber daya yang lebih baik dan penyesuaian tingkat dukungan berdasarkan tingkat keparahan insiden.
- Komunikasi yang lebih baik dengan para pemangku kepentingan dan pelanggan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jalur eskalasi yang didefinisikan dengan benar sangat penting untuk respons insiden yang cepat. AWS Systems Manager Incident Manager mendukung pengaturan rencana eskalasi terstruktur dan jadwal panggilan, yang memperingatkan personel yang tepat sehingga mereka siap untuk bertindak ketika insiden terjadi.

Langkah-langkah implementasi

1. Siapkan permintaan eskalasi: Siapkan CloudWatch alarm untuk membuat insiden di. AWS Systems Manager Incident Manager
2. Siapkan jadwal panggilan: Buat jadwal panggilan di Incident Manager yang selaras dengan jalur eskalasi Anda. Bekali personel siaga dengan izin dan alat yang diperlukan untuk bertindak cepat.
3. Detail prosedur eskalasi:
   - Tentukan kondisi-kondisi spesifik yang membuat insiden harus dieskalasi.
   - Buat rencana eskalasi di Incident Manager.
   - Saluran eskalasi harus terdiri dari suatu kontak atau jadwal personel siaga.
   - Tentukan peran dan tanggung jawab tim di setiap tingkat eskalasi.
4. Tindakan mitigasi sebelum persetujuan: Lakukan kerja sama dengan pengambil keputusan untuk menyetujui tindakan di awal untuk skenario yang diantisipasi. Gunakan runbook Systems Manager Automation yang terintegrasi dengan Incident Manager untuk mempercepat resolusi insiden.
5. Tentukan kepemilikan: Identifikasi dengan jelas pemilik internal untuk setiap langkah jalur eskalasi.
6. Sediakan detail eskalasi pihak ketiga:
   - Dokumentasikan perjanjian tingkat layanan pihak ketiga (SLAs), dan selaraskan dengan tujuan internal.
   - Tetapkan protokol yang jelas untuk komunikasi vendor selama terjadi insiden.
   - Integrasikan kontak vendor ke dalam alat-alat manajemen insiden sehingga bisa diakses langsung.
   - Lakukan latihan rutin yang menyertakan skenario respons pihak ketiga.
   - Jaga agar informasi eskalasi vendor terdokumentasi dengan baik dan mudah diakses.
7. Latih dan latih rencana eskalasi: Latih tim Anda menjalankan proses eskalasi dan lakukan latihan respons insiden rutin atau hari permainan. Pelanggan Dukungan Perusahaan dapat meminta Lokakarya Manajemen Insiden.

8. Lanjutkan untuk perbaikan: Tinjau efektivitas jalur eskalasi Anda secara rutin. Perbarui proses Anda berdasarkan pelajaran yang dipetik dari insiden yang sudah lewat (post-mortem) dan umpan balik berkelanjutan.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti
- OPS10-BP02 Memiliki proses per peringatan
- OPS11-BP02 Lakukan analisis pasca-insiden

Dokumen terkait:

- AWS Systems Manager Incident Manager Rencana Eskalasi
- Bekerja dengan jadwal panggilan di Incident Manager
- Membuat dan Mengelola Runbook
- Manajemen akses sementara yang ditinggikan dengan AWS IAM Identity Center
- Atlassian - Kebijakan eskalasi untuk manajemen insiden yang efektif

OPS10-BP05 Menentukan rencana komunikasi pelanggan untuk peristiwa yang berdampak pada layanan

Komunikasi yang efektif selama peristiwa yang berdampak pada layanan sangat penting untuk menjaga kepercayaan dan transparansi dengan pelanggan. Rencana komunikasi yang terdefinisi dengan baik membantu organisasi Anda berbagi informasi dengan cepat dan jelas, baik secara internal maupun eksternal, selama insiden.

Hasil yang diinginkan:

- Rencana komunikasi yang solid sebagai pedoman yang efektif bagi para pelanggan dan pemangku kepentingan selama peristiwa yang berdampak pada layanan.
- Transparansi dalam komunikasi untuk membangun kepercayaan dan mengurangi kecemasan pelanggan.

- Meminimalkan dampak peristiwa yang berdampak pada layanan terhadap pengalaman pelanggan dan operasional bisnis.

Anti-pola umum:

- Komunikasi yang tidak memadai atau tertunda menyebabkan kebingungan dan ketidakpuasan pada pelanggan.
- Pesan yang terlalu teknis atau tidak jelas akan gagal menyampaikan dampak sebenarnya pada pengguna.
- Tidak ada strategi komunikasi yang telah ditentukan sebelumnya, sehingga menghasilkan pesan yang tidak konsisten dan reaktif.

Manfaat menjalankan praktik terbaik ini:

- Meningkatkan kepercayaan dan kepuasan pelanggan dengan melakukan komunikasi yang proaktif dan jelas.
- Mengurangi beban pada tim dukungan dengan menangani kekhawatiran pelanggan terlebih dahulu.
- Meningkatkan kemampuan untuk mengelola dan memulihkan insiden secara efektif.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Pembuatan rencana komunikasi yang komprehensif untuk peristiwa-peristiwa yang berdampak pada layanan melibatkan banyak aspek, mulai dari pemilihan saluran yang tepat hingga penyusunan pesan dan nada pesan. Rencana harus dapat disesuaikan, dapat diskalakan, dan memenuhi skenario pemadaman (outage) yang berbeda-beda.

Langkah-langkah implementasi

1. Menentukan peran dan tanggung jawab:
    - Tugaskan manajer insiden utama untuk mengawasi aktivitas respons insiden.
    - Tunjuk seorang manajer komunikasi yang bertanggung jawab untuk mengoordinasikan semua komunikasi eksternal dan internal.

- Libatkan manajer dukungan untuk menyediakan komunikasi yang konsisten melalui tiket dukungan.

2. Identifikasi saluran komunikasi: Pilih saluran seperti obrolan di tempat kerjaSMS, email, media sosial, pemberitahuan dalam aplikasi, dan halaman status. Saluran-saluran tersebut harus tangguh dan mampu beroperasi secara independen selama terjadi peristiwa yang berdampak pada layanan.
3. Lakukan komunikasi dengan cepat, jelas, dan rutin kepada pelanggan:
   - Kembangkan templat-templat untuk berbagai skenario gangguan layanan, dengan menekankan kesederhanaan dan detail-detail penting. Sertakan informasi tentang gangguan layanan, waktu penyelesaian yang diharapkan, dan dampak.
   - Gunakan Amazon Pinpoint untuk memberi tahu para pelanggan menggunakan notifikasi push, notifikasi dalam aplikasi, email, pesan teks, pesan suara, dan pesan melalui saluran khusus.
   - Gunakan Amazon Simple Notification Service (AmazonSNS) untuk mengingatkan pelanggan secara terprogram atau melalui email, pemberitahuan push seluler, dan pesan teks.
   - Komunikasikan status melalui dasbor dengan membagikan CloudWatch dasbor Amazon secara publik.
   - Dorong keterlibatan media sosial:
     - Pantau media sosial secara aktif untuk memahami sentimen pelanggan.
     - Buat postingan di platform media sosial untuk menyampaikan informasi terbaru kepada publik dan menciptakan keterlibatan komunitas.
     - Siapkan templat untuk melakukan komunikasi media sosial yang konsisten dan jelas.
4. Mengkoordinasikan komunikasi internal: Menerapkan protokol internal menggunakan alat seperti AWS Chatbot untuk koordinasi dan komunikasi tim. Gunakan CloudWatch dasbor untuk mengkomunikasikan status.
5. Atur komunikasi dengan alat dan layanan-layanan khusus:
   - Gunakan AWS Systems Manager Incident Manager dengan AWS Chatbot untuk mengatur saluran obrolan khusus untuk komunikasi internal dan koordinasi waktu nyata selama insiden.
   - Gunakan AWS Systems Manager Incident Manager runbook untuk mengotomatiskan notifikasi pelanggan melalui Amazon Pinpoint, SNS Amazon, atau alat pihak ketiga seperti platform media sosial selama insiden.
   - Integrasikan alur kerja persetujuan di dalam runbook untuk meninjau dan mengotorisasi semua komunikasi eksternal secara opsional sebelum dikirim.
6. Latih dan tingkatkan:

- Lakukan pelatihan tentang penggunaan alat dan strategi komunikasi. Berdayakan tim untuk mengambil keputusan secara tepat waktu selama terjadi insiden.
- Uji rencana komunikasi dengan menjalankan latihan rutin atau game day. Gunakan pengujian ini untuk menyempurnakan perpesanan dan mengevaluasi efektivitas saluran.
- Implementasikan mekanisme umpan balik untuk menilai efektivitas komunikasi selama terjadi insiden. Terus kembangkan rencana komunikasi berdasarkan umpan balik dan perubahan kebutuhan.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- OPS07-BP03 Gunakan runbook untuk melakukan prosedur
- OPS10-BP06 Komunikasikan status melalui dasbor
- OPS11-BP02 Lakukan analisis pasca-insiden

Dokumen terkait:

- Atlassian - Praktik terbaik komunikasi insiden
- Atlassian - Cara menulis pembaruan status yang baik
- PagerDuty - Panduan untuk Komunikasi Insiden

Video terkait:

- Atlassian - Buat rencana komunikasi insiden Anda sendiri: Templat insiden

Contoh terkait:

- AWS Health Dasbor
- Contoh pembaruan AWS status

### OPS10-BP06 Komunikasikan status melalui dasbor

Gunakan dasbor sebagai alat strategis untuk menyampaikan status operasional waktu nyata dan metrik utama kepada audiens yang berbeda, termasuk tim teknis internal, pimpinan, dan pelanggan. Dasbor ini menawarkan representasi visual tersentralisasi tentang kesehatan sistem dan kinerja bisnis, sehingga meningkatkan transparansi dan efisiensi pengambilan keputusan.

Hasil yang diinginkan:

- Dasbor Anda memberikan gambaran yang komprehensif tentang sistem dan metrik-metrik bisnis yang relevan untuk berbagai pemangku kepentingan.
- Para pemangku kepentingan dapat mengakses informasi operasional secara proaktif, sehingga mengurangi kebutuhan permintaan status yang harus sering kali dilakukan.
- Pengambilan keputusan waktu nyata disempurnakan selama operasi dan insiden normal.

Anti-pola umum:

- Rekayasawan yang bergabung dengan panggilan manajemen insiden mengharuskan adanya pembaruan status untuk mengejar ketertinggalan.
- Mengandalkan pelaporan manual untuk manajemen, yang menyebabkan ketertundaan dan potensi ketidakakuratan.
- Tim operasi sering terganggu dengan permintaan pembaruan status selama terjadi insiden.

Manfaat menjalankan praktik terbaik ini:

- Memberdayakan para pemangku kepentingan dengan akses langsung ke informasi penting, sehingga mendorong pengambilan keputusan tepat yang berdasar informasi.
- Mengurangi inefisiensi operasional dengan meminimalkan pelaporan manual dan permintaan status yang sering dilakukan.
- Meningkatkan transparansi dan kepercayaan melalui visibilitas waktu nyata pada kinerja sistem dan metrik-metrik bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Dasbor menyampaikan status sistem dan metrik-metrik bisnis Anda secara efektif dan dapat disesuaikan dengan kebutuhan kelompok audiens yang berbeda. Alat seperti CloudWatch dasbor Amazon dan Amazon QuickSight membantu Anda membuat dasbor interaktif dan real-time untuk pemantauan sistem dan intelijen bisnis.

Langkah-langkah implementasi

1. Identifikasi kebutuhan pemangku kepentingan: Tentukan kebutuhan informasi khusus untuk kelompok audiens yang berbeda-beda, seperti tim teknis, pimpinan, dan pelanggan.
2. Pilih alat yang tepat: Pilih alat yang sesuai seperti CloudWatch dasbor Amazon untuk pemantauan sistem dan Amazon QuickSight untuk intelijen bisnis interaktif.
3. Rancang dasbor yang efektif:
    - Rancang dasbor untuk menyajikan metrik yang relevan dengan jelas danKPIs, memastikannya dapat dimengerti dan dapat ditindaklanjuti.
    - Gabungkan tampilan tingkat sistem dan tingkat bisnis sesuai kebutuhan.
    - Sertakan dasbor tingkat tinggi (untuk gambaran umum) dan dasbor tingkat rendah (untuk analisis mendetail).
    - Integrasikan alarm otomatis di dalam dasbor untuk menyoroti masalah-masalah kritis.
    - Buatlah anotasi dasbor dengan sasaran dan ambang batas metrik-metrik penting untuk visibilitas langsung.
4. Integrasikan sumber data:
    - Gunakan Amazon CloudWatch untuk menggabungkan dan menampilkan metrik dari berbagai AWS layanan dan metrik kueri dari sumber data lain, menciptakan tampilan terpadu dari metrik kesehatan dan bisnis sistem Anda.
    - Gunakan fitur seperti Wawasan CloudWatch Log untuk menanyakan dan memvisualisasikan data log dari berbagai aplikasi dan layanan.
5. Berikan akses mandiri:
    - Bagikan CloudWatch dasbor dengan pemangku kepentingan terkait untuk akses informasi layanan mandiri menggunakan fitur berbagi dasbor.
    - Pastikan dasbor mudah diakses dan memberikan up-to-date informasi real-time.
6. Perbarui dan perbaiki secara rutin:

- Lakukan pembaruan dan penyempurnaan secara terus-menerus pada dasbor agar selaras dengan kebutuhan bisnis yang terus berkembang dan umpan balik yang diberikan para pemangku kepentingan.
- Tinjau dasbor secara rutin agar tetap relevan dan efektif untuk menyampaikan informasi yang diperlukan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS08-BP05 Buat dasbor

Dokumen terkait:

- Membangun dasbor untuk visibilitas operasional
- Menggunakan CloudWatch dasbor Amazon
- Membuat dasbor fleksibel dengan variabel dasbor
- Berbagi CloudWatch dasbor
- Metrik kueri dari sumber data lain
- Tambahkan widget khusus ke CloudWatch dasbor

Contoh terkait:

- Lokakarya Satu Observabilitas - Dasbor

OPS10-BP07 Otomatiskan tanggapan terhadap acara

Mengotomatiskan respons peristiwa sangatlah penting untuk penanganan operasional yang cepat, konsisten, dan bebas kesalahan. Ciptakan proses yang efisien dan gunakan alat untuk mengelola dan merespons peristiwa secara otomatis, sehingga meminimalkan intervensi manual dan meningkatkan efektivitas operasional.

Hasil yang diinginkan:

- Mengurangi kesalahan manusia dan waktu resolusi yang lebih cepat melalui otomatisasi.
- Penanganan peristiwa operasional yang konsisten dan andal.

- Peningkatan efisiensi operasional dan keandalan sistem.

Anti-pola umum:

- Penanganan peristiwa secara manual menyebabkan terjadinya penundaan dan kesalahan.
- Otomatisasi diabaikan dalam tugas-tugas penting yang repetitif.
- Tugas manual yang repetitif menyebabkan kewalahan akibat peringatan dan terlewatkannya masalah-masalah kritis.

Manfaat menjalankan praktik terbaik ini:

- Respons peristiwa yang lebih cepat, sehingga mengurangi waktu henti sistem.
- Operasi yang andal dengan penanganan peristiwa yang otomatis dan konsisten.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Integrasikan otomatisasi untuk menciptakan alur kerja operasional yang efisien dan meminimalkan intervensi manual.

Langkah-langkah implementasi

1. Identifikasi peluang otomatisasi: Tentukan tugas-tugas repetitif untuk diotomatisasi, seperti remediasi masalah, pengayaan tiket, manajemen kapasitas, penskalaan, deployment, dan pengujian.
2. Identifikasi perintah-perintah otomatisasi:
   - Menilai dan menentukan kondisi atau metrik tertentu yang memulai respons otomatis menggunakan tindakan CloudWatch alarm Amazon.
   - Gunakan Amazon EventBridge untuk merespons peristiwa dalam AWS layanan, beban kerja khusus, dan aplikasi SaaS.
   - Pertimbangkan peristiwa inisiasi seperti entri log tertentu, ambang metrik kinerja, atau perubahan status sumber daya. AWS
3. Implementasikan otomatisasi berbasis peristiwa:
   - Gunakan runbook AWS Systems Manager Otomasi untuk menyederhanakan tugas pemeliharaan, penerapan, dan remediasi.

- Membuat insiden di Manajer Insiden secara otomatis mengumpulkan dan menambahkan detail tentang AWS sumber daya yang terlibat ke insiden tersebut.
- Secara proaktif memantau kuota menggunakan Monitor Kuota untuk AWS.
- Secara otomatis menyesuaikan kapasitas dengan AWS Auto Scaling untuk menjaga ketersediaan dan kinerja.
- Otomatiskan jaringan pipa pengembangan dengan Amazon. CodeCatalyst
- Uji asap atau terus memantau titik akhir dan APIs menggunakan pemantauan sintetis.

4. Lakukan mitigasi risiko melalui otomatisasi:
   - Menerapkan respons keamanan otomatis untuk mengatasi risiko dengan cepat.
   - Gunakan AWS Systems Manager State Manager untuk mengurangi penyimpangan konfigurasi.
   - Memperbaiki sumber daya yang tidak sesuai dengan. Aturan AWS Config

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti
- OPS10-BP02 Memiliki proses per peringatan

Dokumen terkait:

- Menggunakan runbook Systems Manager Automation dengan Incident Manager
- Membuat insiden di Incident Manager
- AWS kuota layanan
- Pantau penggunaan sumber daya dan kirim notifikasi saat mendekati kuota
- AWS Auto Scaling
- Apa itu Amazon CodeCatalyst?
- Menggunakan CloudWatch alarm Amazon
- Menggunakan tindakan CloudWatch alarm Amazon
- Memediasi Sumber Daya yang Tidak Sesuai dengan Aturan AWS Config
- Membuat metrik dari peristiwa log dengan menggunakan filter

- AWS Systems Manager State Manager

Video terkait:

- Buat Runbook Otomasi dengan AWS Systems Manager
- Cara mengotomatiskan Operasi TI di AWS
- AWS Security Hub aturan otomatisasi
- Mulai proyek perangkat lunak Anda dengan cepat dengan CodeCatalyst cetak biru Amazon

Contoh terkait:

- Amazon CodeCatalyst Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat modern
- Lokakarya Satu Observabilitas
- Menanggapi insiden menggunakan Incident Manager

## Kembangkan

Pertanyaan

- OPS11. Bagaimana cara mengembangkan operasi?

### OPS11. Bagaimana cara mengembangkan operasi?

Luangkan waktu dan sumber daya khusus untuk peningkatan bertahap yang hampir berkelanjutan untuk meningkatkan dan efisiensi operasi Anda.

Praktik terbaik

- OPS11-BP01 Memiliki proses untuk perbaikan berkelanjutan
- OPS11-BP02 Lakukan analisis pasca-insiden
- OPS11-BP03 Menerapkan loop umpan balik
- OPS11-BP04 Melakukan manajemen pengetahuan
- OPS11-BP05 Tentukan driver untuk perbaikan
- OPS11-BP06 Validasi wawasan

- OPS11-BP07 Lakukan tinjauan metrik operasi
- OPS11-BP08 Mendokumentasikan dan berbagi pelajaran
- OPS11-BP09 Alokasikan waktu untuk melakukan perbaikan

OPS11-BP01 Memiliki proses untuk perbaikan berkelanjutan

Evaluasi beban kerja Anda berdasarkan praktik terbaik arsitektur internal dan eksternal. Lakukan tinjauan beban kerja yang sering dan terencana. Prioritaskan peluang perbaikan ke dalam jadwal pengembangan perangkat lunak Anda.

Hasil yang diinginkan:

- Anda sering menganalisis beban kerja berdasarkan praktik-praktik terbaik arsitektur.
- Anda memberikan peluang perbaikan dengan prioritas yang setara pada fitur-fitur di dalam proses pengembangan perangkat lunak Anda.

Anti-pola umum:

- Anda belum menjalankan peninjauan arsitektur pada beban kerja Anda sejak melakukan deployment beberapa tahun lalu.
- Anda memberikan prioritas yang lebih rendah untuk peluang perbaikan. Dibandingkan dengan fitur-fitur baru, peluang ini tetap berada di backlog.
- Tidak ada standar untuk mengimplementasikan modifikasi terhadap praktik-praktik terbaik untuk organisasi.

Manfaat menjalankan praktik terbaik ini:

- Beban kerja Anda disimpan up-to-date pada praktik terbaik arsitektur.
- Anda mengembangkan beban kerja Anda secara terencana.
- Anda dapat memanfaatkan praktik-praktik terbaik organisasi untuk meningkatkan semua beban kerja.
- Anda menghasilkan keuntungan stabil yang memberikan dampak kumulatif, yang mendorong efisiensi yang lebih menyeluruh.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Lakukan tinjauan arsitektur dari beban kerja Anda dalam rentang waktu yang lebih pendek. Gunakan praktik-praktik terbaik internal dan eksternal, evaluasi beban kerja Anda, dan identifikasi peluang perbaikan. Prioritaskan peluang perbaikan ke dalam jadwal pengembangan perangkat lunak Anda.

Langkah-langkah implementasi

1. Lakukan peninjauan arsitektur secara berkala pada beban kerja produksi Anda dengan frekuensi yang sudah disepakati. Gunakan standar arsitektur terdokumentasi yang mencakup praktik terbaik AWS-spesifik.
   a. Gunakan standar yang ditetapkan secara internal untuk peninjauan ini. Jika Anda tidak memiliki standar internal, gunakan Kerangka Kerja AWS Well-Architected.
   b. Gunakan AWS Well-Architected Tool untuk membuat lensa kustom dari praktik terbaik internal Anda dan lakukan tinjauan arsitektur Anda.
   c. Hubungi Arsitek AWS Solusi atau Manajer Akun Teknis Anda untuk melakukan Tinjauan Kerangka Kerja Well-Architected yang dipandu atas beban kerja Anda.
2. Prioritaskan peluang perbaikan yang diidentifikasi selama peninjauan ke dalam proses pengembangan perangkat lunak Anda.

Tingkat upaya untuk rencana implementasi: Rendah. Anda dapat menggunakan AWS Well-Architected Framework untuk melakukan tinjauan arsitektur tahunan Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP02 Lakukan analisis pasca-insiden
- OPS11-BP08 Mendokumentasikan dan berbagi pelajaran
- OPS04 Menerapkan Observabilitas

Dokumen terkait:

- AWS Well-Architected Tool - Lensa kustom
- Laporan Resmi AWS Well-Architected - Proses peninjauan
- Sesuaikan Ulasan Well-Architected menggunakan Lensa Kustom dan AWS Well-Architected Tool
- Menerapkan siklus AWS hidup Lensa Kustom yang Dirancang dengan Baik di organisasi Anda

Video terkait:

- Well-Architected Labs - Level 100: Lensa Kustom aktif AWS Well-Architected Tool
- AWS Re:invent 2023 - Menskalakan praktik terbaik yang Dirancang dengan Baik AWS di seluruh organisasi Anda

Contoh terkait:

- AWS Well-Architected Tool

OPS11-BP02 Lakukan analisis pasca-insiden

Tinjau peristiwa yang memengaruhi pelanggan dan identifikasi faktor yang berkontribusi serta tindakan pencegahannya. Gunakan informasi ini untuk mengembangkan langkah-langkah mitigasi untuk meminimalkan atau mencegah kemungkinan terjadi lagi. Kembangkan prosedur untuk respons efektif dan cepat. Komunikasikan faktor-faktor yang berkontribusi dan tindakan-tindakan korektif yang diperlukan, yang disesuaikan dengan audiens target.

Hasil yang diinginkan:

- Anda telah menetapkan proses manajemen insiden yang mencakup analisis pasca-insiden.
- Anda menerapkan rencana observabilitas untuk mengumpulkan data tentang peristiwa.
- Dengan data ini, Anda memahami dan mengumpulkan metrik yang mendukung proses analisis pasca-insiden Anda.
- Anda belajar dari insiden untuk meningkatkan hasil di masa depan.

Anti-pola umum:

- Anda mengelola server aplikasi. Kira-kira setiap 23 jam 55 menit, semua sesi aktif Anda dihapus. Anda berupaya mengidentifikasi masalah yang terjadi di server aplikasi Anda. Anda menduga bahwa masalah ini mungkin masalah jaringan, tetapi tidak dapat memperoleh bantuan dari tim jaringan karena mereka terlalu sibuk. Anda tidak menetapkan proses di awal yang dapat Anda jadikan panduan untuk mendapatkan dukungan dan mengumpulkan informasi yang dibutuhkan guna mengetahui masalah yang sedang terjadi.
- Anda mengalami kehilangan data di dalam beban kerja Anda. Hal ini baru pertama kali terjadi dan penyebabnya belum jelas. Anda menganggap bahwa kejadian ini tidak penting karena Anda dapat

membuat ulang data. Kehilangan data makin sering terjadi dan memengaruhi pelanggan Anda. Hal ini juga menambah beban operasional Anda karena harus memulihkan data yang hilang.

Manfaat menjalankan praktik terbaik ini:

- Anda memiliki proses yang telah ditetapkan di awal untuk menentukan komponen, kondisi, tindakan, dan peristiwa yang berkontribusi terhadap suatu insiden, yang membantu Anda mengidentifikasi peluang untuk perbaikan.
- Anda menggunakan data dari analisis pasca-insiden untuk melakukan perbaikan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Gunakan sebuah proses untuk menentukan faktor yang berkontribusi. Tinjau semua insiden yang memengaruhi pelanggan. Buatlah sebuah proses untuk mengidentifikasi dan membuat dokumentasi dari faktor-faktor yang berkontribusi terhadap sebuah insiden agar Anda dapat mengembangkan langkah-langkah mitigasi untuk membatasi atau mencegah kejadian serupa serta mengembangkan prosedur untuk merespons dengan cepat dan efektif. Komunikasikan akar masalah insiden sebagaimana mestinya, dan sesuaikan komunikasi dengan audiens target Anda. Bagikan pembelajaran secara terbuka di dalam organisasi Anda.

Langkah-langkah implementasi

1. Kumpulkan meterik-metrik seperti perubahan deployment, perubahan konfigurasi, waktu mulai insiden, waktu alarm, waktu keterlibatan, waktu mulai mitigasi, dan waktu penyelesaian insiden.
2. Jelaskan titik-titik waktu utama pada kronologi (timeline) untuk memahami peristiwa insiden.
3. Ajukan pertanyaan-pertanyaan berikut:
    a. Apakah Anda dapat mempersingkat waktu deteksi?
    b. Apakah ada pembaruan metrik dan alarm yang dapat mendeteksi insiden lebih dini?
    c. Apakah Anda dapat mempersingkat waktu diagnosis?
    d. Apakah ada pembaruan pada rencana respons atau rencana eskalasi Anda yang melibatkan perespons yang tepat lebih dini?
    e. Apakah Anda dapat mempersingkat waktu mitigasi?
    f. Apakah ada langkah-langkah runbook atau panduan yang dapat Anda tambahkan atau tingkatkan?

g. Apakah Anda dapat mencegah terjadinya insiden di masa mendatang?

4. Buat daftar periksa dan tindakan. Lacak dan selesaikan semua tindakan.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP01 Memiliki proses untuk perbaikan berkelanjutan
- OPS4 - Menerapkan observabilitas

Dokumen terkait:

- Menjalankan analisis pasca-insiden di Incident Manager
- Peninjauan Kesiapan Operasional

OPS11-BP03 Menerapkan loop umpan balik

Loop umpan balik menyediakan wawasan yang dapat ditindaklanjuti yang mendorong pengambilan keputusan. Masukkan loop umpan balik ke dalam prosedur dan beban kerja Anda. Ini akan membantu Anda mengidentifikasi permasalahan dan area yang memerlukan perbaikan. Loop umpan balik juga memvalidasi investasi yang dilakukan dalam upaya perbaikan. Loop umpan balik ini adalah landasan untuk meningkatkan beban kerja Anda secara berkelanjutan.

Loop umpan balik terbagi dalam dua kategori: umpan balik langsung dan analisis retrospektif. Umpan balik langsung (immediate feedback) dikumpulkan melalui peninjauan kinerja dan hasil dari aktivitas operasi. Umpan balik ini berasal dari anggota tim, pelanggan, atau output otomatis dari aktivitas. Umpan balik langsung diterima dari hal-hal seperti pengujian A/B dan pengiriman fitur baru, dan ini penting bagi gagal cepat (fail fast).

Analisis retrospektif dilakukan secara rutin untuk menangkap umpan balik dari peninjauan metrik dan hasil operasional dari waktu ke waktu. Retrospektif ini terjadi pada akhir sprint, secara terjadwal, atau setelah perilisan atau peristiwa besar. Tipe loop umpan balik ini memvalidasi investasi dalam operasi atau beban kerja Anda. Loop umpan balik ini akan membantu Anda mengukur keberhasilan dan memvalidasi strategi Anda.

Hasil yang diinginkan: Anda menggunakan umpan balik langsung dan analisis retrospektif untuk mendorong perbaikan. Terdapat sebuah mekanisme untuk mendapatkan umpan balik dari pengguna dan anggota tim. Analisis retrospektif digunakan untuk mengidentifikasi kecenderungan yang mendorong perbaikan.

Anti-pola umum:

- Anda meluncurkan fitur baru tetapi tidak ada cara untuk menerima umpan balik pelanggan tentangnya.
- Setelah berinvestasi dalam perbaikan operasi, Anda tidak melakukan analisis retrospektif untuk memvalidasinya.
- Anda mengumpulkan umpan balik pelanggan tetapi tidak meninjaunya secara rutin.
- Loop umpan balik mendatangkan item-item tindakan yang diajukan tetapi item-item tersebut tidak disertakan dalam proses pengembangan perangkat lunak.
- Pelanggan tidak menerima umpan balik tentang perbaikan yang mereka ajukan.

Manfaat menjalankan praktik terbaik ini:

- Anda dapat bekerja mundur (work backward) dari pelanggan untuk mendorong fitur-fitur baru.
- Budaya organisasi Anda dapat merespons perubahan lebih cepat.
- Tren digunakan untuk mengidentifikasi peluang perbaikan.
- Retrospektif memvalidasi investasi yang dilakukan pada beban kerja dan operasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Dengan mengimplementasikan praktik-praktik terbaik ini, Anda dapat menggunakan umpan balik langsung dan analisis retrospektif. Loop umpan balik ini mendorong perbaikan. Terdapat banyak mekanisme untuk umpan balik langsung, termasuk survei, jajak pendapat pelanggan, atau formulir umpan balik. Organisasi Anda juga menggunakan retrospektif untuk mengidentifikasi peluang-peluang perbaikan dan memvalidasi inisiatif.

Contoh pelanggan

AnyCompany Retail membuat formulir web di mana pelanggan dapat memberikan umpan balik atau melaporkan masalah. Selama melakukan scrum mingguan, umpan balik pengguna dievaluasi

oleh tim pengembangan perangkat lunak. Umpan balik digunakan secara rutin sebagai landasan pengembangan platform mereka. Mereka melakukan analisis retrospektif di akhir setiap sprint untuk mengidentifikasi item yang ingin mereka tingkatkan.

Langkah-langkah implementasi

1. Umpan balik langsung
   - Anda memerlukan sebuah mekanisme untuk menjangkau umpan balik dari pelanggan dan anggota tim. Aktivitas operasi Anda juga dapat dikonfigurasi untuk menghadirkan umpan balik otomatis.
   - Organisasi Anda perlu sebuah proses untuk meninjau umpan balik ini, menentukan hal-hal yang harus ditingkatkan, dan menjadwalkan perbaikan.
   - Umpan balik harus ditambahkan ke dalam proses pengembangan perangkat lunak Anda.
   - Seiring Anda melakukan perbaikan, lakukan tindak lanjut dengan pemberi umpan balik.
     - Anda dapat menggunakan AWS Systems Manager OpsCenteruntuk membuat dan melacak peningkatan ini sebagai OpsItems.
2. Analisis retrospektif
   - Lakukan retrospektif di akhir siklus pengembangan, pada jadwal yang ditetapkan, atau setelah perilisan besar.
   - Kumpulkan para pemangku kepentingan yang terlibat dalam beban kerja untuk melakukan rapat retrospektif.
   - Buat tiga kolom di papan tulis atau lembar kerja: Hentikan, Mulai, dan Pertahankan
     - Stop adalah untuk apa pun yang Anda ingin tidak dilakukan lagi oleh tim Anda.
     - Start adalah gagasan yang ingin mulai Anda lakukan.
     - Keep adalah untuk item-item yang ingin tetap Anda lakukan.
   - Keliling ruangan dan kumpulkan umpan balik dari para pemangku kepentingan.
   - Buat prioritas umpan balik. Tetapkan tindakan dan pemangku kepentingan untuk item-item Mulai atau Pertahankan.
   - Tambahkan tindakan-tindakan ke proses pengembangan perangkat lunak Anda dan sampaikan pembaruan status kepada para pemangku kepentingan seiring Anda melakukan perbaikan.

Tingkat upaya untuk rencana implementasi: Sedang. Untuk mengimplementasikan praktik terbaik ini, Anda memerlukan cara untuk menyerap umpan balik langsung dan menganalisisnya. Selain itu, Anda perlu membangun sebuah proses analisis retrospektif.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP01 Mengevaluasi kebutuhan pelanggan: Loop umpan balik adalah sebuah mekanisme untuk mengumpulkan kebutuhan pelanggan eksternal.
- OPS01-BP02 Mengevaluasi kebutuhan nasabah internal: Para pemangku kepentingan internal dapat menggunakan loop umpan balik untuk menyampaikan kebutuhan dan persyaratan.
- OPS11-BP02 Lakukan analisis pasca-insiden: Analisis pasca-insiden adalah bentuk analisis retrospektif yang penting yang dilakukan setelah insiden.
- OPS11-BP07 Lakukan tinjauan metrik operasi: Peninjauan metrik operasi mengidentifikasi tren dan area perbaikan.

Dokumen terkait:

- 7 Jebakan yang Harus Dihindari Saat Membangun CCOE
- Playbook Tim Atlassian - Retrospektif
- Definisi Email: Loop Umpan Balik
- Membangun Loop Umpan Balik Berdasarkan Tinjauan Kerangka AWS Well-Architected
- IBMMetodologi Garasi - Pegang retrospektif
- Investopedia — Siklus PDCS
- Memaksimalkan Efektivitas Developer oleh Tim Cochran
- Ulasan Kesiapan Operasi (ORR) Whitepaper - Iterasi
- ITILCSI- Peningkatan Layanan Berkelanjutan
- Saat Toyota bertemu e-commerce: Bersandar pada Amazon

Video terkait:

- Membangun Loop Umpan Balik Pelanggan yang Efektif

Contoh terkait:

- Astuto - Alat umpan balik pelanggan sumber terbuka
- AWS Solusi - Q nABot on AWS

- Fider - Platform untuk mengatur umpan balik pelanggan

Layanan terkait:

- AWS Systems Manager OpsCenter

OPS11-BP04 Melakukan manajemen pengetahuan

Manajemen pengetahuan membantu anggota tim menemukan informasi untuk melakukan pekerjaan mereka. Di dalam organisasi yang mau belajar, informasi dibagikan secara bebas sehingga individu diberdayakan. Informasi dapat ditemukan atau dicari. Informasi bersifat akurat dan mutakhir. Ada mekanisme untuk membuat informasi baru, memperbarui informasi yang sudah ada, dan mengarsipkan informasi yang kedaluwarsa. Contoh paling umum dari platform manajemen pengetahuan adalah suatu sistem manajemen konten seperti wiki.

Hasil yang diinginkan:

- Anggota tim memiliki akses ke informasi yang akurat secara tepat waktu.
- Informasi dapat dicari.
- Ada mekanisme untuk menambahkan, memperbarui, dan mengarsipkan informasi.

Anti-pola umum:

- Tidak ada penyimpanan pengetahuan tersentralisasi. Anggota tim mengelola catatan mereka sendiri di mesin mereka secara lokal.
- Anda memiliki wiki yang di-hosting secara mandiri tetapi tidak ada mekanisme untuk mengelola informasi, yang mengakibatkan informasi menjadi kedaluwarsa.
- Seseorang melihat ada informasi yang kurang tetapi tidak ada proses untuk meminta penambahannya ke tim wiki. Mereka menambahkannya sendiri tetapi mereka melewatkan langkah yang penting, sehingga mengakibatkan terjadinya gangguan (outage).

Manfaat menjalankan praktik terbaik ini:

- Anggota tim diberdayakan karena informasi dibagikan secara bebas.
- Anggota tim baru menjalani masa orientasi dengan lebih cepat karena dokumentasinya mutakhir dan dapat dicari.

- Informasi bersifat tepat waktu, akurat, dan dapat ditindaklanjuti.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Manajemen pengetahuan adalah segi penting dari organisasi yang mau belajar. Untuk memulai, Anda memerlukan tempat penyimpanan terpusat guna menyimpan pengetahuan Anda (contoh yang umum yakni wiki yang di-hosting secara mandiri). Anda harus membuat proses untuk menambahkan, memperbarui, dan mengarsipkan pengetahuan. Buatlah standar mengenai apa yang harus didokumentasikan dan izinkan semua orang memberi kontribusi.

Contoh pelanggan

AnyCompany Retail menghosting Wiki internal tempat semua pengetahuan disimpan. Anggota tim didorong untuk menambahkan pengetahuan seiring pengerjaan tugas sehari-hari mereka. Setiap tiga bulan sekali, tim lintas fungsi mengevaluasi halaman mana yang paling jarang diperbarui dan menentukan apakah halaman tersebut harus diarsipkan atau diperbarui.

Langkah-langkah implementasi

1. Mulailah dengan mengidentifikasi sistem manajemen konten tempat pengetahuan akan disimpan. Dapatkan kesepakatan dari para para pemangku kepentingan di seluruh organisasi Anda.
   a. Jika Anda belum memiliki sistem manajemen konten, pertimbangkan untuk menjalankan wiki yang di-hosting secara mandiri atau menggunakan tempat penyimpanan kontrol versi sebagai titik awal.
2. Kembangkan runbook untuk menambahkan, memperbarui, dan mengarsipkan informasi. Didik tim Anda tentang proses-proses ini.
3. Identifikasi pengetahuan apa yang harus disimpan di sistem manajemen konten. Mulailah dengan aktivitas harian (runbook dan playbook) yang dilakukan anggota tim. Bekerja samalah dengan para pemangku kepentingan untuk memprioritaskan pengetahuan yang akan ditambahkan.
4. Secara berkala, bekerja dengan pemangku kepentingan untuk mengidentifikasi out-of-date informasi dan mengarsipkannya atau memperbaruinya.

Tingkat upaya untuk rencana implementasi: Sedang. Jika Anda belum memiliki sistem manajemen konten, Anda dapat membuat wiki yang di-hosting secara mandiri atau menggunakan tempat penyimpanan dokumen dengan pengontrolan versi.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS11-BP08 Mendokumentasikan dan berbagi pelajaran - Manajemen pengetahuan memfasilitasi pembagian informasi tentang pelajaran yang didapatkan.

Dokumen terkait:

- Atlassian - Manajemen Pengetahuan

Contoh terkait:

- DokuWiki
- Gollum
- MediaWiki
- Wiki.js

OPS11-BP05 Tentukan driver untuk perbaikan

Identifikasi pendorong perbaikan untuk membantu Anda mengevaluasi dan memprioritaskan peluang berdasarkan data dan loop umpan balik. Jelajahi peluang perbaikan di dalam sistem dan proses Anda, dan otomatiskan jika sesuai.

Hasil yang diinginkan:

- Anda melacak data dari seluruh lingkungan Anda.
- Anda mengorelasikan peristiwa dan aktivitas dengan hasil bisnis.
- Anda dapat mencari kesamaan dan perbedaan di antara lingkungan dan sistem.
- Anda memelihara riwayat aktivitas mendetail untuk deployment dan hasil Anda.
- Anda mengumpulkan data untuk mendukung postur keamanan Anda.

Anti-pola umum:

- Anda mengumpulkan data dari seluruh lingkungan Anda tetapi tidak mengorelasikan peristiwa dan aktivitas.

- Anda mengumpulkan data terperinci dari seluruh perkebunan Anda, dan itu mendorong Amazon CloudWatch dan AWS CloudTrail aktivitas serta biaya yang tinggi. Namun, Anda tidak menggunakan data ini secara bermakna.
- Anda tidak memperhitungkan hasil bisnis ketika menentukan pendorong untuk perbaikan.
- Anda tidak mengukur dampak fitur-fitur baru.

Manfaat menjalankan praktik terbaik ini:

- Anda meminimalkan dampak motivasi berbasis peristiwa atau investasi emosional dengan menentukan kriteria perbaikan.
- Anda merespons peristiwa-peristiwa bisnis, bukan hanya peristiwa teknis.
- Anda mengukur lingkungan Anda untuk mengidentifikasi area-area perbaikan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Pahami pendorong perbaikan: Anda sebaiknya hanya melakukan perubahan pada suatu sistem ketika hasil-hasil yang diinginkan didukung.
  - Kemampuan yang diinginkan: Lakukan evaluasi terhadap fitur dan kemampuan yang diinginkan saat mengevaluasi peluang-peluang perbaikan.
    - Apa yang baru dengan AWS
  - Masalah yang tidak dapat diterima: Lakukan evaluasi pada masalah, bug, dan kerentanan yang tidak dapat diterima saat mengevaluasi peluang-peluang perbaikan. Lacak opsi penyesuaian ukuran, dan cari peluang optimalisasi.
    - Buletin Keamanan Terkini AWS
    - AWS Trusted Advisor
    - Cloud Intelligence Dashboard
  - Persyaratan kepatuhan: Lakukan evaluasi terhadap pembaruan dan perubahan yang diperlukan untuk mempertahankan kepatuhan Anda terhadap peraturan, kebijakan, atau agar tetap memperoleh dukungan pihak ketiga, saat meninjau peluang untuk perbaikan.
    - Kepatuhan AWS
    - Program Kepatuhan AWS
    - Berita Terbaru Kepatuhan AWS

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01 Prioritas organisasi
- OPS02 Hubungan dan Kepemilikan
- OPS04-BP01 Identifikasi indikator kinerja utama
- OPS08 Memanfaatkan Observabilitas Beban Kerja
- OPS09 Memahami Kesehatan Operasional
- OPS11-BP03 Menerapkan loop umpan balik

Dokumen terkait:

- Amazon Athena
- Amazon QuickSight
- Kepatuhan AWS
- Berita Terbaru Kepatuhan AWS
- Program Kepatuhan AWS
- AWS Glue
- Buletin Keamanan Terkini AWS
- AWS Trusted Advisor
- Ekspor data log Anda ke Amazon S3
- Yang Baru dengan AWS
- Keharusan Inovasi yang Berpusat pada Pelanggan
- Transformasi Digital: Kehebohan Sementara atau Kebutuhan Strategis?

Video Terkait

- AWS Re: invent 2023 - Meningkatkan efisiensi dan ketahanan operasional dengan (0) AWS Support SUP31

### OPS11-BP06 Validasi wawasan

Tinjau respons dan hasil analisis Anda dengan tim lintas fungsi serta pemilik bisnis. Gunakan tinjauan tersebut untuk menetapkan pemahaman umum, mengidentifikasi dampak-dampak tambahan, dan menentukan alur tindakan. Sesuaikan respons sebagaimana mestinya.

Hasil yang diinginkan:

- Anda meninjau wawasan bersama dengan para pemilik bisnis secara rutin. Pemilik bisnis memberikan konteks tambahan untuk wawasan yang baru diperoleh.
- Anda meninjau wawasan dan meminta umpan balik dari rekan-rekan di bidang teknis, dan Anda membagikan pembelajaran Anda ke seluruh tim.
- Anda memublikasikan data dan wawasan untuk ditinjau oleh tim teknis dan bisnis lainnya. Anda memperhitungkan pembelajaran Anda untuk praktik-praktik baru oleh departemen lain.
- Ringkas dan tinjau wawasan baru bersama para pemimpin senior. Pemimpin senior menggunakan wawasan baru untuk menentukan strategi.

Anti-pola umum:

- Anda merilis fitur baru. Fitur ini mengubah beberapa perilaku pelanggan Anda. Observabilitas Anda tidak memperhitungkan perubahan ini. Anda tidak mengukur manfaat perubahan ini.
- Anda mendorong pembaruan baru dan mengabaikan penyegaran AndaCDN. CDNCache tidak lagi kompatibel dengan rilis terbaru. Anda mengukur persentase permintaan dengan kesalahan. Semua pengguna Anda melaporkan HTTP 400 kesalahan saat berkomunikasi dengan server backend. Anda menyelidiki kesalahan klien dan menemukan bahwa waktu Anda terbuang sia-sia karena Anda mengukur dimensi yang salah.
- Perjanjian tingkat layanan Anda menetapkan waktu aktif 99,9%, dan sasaran titik pemulihan Anda adalah empat jam. Pemilik layanan menyatakan bahwa sistem memiliki nol waktu henti (down time). Anda mengimplementasikan solusi replikasi yang mahal dan kompleks, yang menyita banyak waktu dan uang.

Manfaat menjalankan praktik terbaik ini:

- Ketika Anda memvalidasi wawasan bersama para pemilik bisnis dan orang yang ahli di bidangnya, Anda membangun pemahaman yang sama dan memandu perbaikan dengan lebih efektif.
- Anda menemukan masalah tersembunyi dan memperhitungkannya untuk keputusan masa depan.

- Fokus Anda beralih dari hasil teknis ke hasil bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Validasikan wawasan: Berinteraksi dengan para pemilik bisnis dan orang yang ahli di bidangnya untuk memastikan ada pemahaman dan kesepakatan bersama tentang makna data yang dikumpulkan. Identifikasi masalah-masalah tambahan, dampak potensial, dan tentukan alur tindakan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS01-BP06 Mengevaluasi pengorbanan sambil mengelola manfaat dan risiko
- OPS02-BP06 Tanggung jawab antar tim telah ditentukan sebelumnya atau dinegosiasikan
- OPS11-BP03 Menerapkan loop umpan balik

Dokumen terkait:

- Merancang Cloud Center of Excellence (CCOE)

Video terkait:

- Membangun observabilitas untuk meningkatkan ketahanan

OPS11-BP07 Lakukan tinjauan metrik operasi

Lakukan analisis retrospektif rutin terhadap metrik operasi dengan peserta lintas tim dari berbagai area bisnis. Gunakan tinjauan ini untuk mengidentifikasi peluang perbaikan, potensi pilihan tindakan, dan untuk membagikan pelajaran yang diperoleh. Cari peluang perbaikan di semua lingkungan Anda (misalnya pengembangan, pengujian, dan produksi).

Hasil yang diinginkan:

- Anda sering meninjau metrik yang memengaruhi bisnis
- Anda mendeteksi dan meninjau anomali melalui kemampuan observabilitas Anda

- Anda menggunakan data untuk mendukung hasil dan sasaran bisnis

Anti-pola umum:

- Jendela pemeliharaan Anda mengganggu promosi retail yang signifikan. Bisnis tidak tahu bahwa ada jadwal pemeliharaan standar yang dapat ditunda jika terdapat peristiwa lain yang memengaruhi bisnis.
- Anda mengalami pemadaman berkepanjangan karena Anda umumnya menggunakan pustaka yang sudah usang di organisasi Anda. Sejak saat itu Anda beralih ke pustaka yang didukung. Tim-tim lain yang ada di organisasi Anda tidak tahu bahwa mereka terpapar risiko.
- Anda tidak secara teratur meninjau pencapaian pelangganSLAs. Anda sedang tren untuk tidak bertemu pelanggan SLAs Anda. Ada hukuman finansial yang terkait dengan tidak bertemu pelanggan SLAs Anda.

Manfaat menjalankan praktik terbaik ini:

- Ketika Anda melakukan pertemuan rutin untuk meninjau-metrik metrik operasi, peristiwa, dan insiden, Anda dapat menjaga pemahaman bersama lintas tim.
- Tim Anda bertemu secara rutin untuk meninjau metrik dan insiden, yang memposisikan Anda untuk mengambil tindakan terhadap risiko dan mengenali pelanggan. SLAs
- Anda berbagi pelajaran yang diperoleh, yang menyediakan data untuk penyusunan prioritas dan perbaikan tertarget untuk hasil bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Lakukan analisis retrospektif rutin terhadap metrik operasi dengan peserta lintas tim dari berbagai area bisnis.
- Libatkan pemangku kepentingan, termasuk tim bisnis, pengembangan, dan operasi, untuk memvalidasi temuan dari umpan balik langsung dan analisis retrospektif, serta untuk membagikan pelajaran yang diperoleh.
- Gunakan wawasan mereka untuk mengidentifikasi peluang perbaikan dan potensi pilihan tindakan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS08-BP05 Buat dasbor
- OPS09-BP03 Meninjau metrik operasi dan memprioritaskan peningkatan
- OPS10-BP01 Gunakan proses untuk manajemen peristiwa, insiden, dan masalah

Dokumen terkait:

- Amazon CloudWatch
- CloudWatch Referensi metrik dan dimensi Amazon
- Menerbitkan metrik kustom
- Menggunakan CloudWatch metrik Amazon
- Dasbor dan visualisasi dengan CloudWatch

OPS11-BP08 Mendokumentasikan dan berbagi pelajaran

Dokumentasikan dan bagikan pelajaran yang didapatkan dari aktivitas operasional sehingga Anda dapat menggunakannya secara internal dan di seluruh tim. Anda harus membagikan pelajaran yang didapatkan oleh tim Anda guna meningkatkan manfaat di seluruh organisasi Anda. Bagikan informasi dan sumber daya untuk mencegah kesalahan yang dapat dihindari dan memudahkan upaya pengembangan, dan berfokus pada pengiriman fitur-fitur yang diinginkan.

Gunakan AWS Identity and Access Management (IAM) untuk menentukan izin yang mengizinkan akses terkontrol ke sumber daya yang ingin Anda bagikan di dalam dan di seluruh akun.

Hasil yang diinginkan:

- Anda menggunakan repositori terkontrol versi untuk membagikan pustaka aplikasi, prosedur dalam skrip, dokumentasi prosedur, dan dokumentasi sistem lainnya.
- Anda membagikan standar infrastruktur Anda dalam bentuk templat AWS CloudFormation terkontrol versi.
- Anda meninjau pelajaran yang didapatkan di seluruh tim.

Anti-pola umum:

- Anda mengalami pemadaman berkepanjangan karena organisasi Anda umumnya menggunakan pustaka yang mengandung masalah. Sejak saat itu Anda beralih ke pustaka yang andal. Tim-tim lain di organisasi Anda tidak mengetahui bahwa mereka terpapar risiko. Tidak ada orang yang mendokumentasikan dan membagikan pengalaman dengan pustaka ini, dan mereka tidak menyadari risiko tersebut.
- Anda mengidentifikasi sebuah masalah edge di dalam layanan mikro yang digunakan bersama secara internal yang menyebabkan terganggunya sesi. Anda pun memperbarui panggilan Anda ke layanan guna menghindari masalah edge tersebut. Tim-tim lain yang ada di organisasi Anda tidak tahu bahwa mereka terpapar risiko.
- Anda telah menemukan cara untuk secara signifikan mengurangi persyaratan CPU pemanfaatan untuk salah satu layanan mikro Anda. Anda tidak tahu bahwa tim lain bisa memanfaatkan teknik ini.

Manfaat menerapkan praktik terbaik ini: Bagikan pelajaran yang didapatkan untuk mendukung perbaikan dan memaksimalkan manfaat pengalaman.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

- Dokumentasikan dan bagikan pelajaran yang didapatkan Miliki prosedur untuk mendokumentasikan pelajaran didapatkan dari aktivitas operasional dan analisis retrospektif agar dapat digunakan oleh tim lain.
- Bagikan pembelajaran: Miliki prosedur untuk membagikan pelajaran yang didapatkan serta artefak terkait ke seluruh tim. Sebagai contoh, bagikan prosedur, panduan, tata kelola, dan praktik terbaik yang telah diperbarui melalui wiki yang dapat diakses. Bagikan skrip, kode, dan pustaka melalui repositori umum.
  - Mendelegasikan akses ke lingkungan Anda AWS
  - Bagikan AWS CodeCommit repositori

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP06 Tanggung jawab antar tim telah ditentukan sebelumnya atau dinegosiasikan
- OPS05-BP01 Gunakan kontrol versi
- OPS05-BP06 Bagikan standar desain

- OPS11-BP03 Menerapkan loop umpan balik
- OPS11-BP07 Lakukan tinjauan metrik operasi

Dokumen terkait:

- Kurangi penundaan proyek dengan solusi docs-as-code

Video terkait:

- Mendelegasikan akses ke lingkungan Anda AWS
- AWS Support Anda | Menjelajahi Latihan Diskusi Manajemen Insiden

OPS11-BP09 Alokasikan waktu untuk melakukan perbaikan

Dedikasikan waktu dan sumber daya dalam proses Anda untuk memungkinkan peningkatan bertahap yang berkelanjutan.

Hasil yang diinginkan:

- Anda dapat membuat duplikat lingkungan sementara, yang menurunkan risiko, usaha, serta biaya eksperimen dan pengujian.
- Lingkungan duplikat ini dapat digunakan untuk menguji kesimpulan dari analisis dan eksperimen Anda, serta mengembangkan dan menguji peningkatan terencana.
- Anda menjalankan gamedays, dan Anda menggunakan Fault Injection Service (FIS) untuk menyediakan kontrol dan pagar pembatas yang dibutuhkan tim untuk menjalankan eksperimen di lingkungan seperti produksi.

Anti-pola umum:

- Ada masalah performa yang diketahui dalam aplikasi Anda. Ini ditambahkan ke backlog di balik setiap implementasi fitur terencana. Jika peringkat fitur terencana yang ditambahkan tetap konstan, masalah performa tidak akan pernah tertangani.
- Untuk mendukung peningkatan berkelanjutan yang disetujui, administrator dan developer menggunakan seluruh waktu tambahan mereka untuk memilih dan mengimplementasikan peningkatan. Tidak ada peningkatan yang diselesaikan.
- Penerimaan operasional sudah selesai, dan Anda tidak menguji praktik operasional lagi.

Manfaat menerapkan praktik terbaik ini: Dengan mendedikasikan waktu dan sumber daya dalam proses, Anda memungkinkan peningkatan bertahap yang berkelanjutan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

- Alokasikan waktu untuk membuat peningkatan: Dedikasikan waktu dan sumber daya dalam proses Anda untuk membuat peningkatan bertahap yang berkelanjutan.
- Implementasikan perubahan guna meningkatkan dan mengevaluasi hasil untuk menentukan keberhasilan.
- Jika hasilnya tidak memenuhi tujuan, dan peningkatan masih menjadi prioritas, lakukan tindakan alternatif.
- Simulasikan beban kerja produksi melalui game day, dan gunakan pembelajaran dari simulasi ini untuk melakukan peningkatan.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP08 Gunakan beberapa lingkungan

Video terkait:

- AWS RE: invent 2023 - Meningkatkan ketahanan aplikasi dengan Fault Injection Service AWS

# Keamanan

Pilar Keamanan berkenaan dengan kemampuan untuk melindungi data, sistem, dan aset untuk memanfaatkan teknologi cloud guna meningkatkan keamanan Anda. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keamanan.

Area praktik terbaik

- Fondasi keamanan
- Pengelolaan identitas dan akses
- Deteksi

- Perlindungan infrastruktur
- Perlindungan data
- Respons insiden
- Keamanan aplikasi

## Fondasi keamanan

Pertanyaan

- SEC1. Bagaimana cara Anda mengoperasikan beban kerja Anda dengan aman?

### SEC1. Bagaimana cara Anda mengoperasikan beban kerja Anda dengan aman?

Untuk mengoperasikan beban kerja dengan aman, Anda harus menerapkan praktik terbaik yang menyeluruh ke setiap area keamanan. Pilih persyaratan dan proses yang telah Anda tetapkan dalam keunggulan operasional di tingkat organisasi dan beban kerja, lalu terapkan ke semua area. Tetap up to date dengan rekomendasi industri AWS dan intelijen ancaman membantu Anda mengembangkan model ancaman dan tujuan pengendalian Anda. Otomatisasi proses, pengujian, dan validasi keamanan memungkinkan Anda menskalakan operasi keamanan Anda.

Praktik terbaik

- SEC01-BP01 Memisahkan beban kerja menggunakan akun
- SEC01-BP02 Pengguna root akun aman dan properti
- SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan pengendalian
- SEC01-BP04 Tetap up to date dengan ancaman dan rekomendasi keamanan
- SEC01-BP05 Mengurangi ruang lingkup manajemen keamanan
- SEC01-BP06 Mengotomatiskan penerapan kontrol keamanan standar
- SEC01-BP07 Mengidentifikasi ancaman dan memprioritaskan mitigasi menggunakan model ancaman
- SEC01-BP08 Mengevaluasi dan menerapkan layanan dan fitur keamanan baru secara teratur

#### SEC01-BP01 Memisahkan beban kerja menggunakan akun

Terapkan pagar pembatas umum dan isolasi di antara lingkungan (seperti produksi, pengembangan, dan pengujian) dan beban kerja melalui strategi multi-akun. Pemisahan di tingkat akun sangat

disarankan karena hal ini dapat memberikan batasan isolasi yang kuat untuk keamanan, tagihan, dan akses.

Hasil yang diinginkan: Struktur akun yang mengisolasi operasi cloud, beban kerja yang tidak terkait, dan lingkungan ke dalam akun terpisah, meningkatkan keamanan di seluruh infrastruktur cloud.

Anti-pola umum:

- Menempatkan beberapa beban kerja yang tidak saling berkaitan dengan berbagai tingkat sensitivitas data ke dalam akun yang sama.
- Struktur unit organisasi (OU) yang tidak ditentukan dengan baik.

Manfaat menjalankan praktik terbaik ini:

- Mengurangi cakupan dampak jika beban kerja tidak sengaja diakses.
- Tata kelola pusat akses ke AWS layanan, sumber daya, dan Wilayah.
- Keamanan infrastruktur cloud terjaga dengan kebijakan dan administrasi tersentralisasi pada layanan keamanan.
- Pembuatan akun dan proses pemeliharaan otomatis.
- Audit infrastruktur terpusat untuk persyaratan kepatuhan dan peraturan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Akun AWS menyediakan batas isolasi keamanan antara beban kerja atau sumber daya yang beroperasi pada tingkat sensitivitas yang berbeda. AWS menyediakan alat untuk mengelola beban kerja cloud Anda dalam skala besar melalui strategi multi-akun untuk memanfaatkan batas isolasi ini. Untuk panduan tentang konsep, pola, dan implementasi strategi multi-akun AWS, lihat Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun.

Bila Anda memiliki beberapa Akun AWS di bawah manajemen pusat, akun Anda harus diatur ke dalam hierarki yang ditentukan oleh lapisan unit organisasi (OUs). Kontrol keamanan kemudian dapat diatur dan diterapkan ke akun OUs dan anggota, menetapkan kontrol pencegahan yang konsisten pada akun anggota dalam organisasi. Kontrol keamanan diwariskan, memungkinkan Anda memfilter izin yang tersedia untuk akun anggota yang berada di tingkat yang lebih rendah dalam hierarki OU. Untuk membuat desain yang baik, memanfaatkan pewarisan ini untuk mengurangi jumlah dan

kerumitan kebijakan keamanan yang diperlukan untuk mencapai kontrol keamanan yang diinginkan untuk setiap akun anggota.

AWS Organizationsdan AWS Control Towermerupakan dua layanan yang dapat Anda gunakan untuk menerapkan dan mengelola struktur multi-akun ini di AWS lingkungan Anda. AWS Organizations memungkinkan Anda untuk mengatur akun ke dalam hierarki yang ditentukan oleh satu atau lebih lapisanOUs, dengan setiap OU berisi sejumlah akun anggota. Kebijakan kontrol layanan (SCPs) memungkinkan administrator organisasi untuk membuat kontrol pencegahan terperinci pada akun anggota, dan AWS Configdapat digunakan untuk membuat kontrol proaktif dan detektif pada akun anggota. Banyak AWS layanan terintegrasi dengan AWS Organizations untuk menyediakan kontrol administratif yang didelegasikan dan melakukan tugas khusus layanan di semua akun anggota dalam organisasi.

Berlapis di atas AWS Organizations, AWS Control Towermenyediakan pengaturan praktik terbaik satu-klik untuk AWS lingkungan multi-akun dengan landing zone. Zona landasan adalah titik masuk ke lingkungan multi-akun yang dibuat oleh Control Tower. Control Tower memberikan beberapa manfaat dibanding AWS Organizations. Tiga manfaat yang memberikan tata kelola akun yang lebih baik adalah:

- Kontrol keamanan wajib terintegrasi yang diterapkan secara otomatis ke akun yang diterima di organisasi.
- Kontrol opsional yang dapat dihidupkan atau dimatikan untuk satu set tertentuOUs.
- AWS Control Tower Account Factory menyediakan penerapan otomatis akun yang berisi baseline yang telah disetujui sebelumnya dan opsi konfigurasi di dalam organisasi Anda.

Langkah-langkah implementasi

1. Merancang struktur unit organisasi: Struktur unit organisasi yang dirancang dengan baik mengurangi beban manajemen yang diperlukan untuk membuat dan memelihara kebijakan kontrol layanan dan kontrol keamanan lainnya. Struktur unit organisasi Anda harus selaras dengan kebutuhan bisnis Anda, sensitivitas data, dan struktur beban kerja.
2. Buat zona landasan untuk lingkungan multi-akun Anda: zona landasan menyediakan fondasi keamanan dan infrastruktur yang konsisten dari mana organisasi Anda dapat dengan cepat mengembangkan, meluncurkan, dan menerapkan beban kerja. Anda dapat menggunakan zona landasan atau AWS Control Tower yang dibuat khusus untuk mengatur lingkungan Anda.
3. Tetapkan pagar pembatas: Terapkan pagar pembatas keamanan yang konsisten untuk lingkungan Anda melalui zona landasan Anda. AWS Control Tower menyediakan daftar kontrol wajib dan

opsional yang dapat digunakan. Deployment kontrol wajib dilakukan secara otomatis saat mengimplementasikan Control Tower. Lihat daftar kontrol yang sangat direkomendasikan dan opsional, kemudian implementasikan kontrol yang sesuai dengan kebutuhan Anda.

4. Batasi akses ke Wilayah yang baru ditambahkan: Untuk yang baru Wilayah AWS, IAM sumber daya seperti pengguna dan peran hanya disebarkan ke Wilayah yang Anda tentukan. Tindakan ini dapat dilakukan melalui konsol saat menggunakan Control Tower, atau dengan menyesuaikan kebijakan IAM izin AWS Organizations.
5. Pertimbangkan AWS CloudFormation StackSets: StackSets membantu Anda menyebarkan sumber daya termasuk IAM kebijakan, peran, dan grup ke berbagai Akun AWS dan Wilayah dari templat yang disetujui.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP04 Mengandalkan penyedia identitas terpusat

Dokumen terkait:

- AWS Control Tower
- Pedoman Audit Keamanan AWS
- IAMPraktik Terbaik
- Gunakan CloudFormation StackSets untuk menyediakan sumber daya di beberapa wilayah Akun AWS dan
- Organizations FAQ
- AWS Organizations terminologi dan konsep
- Praktik Terbaik untuk Kebijakan Kontrol Layanan di Lingkungan AWS Organizations Multi-Akun
- Panduan Referensi Manajemen Akun AWS
- Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun

Video terkait:

- Aktifkan adopsi AWS dalam skala besar dengan menggunakan otomatisasi dan tata kelola
- Praktik Terbaik Keamanan dengan Cara Well-Architected

- Membangun dan Mengatur Beberapa Akun menggunakan AWS Control Tower
- Aktifkan Control Tower untuk Organisasi yang Ada

Lokakarya terkait:

- Hari Perendaman Control Tower

SEC01-BP02 Pengguna root akun aman dan properti

Pengguna root adalah pengguna yang paling istimewa dalam sebuah Akun AWS, dengan akses administratif penuh ke semua sumber daya dalam akun, dan dalam beberapa kasus tidak dapat dibatasi oleh kebijakan keamanan. Melakukan deaktivasi akses terprogram ke pengguna root, menerapkan kontrol yang sesuai untuk pengguna root, serta tidak menggunakan pengguna root secara rutin membantu mengurangi risiko tersebarnya kredensial root secara tidak sengaja dan penyusupan di lingkungan cloud.

Hasil yang diinginkan: Mengamankan pengguna root membantu mengurangi kemungkinan kerusakan yang tidak disengaja atau disengaja dapat terjadi melalui penyalahgunaan kredensial pengguna root. Menerapkan kontrol-kontrol detektif juga dapat memberikan peringatan kepada personel yang tepat saat ada tindakan dilakukan menggunakan pengguna root.

Anti-pola umum:

- Menggunakan pengguna root untuk tugas selain yang memerlukan kredensial pengguna root.
- Tidak melakukan pengujian terhadap rencana-rencana darurat secara rutin untuk memverifikasi fungsi infrastruktur, proses, dan personel penting dalam keadaan darurat.
- Hanya mempertimbangkan alur masuk akun biasa dan tidak mempertimbangkan atau menguji metode pemulihan akun lainnya.
- Tidak menanganiDNS, server email, dan penyedia telepon sebagai bagian dari perimeter keamanan kritis, karena ini digunakan dalam aliran pemulihan akun.

Manfaat menerapkan praktik terbaik ini: Mengamankan akses ke pengguna root akan membangun keyakinan bahwa tindakan-tindakan yang dilakukan di akun Anda sudah dikontrol dan diaudit.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS menawarkan banyak alat untuk membantu mengamankan akun Anda. Namun, karena beberapa tindakan ini tidak dinyalakan secara default, Anda harus mengambil tindakan langsung untuk mengimplementasikannya. Pertimbangkan rekomendasi berikut sebagai langkah-langkah dasar untuk mengamankan Akun AWS Anda. Saat mengimplementasikan langkah-langkah ini, penting halnya untuk membangun sebuah proses yang dilakukan untuk menilai dan memantau kontrol keamanan secara berkelanjutan.

Ketika Anda pertama kali membuat Akun AWS, Anda mulai dengan satu identitas yang memiliki akses lengkap ke semua AWS layanan dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root. Anda dapat masuk sebagai pengguna root menggunakan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Karena peningkatan akses yang diberikan kepada pengguna AWS root, Anda harus membatasi penggunaan pengguna AWS root untuk melakukan tugas yang secara khusus memerlukannya. Kredensi login pengguna root harus dijaga ketat, dan otentikasi multi-faktor (MFA) harus selalu digunakan untuk pengguna root. Akun AWS

Selain aliran otentikasi normal untuk masuk ke pengguna root Anda menggunakan nama pengguna, kata sandi, dan otentikasi multi-faktor (MFA) perangkat, ada aliran pemulihan akun untuk masuk ke pengguna Akun AWS root Anda yang diberikan akses ke alamat email dan nomor telepon yang terkait dengan akun Anda. Oleh karena itu, pastikan Anda mengamankan akun email pengguna root yang digunakan untuk mengirimkan email pemulihan dan nomor telepon yang terkait dengan akun tersebut. Juga pertimbangkan potensi dependensi melingkar di mana alamat email yang terkait dengan pengguna root di-host di server email atau sumber daya layanan nama domain (DNS) dari yang sama. Akun AWS

Saat menggunakan AWS Organizations, ada beberapa yang Akun AWS masing-masing memiliki pengguna root. Satu akun ditetapkan sebagai akun manajemen dan beberapa lapisan akun anggota kemudian dapat ditambahkan di bawah akun manajemen. Prioritaskan pengamanan pengguna root di akun manajemen Anda, lalu hubungi pengguna root akun anggota Anda. Strategi pengamanan pengguna root akun manajemen Anda dapat berbeda dari pengguna root akun anggota, dan Anda dapat menerapkan kontrol keamanan preventif pada pengguna root akun anggota Anda.

Langkah-langkah implementasi

Langkah-langkah implementasi berikut direkomendasikan untuk membuat kontrol bagi pengguna root tersebut. Jika berlaku, rekomendasi direferensikan silang ke benchmark CIS AWS Foundations versi 1.4.0. Selain langkah-langkah ini, konsultasikan pedoman praktik AWS terbaik untuk mengamankan Anda Akun AWS dan sumber daya.

Kontrol pencegahan

1. Siapkan informasi kontak yang akurat untuk akun tersebut.
    a. Informasi ini digunakan untuk aliran pemulihan kata sandi yang hilang, aliran pemulihan akun MFA perangkat yang hilang, dan untuk komunikasi penting terkait keamanan dengan tim Anda.
    b. Gunakan alamat email yang di-host oleh domain perusahaan Anda, sebaiknya dari daftar distribusi, sebagai alamat email pengguna root Anda. Menggunakan daftar distribusi memberikan redundansi tambahan dan keberlanjutan akses ke akun root dalam waktu lama dibanding menggunakan akun email individu.
    c. Nomor telepon yang tercantum pada informasi kontak harus berupa telepon khusus dan aman untuk tujuan ini. Nomor telepon ini tidak boleh dicantumkan atau dibagikan kepada siapa pun.
2. Jangan membuat kunci akses untuk pengguna root. Jika kunci akses ada, hapus (CIS1.4).
    a. Hilangkan kredensial terprogram yang sudah lama (kunci rahasia dan akses) untuk pengguna root.
    b. Jika kunci akses pengguna root sudah ada, Anda harus melakukan transisi proses menggunakan kunci tersebut untuk menggunakan kunci akses sementara dari peran AWS Identity and Access Management (IAM), lalu hapus kunci akses pengguna root.
3. Tentukan apakah Anda perlu menyimpan kredensial untuk pengguna root.
    a. Jika Anda menggunakan AWS Organizations untuk membuat akun anggota baru, kata sandi awal untuk pengguna root pada akun anggota baru diatur ke nilai acak yang tidak terpapar kepada Anda. Pertimbangkan untuk menggunakan alur pengaturan ulang kata sandi dari akun manajemen AWS Organisasi Anda untuk mendapatkan akses ke akun anggota jika diperlukan.
    b. Untuk akun mandiri Akun AWS atau manajemen AWS Organisasi, pertimbangkan untuk membuat dan menyimpan kredensil dengan aman untuk pengguna root. Gunakan MFA untuk pengguna root.
4. Gunakan kontrol pencegahan untuk pengguna root akun anggota di lingkungan AWS multi-akun.
    a. Pertimbangkan untuk menggunakan pagar penjaga pencegahan Jangan Izinkan Pembuatan Kunci Akses Root untuk Pengguna Root untuk akun anggota.
    b. Pertimbangkan untuk menggunakan pagar penjaga pencegahan Jangan Izinkan Tindakan sebagai Pengguna Root untuk akun anggota.
5. Jika Anda memerlukan kredensial untuk pengguna root:
    a. Gunakan kata sandi yang kompleks.
    b. Aktifkan otentikasi multi-faktor (MFA) untuk pengguna root, terutama untuk akun AWS Organizations manajemen (pembayar) (CIS1.5).

c. Pertimbangkan MFA perangkat keras untuk ketahanan dan keamanan, karena perangkat sekali pakai dapat mengurangi kemungkinan perangkat yang berisi MFA kode Anda dapat digunakan kembali untuk tujuan lain. Pastikan MFA perangkat keras yang ditenagai oleh baterai diganti secara teratur. (CIS1,6)

   - MFAUntuk mengkonfigurasi pengguna root, ikuti instruksi untuk membuat perangkat virtual MFA atau MFAperangkat keras.

d. Pertimbangkan untuk mendaftarkan beberapa MFA perangkat untuk cadangan. Hingga 8 MFA perangkat diperbolehkan per akun.

   - Perhatikan bahwa mendaftarkan lebih dari satu MFA perangkat untuk pengguna root secara otomatis mematikan alur untuk memulihkan akun Anda jika MFA perangkat hilang.

e. Simpan kata sandi dengan aman, dan pertimbangkan dependensi melingkar jika menyimpan kata sandi secara elektronik. Jangan menyimpan kata sandi sedemikian rupa sehingga memerlukan akses ke kata sandi yang sama Akun AWS untuk mendapatkannya.

6. Opsional: Coba terapkan jadwal rotasi kata sandi untuk pengguna root secara berkala.

   - Praktik terbaik manajemen kredensial bergantung pada persyaratan peraturan dan kebijakan Anda. Pengguna root MFA yang dilindungi oleh tidak bergantung pada kata sandi sebagai faktor tunggal otentikasi.
   - Mengubah kata sandi pengguna root secara berkala mengurangi risiko bahwa kata sandi yang diketahui orang lain secara tidak sengaja dapat disalahgunakan.

Kontrol detektif

- Buat alarm untuk mendeteksi penggunaan kredensi root (CIS1.7). Amazon GuardDuty dapat memantau dan memperingatkan penggunaan API kredensi pengguna root melalui RootCredentialUsagetemuan.
- Mengevaluasi dan menerapkan kontrol detektif yang termasuk dalam paket kesesuaian Pilar AWS Keamanan yang Dirancang dengan Baik AWS Config untuk, atau jika AWS Control Tower menggunakan, kontrol yang sangat disarankan tersedia di dalam Control Tower.

Panduan operasional

- Tentukan siapa di organisasi Anda yang harus memiliki akses ke kredensial pengguna root.
  - Gunakan aturan dua orang sehingga tidak ada satu orang pun yang memiliki akses ke semua kredensi yang diperlukan dan MFA untuk mendapatkan akses pengguna root.

  - Verifikasi bahwa organisasi, dan bukan satu individu pun, mempertahankan kontrol atas nomor telepon dan alias email yang terkait dengan akun (yang digunakan untuk mengatur ulang kata sandi dan alur MFA ulang).
- Gunakan pengguna root hanya dengan pengecualian (CIS1.7).
  - Pengguna AWS root tidak boleh digunakan untuk tugas sehari-hari, bahkan yang administratif. Hanya masuk sebagai pengguna root untuk melakukan tugas-tugas AWS yang membutuhkan pengguna root. Semua tindakan lainnya harus dilakukan oleh pengguna lain dengan peran yang sesuai.
- Periksa secara berkala apakah akses ke pengguna root berfungsi dengan baik sehingga prosedurnya telah teruji sebelum terjadi situasi darurat yang memerlukan penggunaan kredensial pengguna root.
- Periksa secara berkala apakah alamat email yang terkait dengan akun dan yang tercantum di bawah Kontak Alternatif berfungsi. Pantau kotak masuk email untuk melihat apakah ada notifikasi keamanan yang Anda terima dari `<abuse@amazon.com>`. Selain itu, pastikan semua nomor telepon yang terkait dengan akun saat ini berfungsi dengan baik.
- Siapkan prosedur respons insiden untuk merespons penyalahgunaan akun root. Lihat Panduan Respons Insiden Keamanan AWS dan praktik-praktik terbaik di bagian Respons Insiden di laporan resmi Pilar Keamanan untuk mendapatkan informasi lebih lanjut tentang cara membangun strategi respons insiden untuk Akun AWS Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP01 Memisahkan beban kerja menggunakan akun
- SEC02-BP01 Gunakan mekanisme masuk yang kuat
- SEC03-BP02 Berikan akses hak istimewa paling sedikit
- SEC03-BP03 Menetapkan proses akses darurat
- SEC10-BP05 Akses pra-penyediaan

Dokumen terkait:

- AWS Control Tower
- Pedoman Audit Keamanan AWS

- IAMPraktik Terbaik
- Amazon GuardDuty - peringatan penggunaan kredensi root
- tep-by-stepPanduan S tentang pemantauan untuk penggunaan kredensi root melalui CloudTrail
- MFAtoken disetujui untuk digunakan dengan AWS
- Menerapkan akses break glass pada AWS
- 10 item keamanan teratas untuk ditingkatkan di Anda Akun AWS
- Apa yang harus saya lakukan jika saya melihat aktivitas tanpa izin dalam Akun AWS saya?

Video terkait:

- Aktifkan adopsi AWS dalam skala besar dengan menggunakan otomatisasi dan tata kelola
- Praktik Terbaik Keamanan dengan Cara Well-Architected
- Membatasi penggunaan kredensil AWS root dari AWS re:inforce 2022 — Praktik terbaik keamanan dengan AWS IAM

Contoh dan laboratorium terkait:

- Lab: Akun AWS pengaturan dan pengguna root

SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan pengendalian

Berdasarkan persyaratan kepatuhan dan risiko yang diidentifikasi dari model ancaman Anda, dapatkan dan validasikan kontrol dan tujuan kontrol yang perlu Anda terapkan pada beban kerja Anda. Validasi berkelanjutan terhadap kontrol dan tujuan kontrol dapat membantu Anda mengukur efektivitas mitigasi risiko.

Hasil yang diinginkan: Tujuan dari kontrol keamanan terhadap bisnis Anda didefinisikan dengan baik dan selaras dengan persyaratan kepatuhan Anda. Kontrol diimplementasikan dan diberlakukan melalui otomatisasi dan kebijakan serta terus dievaluasi untuk mengetahui efektivitasnya dalam mencapai tujuan Anda. Bukti efektivitas pada suatu titik waktu dan selama periode waktu tertentu dapat mudah dilaporkan kepada auditor.

Anti-pola umum:

- Persyaratan peraturan, ekspektasi pasar, dan standar industri untuk keamanan yang dapat dijamin belum dipahami dengan baik untuk bisnis Anda

- Kerangka kerja keamanan siber dan tujuan kontrol Anda tidak selaras dengan persyaratan bisnis Anda
- Implementasi kontrol tidak sepenuhnya selaras dengan tujuan kontrol Anda secara terukur
- Anda tidak menggunakan otomatisasi untuk melaporkan efektivitas kontrol Anda

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Ada banyak kerangka kerja keamanan siber umum yang bisa menjadi dasar untuk tujuan kontrol keamanan Anda. Pertimbangkan persyaratan peraturan, ekspektasi pasar, dan standar industri untuk bisnis Anda guna menentukan kerangka kerja mana yang paling memenuhi kebutuhan Anda. Contohnya termasuk AICPASOC2,, PCI- HITRUST, ISO27001 DSS, dan NISTSP 800-53.

Untuk tujuan kontrol yang Anda identifikasi, pahami bagaimana AWS layanan yang Anda konsumsi membantu Anda mencapai tujuan tersebut. Gunakan AWS Artifactuntuk menemukan dokumentasi dan laporan yang selaras dengan kerangka kerja target Anda yang menjelaskan ruang lingkup tanggung jawab yang dicakup oleh AWS dan panduan untuk ruang lingkup yang tersisa yang menjadi tanggung jawab Anda. Untuk panduan khusus layanan lebih lanjut saat selaras dengan berbagai pernyataan kontrol kerangka kerja, lihat Panduan Kepatuhan Pelanggan AWS.

Saat Anda menentukan kontrol yang memenuhi tujuan Anda, lakukan kodifikasi pemberlakuan menggunakan kontrol preventif, dan lakukan otomatisasi mitigasi dengan menggunakan kontrol-kontrol detektif. Bantu mencegah konfigurasi dan tindakan sumber daya yang tidak sesuai di seluruh kebijakan kontrol layanan yang Anda AWS Organizations gunakan (). SCP Terapkan aturan AWS Configuntuk memantau dan melaporkan sumber daya yang tidak sesuai, kemudian beralih aturan ke model penegakan setelah yakin dengan perilakunya. Untuk menerapkan set aturan yang telah ditentukan dan dikelola yang selaras dengan kerangka kerja keamanan siber Anda, evaluasi penggunaan standar AWS Security Hub sebagai opsi pertama Anda. Standar AWS Foundational Service Best Practices (FSBP) dan CIS AWS Foundations Benchmark adalah titik awal yang baik dengan kontrol yang selaras dengan banyak tujuan yang dibagi di beberapa kerangka kerja standar. Jika Security Hub tidak secara intrinsik memiliki deteksi kontrol yang diinginkan, maka dapat dilengkapi dengan menggunakan paket kesesuaian.AWS Config

Gunakan Paket APN Mitra yang direkomendasikan oleh tim AWS Global Security and Compliance Acceleration (GSCA) untuk mendapatkan bantuan dari penasihat keamanan, agen konsultan, sistem pengumpulan dan pelaporan bukti, auditor, dan layanan pelengkap lainnya bila diperlukan.

Langkah-langkah implementasi

1. Evaluasi kerangka kerja keamanan siber umum, dan selaraskan tujuan kontrol Anda dengan kerangka kerja yang dipilih.
2. Dapatkan dokumentasi yang relevan tentang panduan dan tanggung jawab untuk menggunakan kerangka kerja Anda AWS Artifact. Pahami bagian kepatuhan mana yang berada di AWS sisi model tanggung jawab bersama dan bagian mana yang menjadi tanggung jawab Anda.
3. PenggunaanSCPs, kebijakan sumber daya, kebijakan kepercayaan peran, dan pagar pembatas lainnya untuk mencegah konfigurasi dan tindakan sumber daya yang tidak sesuai.
4. Evaluasi penerapan standar Security Hub dan paket AWS Config kesesuaian yang sesuai dengan tujuan kontrol Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC03-BP01 Tentukan persyaratan akses
- SEC04-BP01 Mengkonfigurasi layanan dan pencatatan aplikasi
- SEC07-BP01 Memahami skema klasifikasi data Anda
- OPS01-BP03 Mengevaluasi persyaratan tata kelola
- OPS01-BP04 Mengevaluasi persyaratan kepatuhan
- PERF01-BP05 Menggunakan kebijakan dan arsitektur referensi
- COST02-BP01 Mengembangkan kebijakan berdasarkan kebutuhan organisasi Anda

Dokumen terkait:

- Panduan Kepatuhan Pelanggan AWS

Alat terkait:

- AWS Artifact

SEC01-BP04 Tetap up to date dengan ancaman dan rekomendasi keamanan

Terus ikuti perkembangan ancaman dan mitigasi terbaru dengan memantau publikasi intelijen dan umpan data ancaman industri untuk mendapatkan pemberitahuan. Evaluasi penawaran layanan terkelola yang diperbarui secara otomatis berdasarkan data ancaman terbaru.

Hasil yang diinginkan: Anda tetap mendapat informasi karena publikasi industri diperbarui dengan ancaman dan rekomendasi terbaru. Anda menggunakan otomatisasi untuk mendeteksi potensi kerentanan dan paparan saat Anda mengidentifikasi ancaman baru. Anda mengambil tindakan mitigasi terhadap ancaman tersebut. Anda mengadopsi AWS layanan yang secara otomatis diperbarui dengan intelijen ancaman terbaru.

Anti-pola umum:

- Tidak memiliki mekanisme yang andal dan dapat diulangi untuk terus mendapatkan informasi tentang intelijen ancaman terbaru.
- Memelihara inventaris manual berisi portofolio teknologi, beban kerja, dan dependensi Anda yang memerlukan peninjauan oleh manusia untuk menemukan potensi kerentanan dan paparan.
- Tidak memiliki mekanisme untuk memperbarui beban kerja dan dependensi Anda ke versi terkini yang tersedia yang memberikan mitigasi ancaman yang diketahui.

Manfaat menerapkan praktik terbaik ini: Menggunakan sumber intelijen ancaman untuk tetap up to date akan mengurangi risiko kehilangan perubahan penting pada lanskap ancaman yang dapat memengaruhi bisnis Anda. Penerapan otomatisasi untuk melakukan pemindaian, deteksi, dan remediasi jika ada potensi kerentanan atau paparan dalam beban kerja Anda serta dependensinya dapat membantu Anda memitigasi risiko secara cepat dan terprediksi, dibandingkan dengan alternatif manual. Hal ini membantu mengontrol waktu dan biaya yang terkait dengan mitigasi kerentanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Tinjau publikasi intelijen ancaman tepercaya untuk terus mengikuti perkembangan lanskap ancaman. Konsultasikan basis pengetahuan MITREATT&CK untuk dokumentasi tentang taktik, teknik, dan prosedur permusuhan yang diketahui (). TTPs Daftar Common Vulnerabilities and Exposures (CVE) untuk tetap mendapat informasi tentang kerentanan yang diketahui dalam produk yang Anda andalkan. MITRE Memahami risiko kritis terhadap aplikasi web dengan proyek OWASPTop 10 Open Worldwide Application Security Project (OWASP) yang populer.

Tetap up to date tentang peristiwa AWS keamanan dan langkah-langkah perbaikan yang direkomendasikan dengan Buletin AWS Keamanan untuk. CVEs

Untuk mengurangi upaya dan overhead Anda secara keseluruhan untuk tetap up to date, pertimbangkan untuk menggunakan AWS layanan yang secara otomatis menggabungkan intelijen ancaman baru dari waktu ke waktu. Misalnya, Amazon GuardDuty tetap up to date dengan intelijen ancaman industri untuk mendeteksi perilaku anomali dan tanda tangan ancaman dalam akun Anda. Amazon Inspector secara otomatis menyimpan database yang CVEs digunakannya untuk fitur pemindaian berkelanjutan yang diperbarui. Baik AWS WAF maupun AWS Shield Advanced menyediakan grup aturan terkelola yang diperbarui secara otomatis saat ancaman baru muncul.

Kaji pilar keunggulan operasional Well-Architected untuk manajemen dan penambalan armada otomatis.

Langkah-langkah implementasi

- Berlangganlah ke pemberitahuan untuk publikasi intelijen ancaman yang relevan dengan bisnis dan industri Anda. Berlangganlah ke Buletin Keamanan AWS .
- Pertimbangkan untuk mengadopsi layanan yang menggabungkan intelijen ancaman baru secara otomatis, seperti Amazon GuardDuty dan Amazon Inspector.
- Lakukan deployment strategi manajemen dan patching armada yang selaras dengan praktik terbaik Pilar Keunggulan Operasional Well-Architected.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP07 Mengidentifikasi ancaman dan memprioritaskan mitigasi menggunakan model ancaman
- OPS01-BP05 Mengevaluasi lanskap ancaman
- OPS11-BP01 Memiliki proses untuk perbaikan berkelanjutan

SEC01-BP05 Mengurangi ruang lingkup manajemen keamanan

Tentukan apakah Anda dapat mengurangi cakupan keamanan dengan menggunakan AWS layanan yang mengalihkan manajemen kontrol tertentu ke AWS (layanan terkelola). Layanan ini dapat membantu mengurangi tugas pemeliharaan keamanan Anda, seperti penyediaan infrastruktur, penyiapan perangkat lunak, patching, atau pencadangan.

Hasil yang diinginkan: Anda mempertimbangkan ruang lingkup manajemen keamanan Anda saat memilih AWS layanan untuk beban kerja Anda. Biaya overhead manajemen dan tugas pemeliharaan (total biaya kepemilikan, atauTCO) ditimbang terhadap biaya layanan yang Anda pilih, di samping pertimbangan Well-Architected lainnya. Anda memasukkan dokumentasi AWS kontrol dan kepatuhan ke dalam evaluasi kontrol dan prosedur verifikasi Anda.

Anti-pola umum:

- Melakukan deployment beban kerja tanpa sepenuhnya memahami model tanggung jawab bersama untuk layanan yang Anda pilih.
- Meng-host basis data dan teknologi lainnya pada mesin virtual tanpa mengevaluasi sarana yang setara dengan layanan terkelola.
- Tidak menyertakan tugas manajemen keamanan ke dalam total biaya kepemilikan untuk meng-host teknologi pada mesin virtual jika dibandingkan dengan opsi layanan terkelola.

Manfaat menerapkan praktik terbaik ini: Menggunakan layanan terkelola akan dapat mengurangi beban keseluruhan Anda dalam mengelola kontrol keamanan operasional, hal ini dapat mengurangi risiko keamanan dan total biaya kepemilikan Anda. Waktu yang seharusnya dihabiskan untuk tugas-tugas keamanan tertentu dapat diinvestasikan kembali ke tugas-tugas yang memberikan nilai lebih bagi bisnis Anda. Layanan terkelola juga dapat mengurangi cakupan persyaratan kepatuhan Anda dengan mengalihkan sebagian persyaratan kontrol ke AWS.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Ada beberapa cara untuk mengintegrasikan komponen beban kerja Anda di AWS. Menginstal dan menjalankan teknologi di EC2 instans Amazon sering kali mengharuskan Anda untuk mengambil bagian terbesar dari tanggung jawab keamanan secara keseluruhan. Untuk membantu mengurangi beban pengoperasian kontrol tertentu, identifikasi layanan AWS terkelola yang mengurangi ruang lingkup sisi Anda dari model tanggung jawab bersama dan pahami bagaimana Anda dapat menggunakannya dalam arsitektur yang ada. Contohnya termasuk menggunakan Amazon Relational Database Service (Amazon) untuk menyebarkan database, RDS AmazonElastic Kubernetes Service (Amazon) atau Amazon Elastic Container Service EKS (Amazon) untuk mengatur kontainerECS, atau menggunakan opsi tanpa server. Saat membuat aplikasi baru, pikirkan layanan mana yang dapat membantu mengurangi waktu dan biaya dalam mengimplementasikan dan mengelola kontrol keamanan.

Persyaratan kepatuhan juga dapat menjadi faktor ketika memilih layanan. Layanan yang dikelola dapat menggeser kepatuhan beberapa persyaratan ke AWS. Diskusikan dengan tim kepatuhan Anda tentang tingkat kenyamanan mereka dengan mengaudit aspek layanan yang Anda operasikan dan mengelola serta menerima pernyataan kontrol dalam laporan AWS audit yang relevan. Anda dapat memberikan artefak audit yang ditemukan di AWS Artifactauditor atau regulator Anda sebagai bukti kontrol keamanan. AWS Anda juga dapat menggunakan panduan tanggung jawab yang disediakan oleh beberapa artefak AWS audit untuk merancang arsitektur Anda, bersama dengan Panduan Kepatuhan AWS Pelanggan. Panduan ini membantu menentukan kontrol keamanan tambahan yang harus Anda terapkan untuk mendukung kasus penggunaan spesifik di sistem Anda.

Saat menggunakan layanan terkelola, biasakan proses memperbarui sumber daya mereka ke versi yang lebih baru (misalnya, memperbarui versi database yang dikelola oleh AmazonRDS, atau runtime bahasa pemrograman untuk suatu AWS Lambda fungsi). Meskipun layanan terkelola dapat melakukan operasi ini untuk Anda, Anda tetap bertanggung jawab untuk mengonfigurasi waktu pembaruan dan memahami dampaknya pada operasi Anda. Alat seperti AWS Health dapat membantu Anda melacak dan mengelola pembaruan ini di seluruh lingkungan Anda.

Langkah-langkah implementasi

1. Evaluasi komponen beban kerja Anda yang dapat diganti dengan layanan terkelola.
    a. Jika Anda memigrasikan beban kerja ke AWS, pertimbangkan pengurangan manajemen (waktu dan biaya) dan pengurangan risiko saat Anda menilai apakah Anda harus meng-host ulang, memfaktorkan ulang, memplatform ulang, membangun kembali, atau mengganti beban kerja Anda. Terkadang, investasi tambahan pada awal migrasi dapat memiliki penghematan yang signifikan dalam jangka panjang.
2. Pertimbangkan untuk menerapkan layanan terkelolaRDS, seperti Amazon, alih-alih menginstal dan mengelola penyebaran teknologi Anda sendiri.
3. Gunakan panduan tanggung jawab AWS Artifact untuk membantu menentukan kontrol keamanan yang harus Anda lakukan untuk beban kerja Anda.
4. Simpan inventaris sumber daya yang digunakan, dan tetap up-to-date dengan layanan dan pendekatan baru untuk mengidentifikasi peluang baru untuk mengurangi ruang lingkup.

Sumber daya

Praktik-praktik terbaik terkait:

- PERF02-BP01 Pilih opsi komputasi terbaik untuk beban kerja Anda

- PERF03-BP01 Gunakan penyimpanan data yang dibuat khusus yang paling mendukung persyaratan akses dan penyimpanan data Anda
- SUS05-BP03 Gunakan layanan terkelola

Dokumen terkait:

- Acara siklus hidup yang direncanakan untuk AWS Health

Alat terkait:

- AWS Health
- AWS Artifact
- Panduan Kepatuhan Pelanggan AWS

Video terkait:

- Bagaimana cara saya bermigrasi ke instans Amazon RDS atau Aurora SQL My DB menggunakan? AWS DMS
- AWS re:invent 2023 - Kelola acara siklus hidup sumber daya dalam skala besar dengan AWS Health

SEC01-BP06 Mengotomatiskan penerapan kontrol keamanan standar

Terapkan DevOps praktik modern saat Anda mengembangkan dan menerapkan kontrol keamanan yang standar di seluruh AWS lingkungan Anda. Tentukan kontrol dan konfigurasi keamanan standar menggunakan templat Infrastructure as Code (IAc), tangkap perubahan dalam sistem kontrol versi, uji perubahan sebagai bagian dari pipeline CI/CD, dan otomatiskan penerapan perubahan ke lingkungan Anda. AWS

Hasil yang diinginkan: Template IAC menangkap kontrol keamanan standar dan memasukkannya ke sistem kontrol versi. Pipeline CI/CD berada di tempat yang mendeteksi perubahan dan mengotomatiskan pengujian dan penerapan lingkungan Anda. AWS Pagar pembatas diterapkan untuk mendeteksi dan memperingatkan kesalahan konfigurasi dalam templat sebelum melanjutkan ke deployment. Beban kerja di-deploy ke lingkungan yang menerapkan kontrol standar. Tim memiliki akses untuk melakukan deployment konfigurasi layanan yang disetujui melalui mekanisme mandiri.

Strategi pencadangan dan pemulihan yang aman diterapkan untuk konfigurasi kontrol, skrip, dan data terkait.

Anti-pola umum:

- Membuat perubahan pada kontrol keamanan standar Anda secara manual, melalui konsol web atau antarmuka baris perintah.
- Mengandalkan tim beban kerja individual untuk secara manual mengimplementasikan kontrol yang ditentukan oleh tim pusat.
- Mengandalkan tim keamanan pusat untuk melakukan deployment kontrol tingkat beban kerja atas permintaan tim beban kerja.
- Mengizinkan individu atau tim yang sama untuk mengembangkan, menguji, dan melakukan deployment skrip otomatisasi kontrol keamanan tanpa pemisahan tugas atau pemeriksaan dan keseimbangan yang tepat.

Manfaat menerapkan praktik terbaik ini: Menggunakan templat untuk menentukan kontrol keamanan standar Anda akan memungkinkan Anda untuk melacak dan membandingkan perubahan dari waktu ke waktu dengan menggunakan sistem kontrol versi. Penggunaan otomatisasi untuk menguji dan melakukan deployment perubahan akan menghasilkan standardisasi dan prediktabilitas, sehingga meningkatkan peluang deployment yang berhasil dan mengurangi tugas manual yang berulang. Penyediaan mekanisme mandiri bagi tim beban kerja untuk melakukan deployment layanan dan konfigurasi yang disetujui akan mengurangi risiko kesalahan konfigurasi dan kesalahan penggunaan. Hal ini juga membantu mereka memasukkan kontrol lebih awal dalam proses pengembangan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Saat mengikuti praktik yang dijelaskan dalam SEC01-BP01 Beban kerja terpisah menggunakan akun, Anda akan mendapatkan beberapa Akun AWS untuk lingkungan berbeda yang Anda kelola gunakan. AWS Organizations Meskipun masing-masing lingkungan dan beban kerja ini mungkin memerlukan kontrol keamanan yang berbeda, Anda dapat menstandardisasi beberapa kontrol keamanan di seluruh organisasi Anda. Contohnya termasuk mengintegrasikan penyedia identitas tersentralisasi, menentukan jaringan dan firewall, serta mengonfigurasi lokasi standar untuk menyimpan dan menganalisis log. Dengan cara yang sama Anda dapat menggunakan infrastruktur sebagai kode (IAc) untuk menerapkan ketelitian pengembangan kode aplikasi yang sama untuk penyediaan infrastruktur, Anda dapat menggunakan IAc untuk menentukan dan menerapkan kontrol keamanan standar Anda juga.

Jika memungkinkan, tentukan kontrol keamanan Anda dengan cara deklaratif, seperti di AWS CloudFormation, dan simpan dalam sistem kontrol sumber. Gunakan DevOps praktik untuk mengotomatiskan penerapan kontrol Anda untuk rilis yang lebih dapat diprediksi, pengujian otomatis menggunakan alat seperti AWS CloudFormation Guard, dan mendeteksi penyimpangan antara kontrol yang diterapkan dan konfigurasi yang Anda inginkan. Anda dapat menggunakan layanan seperti AWS CodePipeline, AWS CodeBuild, dan AWS CodeDeploy untuk membangun konsep pipeline CI/CD. Pertimbangkan panduan dalam Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun untuk mengonfigurasi layanan ini di akun mereka sendiri yang terpisah dari pipeline penerapan lainnya.

Anda juga dapat menentukan template untuk menstandarisasi pendefinisian dan penerapan Akun AWS, layanan, dan konfigurasi. Teknik ini memungkinkan tim keamanan pusat mengelola penentuan ini dan menyediakannya kepada tim beban kerja melalui pendekatan mandiri. Salah satu cara untuk mencapai hal ini adalah dengan menggunakan Katalog Layanan, di mana Anda dapat memublikasikan templat sebagai produk yang dapat digabungkan oleh tim beban kerja ke dalam deployment pipeline mereka sendiri. Jika Anda menggunakan AWS Control Tower, beberapa templat dan kontrol tersedia sebagai titik awal. Control Tower juga menyediakan kemampuan Account Factory, yang memungkinkan tim beban kerja untuk membuat Akun AWS yang baru menggunakan standar yang Anda tentukan. Kemampuan ini membantu meniadakan dependensi pada tim pusat untuk menyetujui dan membuat akun baru ketika akun tersebut diperlukan oleh tim beban kerja Anda. Anda mungkin memerlukan akun ini untuk mengisolasi komponen beban kerja yang berbeda-beda berdasarkan berbagai alasan, seperti fungsi yang diberikan, sensitivitas data yang diproses, atau perilakunya.

Langkah-langkah implementasi

1. Tentukan cara Anda akan menyimpan dan memelihara templat Anda dalam sebuah sistem kontrol versi.
2. Buat pipeline CI/CD untuk menguji dan menerapkan templat Anda. Tentukan pengujian untuk memeriksa adanya kesalahan konfigurasi dan apakah templat tersebut mematuhi standar perusahaan Anda, atau tidak.
3. Buat katalog templat standar untuk tim beban kerja untuk diterapkan Akun AWS dan layanan sesuai dengan kebutuhan Anda.
4. Implementasikan strategi pencadangan dan pemulihan yang aman untuk konfigurasi kontrol, skrip, dan data terkait Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP01 Gunakan kontrol versi
- OPS05-BP04 Gunakan sistem manajemen build dan deployment
- REL08-BP05 Menyebarkan perubahan dengan otomatisasi
- SUS06-BP01 Mengadopsi metode yang dapat dengan cepat memperkenalkan peningkatan keberlanjutan

Dokumen terkait:

- Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun

Contoh terkait:

- Mengotomatiskan pembuatan akun, dan penyediaan sumber daya menggunakan Service Catalog,, dan AWS OrganizationsAWS Lambda
- Memperkuat DevOps pipa dan melindungi data dengan AWS Secrets Manager, AWS KMS, dan AWS Certificate Manager

Alat terkait:

- AWS CloudFormation Guard
- Akselerator Zona Pendaratan aktif AWS

SEC01-BP07 Mengidentifikasi ancaman dan memprioritaskan mitigasi menggunakan model ancaman

Lakukan pemodelan ancaman untuk mengidentifikasi dan memelihara up-to-date daftar potensi ancaman dan mitigasi terkait untuk beban kerja Anda. Tentukan prioritas ancaman dan sesuaikan mitigasi kontrol keamanan Anda untuk mencegah, mendeteksi, dan merespons ancaman. Anda harus memeriksa kembali dan mempertahankan hal ini dengan mempertimbangkan beban kerja Anda, serta lanskap keamanan yang terus berubah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Apa itu pemodelan ancaman?

“Pemodelan ancaman bekerja untuk mengidentifikasi, berkomunikasi, dan memahami ancaman dan mitigasi dalam konteks melindungi sesuatu yang bernilai.” — Proyek Keamanan Aplikasi Web Terbuka (OWASP) Pemodelan Ancaman Aplikasi

Mengapa Anda harus menjadi model ancaman?

Sistem begitu kompleks. Kompleksitas dan kemampuannya akan makin meningkat seiring waktu, sehingga memberikan nilai bisnis yang lebih banyak dan meningkatkan keterlibatan serta kepuasan pelanggan. Artinya, keputusan desain IT perlu mempertimbangkan peningkatan jumlah kasus penggunaan. Perubahan kompleksitas dan jumlah kasus penggunaan ini biasanya menjadikan pendekatan tidak terstruktur tidak efektif untuk menemukan temuan tentang ancaman dan memitigasinya. Maka, Anda memerlukan pendekatan sistematis untuk menghitung potensi ancaman terhadap sistem, serta untuk melakukan dan memprioritaskan mitigasi untuk memastikan organisasi Anda dapat meningkatkan postur keamanan sistem secara keseluruhan dengan maksimal meski dengan sumber daya terbatas.

Pemodelan ancaman dirancang untuk memberikan pendekatan sistematis ini, yang bertujuan untuk menemukan temuan tentang masalah dan mengatasi masalah tersebut dalam proses desain lebih awal, saat biaya dan upaya mitigasi relatif rendah dibandingkan setelahnya dalam siklus hidup. Pendekatan ini sejalan dengan prinsip industri keamanan shift-left. Pada intinya, pemodelan ancaman akan terintegrasi dengan proses manajemen risiko organisasi serta membantu menentukan kontrol mana yang akan diimplementasikan menggunakan pendekatan berbasis ancaman.

Kapan pemodelan ancaman harus dilakukan?

Mulai pemodelan ancaman dalam siklus hidup beban kerja Anda sedini mungkin. Hal ini membuat Anda lebih fleksibel dalam menentukan tindakan untuk mengatasi ancaman yang teridentifikasi. Seperti halnya bug perangkat lunak, makin dini Anda mengidentifikasi ancaman, makin hemat biaya penanganannya. Model ancaman adalah dokumen hidup (living document) dan akan terus berkembang seiring perubahan beban kerja Anda. Tinjau kembali untuk mempertahankan model ancaman Anda dari waktu ke waktu, termasuk saat terjadi perubahan besar, perubahan dalam lanskap ancaman, atau saat Anda mengadopsi fitur atau layanan baru.

Langkah-langkah implementasi

Bagaimana kita bisa melakukan pemodelan ancaman?

Pemodelan ancaman bisa dijalankan dengan banyak cara. Seperti halnya bahasa pemrograman, setiap model memiliki kelebihan dan kekurangannya masing-masing. Pilih cara yang paling tepat untuk Anda. Salah satu pendekatannya adalah memulai dengan Shostack's 4 Question Frame for Threat Modeling, yang mengajukan pertanyaan terbuka untuk memberikan struktur pada latihan pemodelan ancaman Anda:

1. Apa yang sedang dikerjakan?

   Pertanyaan ini bertujuan untuk membantu memahami dan menentukan sistem yang sedang Anda bangun serta detail sistem tersebut yang relevan dengan keamanan. Membuat model atau diagram adalah cara paling populer untuk menjawab pertanyaan ini, karena membantu Anda memvisualisasikan apa yang Anda bangun, misalnya, menggunakan diagram alir data. Menuliskan asumsi dan detail penting tentang sistem Anda juga dapat membantu Anda menentukan cakupan. Ini memungkinkan semua orang yang berkontribusi pada model ancaman untuk fokus pada hal yang sama, dan menghindari jalan memutar yang memakan waktu ke dalam out-of-scope topik (termasuk versi sistem Anda yang kedaluwarsa). Misalnya, jika Anda sedang membuat aplikasi web, sepertinya tidak layak untuk membuang waktu melakukan pemodelan ancaman untuk urutan boot tepercaya pada sistem operasi untuk klien browser karena desain Anda tidak akan dapat memengaruhi hal ini.
2. Apa yang bisa salah?

   Di sini Anda dapat mengidentifikasi ancaman terhadap sistem Anda. Ancaman adalah tindakan atau peristiwa yang disengaja atau tidak disengaja, yang dampaknya tidak diharapkan dan dapat memengaruhi keamanan sistem Anda. Tanpa mengetahui dengan jelas apa saja potensi permasalahannya, Anda tidak akan tahu cara penanganannya.

   Tidak ada daftar khusus tentang apa saja masalah yang dapat terjadi. Membuat daftar ini membutuhkan curah pendapat dan kolaborasi antara semua individu dalam tim Anda dan persona relevan yang terlibat dalam latihan pemodelan ancaman. Anda dapat membantu brainstorming Anda dengan menggunakan model untuk mengidentifikasi ancaman, seperti, yang menyarankan berbagai kategori untuk dievaluasi: Spoofing STRIDE, Tampering, Repudiation, Information Disclosure, Denial of Service, dan Elevation of privilege. Selain itu, Anda mungkin ingin membantu brainstorming dengan meninjau daftar dan penelitian yang ada untuk mendapatkan inspirasi, termasuk 10 OWASP Teratas, Katalog Ancaman, dan katalog HiTrust ancaman organisasi Anda sendiri.
3. Apa yang akan kita lakukan tentang hal itu?

Sama seperti pertanyaan sebelumnya, tidak ada daftar khusus untuk semua kemungkinan mitigasi. Input dalam langkah ini adalah ancaman yang teridentifikasi, pelaku, dan area peningkatan dari langkah sebelumnya.

Keamanan dan kepatuhan merupakan tanggung jawab bersama antara Anda dan AWS. Perlu dipahami bahwa pertanyaan “Tindakan apa yang akan kita lakukan?” harus disertai dengan pertanyaan “Siapa yang bertanggung untuk melakukan hal ini?”. Memahami keseimbangan tanggung jawab antara Anda dan AWS membantu Anda menjangkau latihan pemodelan ancaman Anda dengan mitigasi yang berada di bawah kendali Anda, yang biasanya merupakan kombinasi opsi konfigurasi AWS layanan dan mitigasi spesifik sistem Anda sendiri.

Untuk AWS bagian dari tanggung jawab bersama, Anda akan menemukan bahwa AWS layanan berada dalam lingkup banyak program kepatuhan. Program-program ini membantu Anda memahami kontrol yang kuat AWS untuk menjaga keamanan dan kepatuhan cloud. Laporan audit dari program ini tersedia untuk diunduh bagi AWS pelanggan AWS Artifact.

Terlepas dari AWS layanan mana yang Anda gunakan, selalu ada elemen tanggung jawab pelanggan, dan mitigasi yang selaras dengan tanggung jawab ini harus dimasukkan dalam model ancaman Anda. Untuk mitigasi kontrol keamanan untuk AWS layanan itu sendiri, Anda ingin mempertimbangkan untuk menerapkan kontrol keamanan di seluruh domain, termasuk domain seperti identitas dan manajemen akses (otentikasi dan otorisasi), perlindungan data (saat istirahat dan dalam perjalanan), keamanan infrastruktur, pencatatan, dan pemantauan. Dokumentasi untuk setiap AWS layanan memiliki bagian keamanan khusus yang memberikan panduan tentang kontrol keamanan untuk dipertimbangkan sebagai mitigasi. Pertimbangkan kode yang Anda tulis dan dependensi kodenya karena hal ini sangat penting, serta tentukan kontrol yang dapat Anda terapkan untuk mengatasi ancaman. Kontrol ini bisa berupa hal-hal seperti validasi input, penanganan sesi, dan dan penanganan batas. Fokus pada kode kustom karena sebagian besar kerentanan seringnya terjadi di area ini.

4. Apakah kita melakukan pekerjaan dengan baik?

Tujuannya adalah agar tim dan organisasi Anda dapat meningkatkan kualitas model ancaman dan kecepatan dalam melakukan pemodelan ancaman dari waktu ke waktu. Peningkatan ini adalah hasil dari gabungan praktik, pembelajaran, pengajaran, dan peninjauan. Untuk membahas lebih dalam dan langsung, disarankan agar Anda dan tim Anda menyelesaikan Pemodelan ancaman dengan cara yang tepat untuk kursus pelatihan pembangun atau lokakarya. Selain itu, jika Anda mencari panduan tentang cara mengintegrasikan pemodelan ancaman ke dalam siklus

pengembangan aplikasi organisasi Anda, lihat Cara mendekati pos pemodelan ancaman di Blog AWS Keamanan.

Komposer Ancaman

Untuk membantu dan memandu Anda dalam melakukan pemodelan ancaman, pertimbangkan untuk menggunakan alat Threat Composer, yang bertujuan untuk mengurangi Anda time-to-value saat pemodelan ancaman. Alat ini membantu Anda melakukan hal berikut:

- Tulis pernyataan ancaman yang berguna yang selaras dengan tata bahasa ancaman yang bekerja dalam suatu alur kerja non-linier alami
- Menghasilkan model ancaman yang dapat dibaca manusia
- Membuat model ancaman yang dapat dibaca mesin untuk memungkinkan Anda memperlakukan model ancaman sebagai kode
- Membantu Anda mengidentifikasi area peningkatan kualitas dan cakupan dengan cepat menggunakan Dasbor Wawasan

Untuk referensi lebih lanjut, silakan kunjungi Komposer Ancaman dan beralih ke Contoh Ruang Kerja yang ditentukan sistem.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan pengendalian
- SEC01-BP04 Tetap up to date dengan ancaman dan rekomendasi keamanan
- SEC01-BP05 Mengurangi ruang lingkup manajemen keamanan
- SEC01-BP08 Mengevaluasi dan menerapkan layanan dan fitur keamanan baru secara teratur

Dokumen terkait:

- Cara mendekati pemodelan ancaman (Blog AWS Keamanan)
- NIST: Panduan untuk Pemodelan Ancaman Sistem Data-Centric

Video terkait:

- AWS Summit ANZ 2021 - Cara mendekati pemodelan ancaman
- AWS Summit ANZ 2022 - Keamanan penskalaan — Optimalkan untuk pengiriman yang cepat dan aman

Pelatihan terkait:

- Pemodelan ancaman dengan cara yang tepat untuk pembangun — AWS Pelatihan mandiri virtual Skill Builder
- Pemodelan ancaman dengan cara yang tepat untuk pembangun — AWS Workshop

Alat terkait:

- Komposer Ancaman

SEC01-BP08 Mengevaluasi dan menerapkan layanan dan fitur keamanan baru secara teratur

Mengevaluasi dan menerapkan layanan dan fitur keamanan dari AWS dan AWS Mitra yang membantu Anda mengembangkan postur keamanan beban kerja Anda.

Hasil yang diinginkan: Anda memiliki praktik standar yang memberi tahu Anda tentang fitur dan layanan baru yang dirilis oleh AWS dan AWS Mitra. Anda mengevaluasi pengaruh kemampuan baru ini terhadap desain kontrol saat ini dan yang baru untuk lingkungan dan beban kerja Anda.

Anti-pola umum:

- Anda tidak berlangganan AWS blog dan RSS feed untuk mempelajari fitur dan layanan baru yang relevan dengan cepat
- Anda mengandalkan berita dan pemberitahuan tentang layanan dan fitur keamanan dari sumber tangan kedua
- Anda tidak mendorong AWS pengguna di organisasi Anda untuk tetap mendapat informasi tentang pembaruan terbaru

Manfaat menerapkan praktik terbaik ini: Ketika Anda selalu menggunakan layanan dan fitur keamanan baru, Anda dapat membuat keputusan-keputusan yang tepat berdasarkan informasi tentang penerapan kontrol di lingkungan cloud dan beban kerja Anda. Sumber-sumber ini membantu meningkatkan kesadaran akan lanskap keamanan yang berkembang dan bagaimana AWS layanan dapat digunakan untuk melindungi dari ancaman baru dan yang muncul.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

AWS menginformasikan pelanggan tentang layanan dan fitur keamanan baru melalui beberapa saluran:

- AWS Apa yang Baru
- AWS Blog Berita
- Blog Keamanan AWS
- Buletin Keamanan AWS
- Ikhtisar dokumentasi AWS

Anda dapat berlangganan topik Pembaruan Fitur AWS Harian menggunakan Amazon Simple Notification Service (AmazonSNS) untuk ringkasan pembaruan harian yang komprehensif. Beberapa layanan keamanan, seperti Amazon GuardDuty dan AWS Security Hub, menyediakan SNS topik mereka sendiri untuk tetap mendapat informasi tentang standar baru, temuan, dan pembaruan lainnya untuk layanan tertentu tersebut.

Layanan dan fitur baru juga diumumkan dan dijelaskan secara rinci selama konferensi, acara, dan webinar yang diselenggarakan di seluruh dunia setiap tahunnya. Catatan khusus adalah konferensi keamanan re:Inforce AWS tahunan dan konferensi re:Invent AWS yang lebih umum. Saluran AWS berita yang disebutkan sebelumnya membagikan pengumuman konferensi ini tentang keamanan dan layanan lainnya, dan Anda dapat melihat sesi pelarian pendidikan mendalam secara online di saluran Acara di.AWS YouTube

Anda juga dapat bertanya kepada tim Akun AWS Anda tentang pembaruan dan rekomendasi layanan keamanan terbaru. Anda dapat menghubungi tim Anda melalui formulir Dukunga Penjualan jika Anda tidak memiliki informasi kontak langsung mereka. Demikian pula, jika Anda berlangganan AWS Enterprise Support, Anda akan menerima pembaruan mingguan dari Manajer Akun Teknis (TAM) dan dapat menjadwalkan pertemuan tinjauan rutin dengan mereka.

Langkah-langkah implementasi

1. Berlangganan berbagai blog dan buletin dengan RSS pembaca favorit Anda atau ke topik Pembaruan Fitur Harian. SNS
2. Evaluasi AWS acara mana yang harus dihadiri untuk mempelajari langsung tentang fitur dan layanan baru.

3. Siapkan rapat dengan Akun AWS tim Anda untuk pertanyaan apa pun tentang memperbarui layanan dan fitur keamanan.
4. Pertimbangkan untuk berlangganan Enterprise Support untuk melakukan konsultasi rutin dengan Technical Account Manager ()TAM.

Sumber daya

Praktik-praktik terbaik terkait:

- PERF01-BP01 Pelajari dan pahami layanan dan fitur cloud yang tersedia
- COST01-BP07 Tetap up-to-date dengan rilis layanan baru

## Pengelolaan identitas dan akses

Pertanyaan

- SEC2. Bagaimana cara mengelola autentikasi untuk manusia dan mesin?
- SEC3. Bagaimana cara mengelola izin untuk manusia dan mesin?

### SEC2. Bagaimana cara mengelola autentikasi untuk manusia dan mesin?

Ada dua jenis identitas yang harus Anda kelola saat mendekati beban AWS kerja yang aman. Pemahaman tentang jenis identitas yang harus Anda kelola dan berikan akses akan membantu Anda memverifikasi identitas yang tepat memiliki akses ke sumber daya yang tepat dalam kondisi yang tepat.

Identitas Manusia: Administrator, pengembang, operator, dan pengguna akhir Anda memerlukan identitas untuk mengakses AWS lingkungan dan aplikasi Anda. Ini adalah anggota organisasi Anda, atau pengguna eksternal dengan siapa Anda berkolaborasi, dan yang berinteraksi dengan AWS sumber daya Anda melalui browser web, aplikasi klien, atau alat baris perintah interaktif.

Identitas Mesin: Aplikasi layanan, alat operasional, dan beban kerja Anda memerlukan identitas untuk membuat permintaan ke AWS layanan, misalnya, untuk membaca data. Identitas ini mencakup mesin yang berjalan di AWS lingkungan Anda seperti EC2 instans atau AWS Lambda fungsi Amazon. Anda juga dapat mengelola identitas mesin untuk pihak eksternal yang membutuhkan akses. Selain itu, Anda mungkin juga memiliki mesin di luar AWS yang membutuhkan akses ke AWS lingkungan Anda.

Praktik terbaik

- SEC02-BP01 Gunakan mekanisme masuk yang kuat
- SEC02-BP02 Gunakan kredenal sementara
- SEC02-BP03 Menyimpan dan menggunakan rahasia dengan aman
- SEC02-BP04 Mengandalkan penyedia identitas terpusat
- SEC02-BP05 Audit dan putar kredensional secara berkala
- SEC02-BP06 Mempekerjakan grup dan atribut pengguna

SEC02-BP01 Gunakan mekanisme masuk yang kuat

Masuk (otentikasi menggunakan kredensi masuk) dapat menimbulkan risiko saat tidak menggunakan mekanisme seperti otentikasi multi-faktor (MFA), terutama dalam situasi di mana kredensi masuk telah diungkapkan secara tidak sengaja atau mudah ditebak. Gunakan mekanisme masuk yang kuat untuk mengurangi risiko ini dengan mewajibkan MFA dan kebijakan kata sandi yang kuat.

Hasil yang diinginkan: Kurangi risiko akses yang tidak diinginkan ke kredensil AWS dengan menggunakan mekanisme masuk yang kuat untuk AWS Identity and Access Management (IAM) pengguna, pengguna Akun AWS root, AWS IAM Identity Center(penerus AWS Single Sign-On), dan penyedia identitas pihak ketiga. Ini berarti mewajibkanMFA, menegakkan kebijakan kata sandi yang kuat, dan mendeteksi perilaku login anomali.

Anti-pola umum:

- Tidak menerapkan kebijakan kata sandi yang kuat untuk identitas Anda termasuk kata sandi yang kompleks dan. MFA
- Kredensial yang sama digunakan oleh beberapa pengguna yang berbeda.
- Tidak menggunakan kontrol-kontrol detektif untuk mengenali aktivitas masuk yang mencurigakan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Ada banyak cara yang bisa digunakan identitas manusia masuk untuk ke AWS. Ini adalah praktik AWS terbaik untuk mengandalkan penyedia identitas terpusat menggunakan federasi (federasi langsung atau menggunakan AWS IAM Identity Center) saat mengautentikasi ke. AWS Dalam kasus tersebut, Anda harus membuat proses masuk yang aman dengan penyedia identitas Anda atau Microsoft Active Directory.

Ketika Anda pertama kali membuka Akun AWS, Anda mulai dengan pengguna Akun AWS root. Anda hanya harus menggunakan pengguna root akun untuk mengatur akses bagi para pengguna Anda (dan untuk tugas-tugas yang memerlukan pengguna root). Penting MFA untuk mengaktifkan pengguna root akun segera setelah membuka Anda Akun AWS dan untuk mengamankan pengguna root menggunakan panduan praktik AWS terbaik.

Jika Anda membuat pengguna masuk AWS IAM Identity Center, amankan proses masuk di layanan tersebut. Untuk identitas konsumen, Anda dapat menggunakan kumpulan pengguna Amazon Cognito dan mengamankan proses masuk yang ada di layanan tersebut, atau dengan menggunakan salah satu penyedia identitas yang didukung oleh kumpulan pengguna Amazon Cognito.

Jika Anda menggunakan AWS Identity and Access Management (IAM) pengguna, Anda akan mengamankan proses masuk menggunakanIAM.

Apa pun metode masuknya, Anda harus menerapkan kebijakan masuk yang kuat.

Langkah-langkah implementasi

Berikut ini adalah beberapa rekomendasi mekanisme masuk yang kuat secara umum. Pengaturan aktual yang Anda konfigurasikan harus ditetapkan oleh kebijakan perusahaan Anda atau menggunakan standar seperti NIST800-63.

- MembutuhkanMFA. Ini adalah praktik IAM terbaik yang diperlukan MFA untuk identitas dan beban kerja manusia. Menghidupkan MFA memberikan lapisan keamanan tambahan yang mengharuskan pengguna memberikan kredensi masuk dan kata sandi satu kali (OTP) atau string yang diverifikasi dan dihasilkan secara kriptografis dari perangkat keras.
- Berlakukan batas minimum karakter kata sandi, yang merupakan faktor utama dari kekuatan kata sandi.
- Berlakukan kompleksitas kata sandi agar kata sandi tidak mudah ditebak.
- Izinkan pengguna untuk mengubah kata sandi mereka sendiri.
- Buatlah identitas individu, bukan kredensial bersama. Dengan membuat identitas individu, Anda dapat memberikan kredensial keamanan yang unik kepada masing-masing pengguna. Penggunaan pengguna individu juga menyediakan kemampuan untuk mengaudit aktivitas dari setiap pengguna.

IAMRekomendasi Pusat Identitas:

- IAMIdentity Center menyediakan kebijakan kata sandi yang telah ditentukan saat menggunakan direktori default yang menetapkan panjang kata sandi, kompleksitas, dan persyaratan penggunaan kembali.
- Aktifkan MFA dan konfigurasikan pengaturan konteks-sadar atau selalu aktif MFA ketika sumber identitas adalah direktori default, atau AD Connector. AWS Managed Microsoft AD
- Memungkinkan pengguna untuk mendaftarkan MFA perangkat mereka sendiri.

Rekomendasi direktori kumpulan pengguna Amazon Cognito:

- Konfigurasikan pengaturan Kekuatan kata sandi.
- Membutuhkan MFA untuk pengguna.
- Gunakan pengaturan keamanan lanjutan kumpulan pengguna Amazon Cognito untuk fitur seperti autentikasi adaptif yang dapat memblokir setiap upaya masuk yang mencurigakan.

IAMrekomendasi pengguna:

- Idealnya Anda menggunakan Pusat IAM Identitas atau federasi langsung. Namun, Anda mungkin memiliki kebutuhan untuk IAM pengguna. Dalam hal ini, tetapkan kebijakan kata sandi untuk IAM pengguna. Anda dapat menggunakan kebijakan kata sandi tersebut untuk menentukan persyaratan, seperti jumlah karakter minimum, atau apakah kata sandi harus terdiri dari karakter non-alfabet atau tidak.
- Buat IAM kebijakan untuk menerapkan MFA proses masuk sehingga pengguna diizinkan mengelola kata sandi dan MFA perangkat mereka sendiri.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP03 Menyimpan dan menggunakan rahasia dengan aman
- SEC02-BP04 Mengandalkan penyedia identitas terpusat
- SEC03-BP08 Bagikan sumber daya dengan aman di dalam organisasi Anda

Dokumen terkait:

- AWS IAM Identity Center Kebijakan Kata Sandi

- IAMkebijakan kata sandi pengguna
- Mengatur kata sandi pengguna Akun AWS root
- Kebijakan kata sandi Amazon Cognito
- AWS kredensialnya
- IAMpraktik terbaik keamanan

Video terkait:

- Mengelola izin pengguna dalam skala besar dengan AWS IAM Identity Center
- Menguasai identitas di setiap lapisan beban kerja

SEC02-BP02 Gunakan kredenal sementara

Saat melakukan autentikasi jenis apa pun, sebaiknya Anda menggunakan kredensial sementara alih-alih kredensial jangka panjang untuk mengurangi atau menghindari risiko, misalnya seperti risiko pengungkapan, pembagian, dan pencurian kredensial.

Hasil yang diinginkan: Untuk mengurangi adanya risiko kredensial jangka panjang, Anda harus menggunakan kredensial sementara sedapat mungkin untuk identitas manusia dan mesin. Kredensi jangka panjang menciptakan banyak risiko, misalnya, mereka dapat diunggah dalam kode ke repositori publik. GitHub Dengan menggunakan kredensial sementara, Anda dapat secara signifikan mengurangi risiko penyusupan kredensial.

Anti-pola umum:

- Pengembang menggunakan kunci akses jangka panjang dari IAM pengguna daripada mendapatkan kredensi sementara dari federasi yang CLI menggunakan.
- Pengembang menyematkan kunci akses jangka panjang dalam kodenya dan mengunggah kode tersebut ke repositori Git publik.
- Pengembang menyematkan kunci akses jangka panjang di aplikasi seluler yang kemudian dibuat tersedia di toko aplikasi.
- Pengguna membagikan kunci akses jangka panjang kepada para pengguna lainnya, atau karyawan yang sudah keluar dari perusahaan tetapi masih memiliki kunci akses jangka panjang.
- Menggunakan kunci akses jangka panjang untuk identitas mesin meski pun dalam kasus ini kredensial sementara dapat digunakan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Gunakan kredensi keamanan sementara alih-alih kredensi jangka panjang untuk semua dan permintaan. AWS API CLI APIdan CLI permintaan ke AWS layanan harus, dalam hampir setiap kasus, ditandatangani menggunakan kunci AWS akses. Permintaan ini dapat Anda tandatangani dengan menggunakan kredensial jangka panjang maupun sementara. Satu-satunya waktu Anda harus menggunakan kredensi jangka panjang, juga dikenal sebagai kunci akses jangka panjang, adalah jika Anda menggunakan IAMpengguna atau pengguna Akun AWS root. Ketika Anda berfederasi ke AWS atau mengambil IAMperan melalui metode lain, kredenal sementara dihasilkan. Bahkan ketika Anda mengakses kredensi login AWS Management Console menggunakan, kredenal sementara dibuat agar Anda dapat melakukan panggilan ke layanan. AWS Anda hanya memerlukan kredensial jangka panjang untuk beberapa situasi saja dan Anda hampir dapat melakukan semua tugas dengan menggunakan kredensial sementara.

Menghindari penggunaan kredensi jangka panjang yang mendukung kredensi sementara harus berjalan seiring dengan strategi mengurangi penggunaan IAM pengguna yang mendukung federasi dan peran. IAM Meskipun IAM pengguna telah digunakan untuk identitas manusia dan mesin di masa lalu, kami sekarang menyarankan untuk tidak menggunakannya untuk menghindari risiko dalam menggunakan kunci akses jangka panjang.

Langkah-langkah implementasi

Untuk identitas manusia seperti karyawan, administrator, pengembang, operator, dan pelanggan:

- Anda harus mengandalkan penyedia identitas terpusat dan meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara. Federasi untuk para pengguna Anda dapat dilakukan baik dengan melakukan federasi langsung ke setiap Akun AWS atau menggunakan AWS IAM Identity Center dan penyedia identitas pilihan Anda. Federasi memberikan sejumlah keuntungan dibandingkan menggunakan IAM pengguna selain menghilangkan kredensi jangka panjang. Pengguna Anda juga dapat meminta kredensi sementara dari baris perintah untuk federasi langsung atau dengan menggunakan Pusat IAMIdentitas. Ini berarti bahwa ada beberapa kasus penggunaan yang memerlukan IAM pengguna atau kredensi jangka panjang untuk pengguna Anda.
- Saat memberikan pihak ketiga, seperti penyedia perangkat lunak sebagai layanan (SaaS), akses ke sumber daya di Akun AWS Anda, Anda dapat menggunakan peran lintas akun dan kebijakan berbasis sumber daya.

- Jika Anda perlu memberikan aplikasi kepada konsumen atau pelanggan akses ke AWS sumber daya Anda, Anda dapat menggunakan kumpulan identitas Amazon Cognito atau kumpulan pengguna Amazon Cognito untuk memberikan kredensi sementara. Izin untuk kredensional dikonfigurasi melalui peran. IAM Anda juga dapat menentukan IAM peran terpisah dengan izin terbatas untuk pengguna tamu yang tidak diautentikasi.

Untuk identitas mesin, Anda mungkin perlu menggunakan kredensial jangka panjang. Dalam kasus ini, Anda harus meminta beban kerja untuk menggunakan kredenal sementara dengan IAM peran untuk diakses. AWS

- Untuk Amazon Elastic Compute Cloud (AmazonEC2), Anda dapat menggunakan peran untuk Amazon EC2.

- AWS Lambdamemungkinkan Anda mengonfigurasi peran eksekusi Lambda untuk memberikan izin layanan untuk melakukan AWS tindakan menggunakan kredenal sementara. Ada banyak model serupa lainnya untuk AWS layanan untuk memberikan kredensi sementara menggunakan IAM peran.
- Untuk perangkat IoT, Anda dapat menggunakan penyedia kredensial AWS IoT Core untuk membuat permintaan kredensial sementara.
- Untuk sistem lokal atau sistem yang berjalan di luar AWS yang memerlukan akses ke AWS sumber daya, Anda dapat menggunakan IAMPeran Di Mana Saja.

Dalam beberapa skenario, kredensial sementara tidak dapat digunakan dan Anda mungkin perlu menggunakan kredensial jangka panjang. Dalam situasi ini, lakukan audit dan rotasi kredensial secara berkala dan rotasi kunci akses secara rutin untuk kasus penggunaan yang memerlukan kredensial jangka panjang. Beberapa contoh yang mungkin memerlukan kredensi jangka panjang termasuk WordPress plugin dan klien pihak ketiga. AWS Dalam situasi di mana Anda harus menggunakan kredensi jangka panjang, atau untuk kredensi selain kunci AWS akses, seperti login database, Anda dapat menggunakan layanan yang dirancang untuk menangani pengelolaan rahasia, seperti. AWS Secrets Manager Secrets Manager akan memudahkan Anda untuk mengelola, merotasi, dan menyimpan rahasia terenkripsi secara aman dengan menggunakan layanan-layanan yang didukung. Untuk informasi selengkapnya tentang cara merotasi kredensial jangka panjang, silakan lihat merotasi kunci akses.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP03 Menyimpan dan menggunakan rahasia dengan aman
- SEC02-BP04 Mengandalkan penyedia identitas terpusat
- SEC03-BP08 Bagikan sumber daya dengan aman di dalam organisasi Anda

Dokumen terkait:

- Kredensial Keamanan Sementara
- Kredensial AWS
- IAMPraktik Terbaik Keamanan
- IAMPeran
- IAMPusat Identitas
- Penyedia Identitas dan Federasi
- Merotasi Kunci Akses
- Solusi Partner Keamanan: Akses dan Kontrol Akses
- Pengguna Root Akun AWS

Video terkait:

- Mengelola izin pengguna dalam skala besar dengan AWS IAM Identity Center
- Menguasai identitas di setiap lapisan beban kerja

SEC02-BP03 Menyimpan dan menggunakan rahasia dengan aman

Beban kerja memerlukan sebuah kemampuan otomatis untuk membuktikan identitasnya ke basis data, sumber daya, dan layanan-layanan pihak ketiga. Ini dilakukan dengan menggunakan kredensil akses rahasia, seperti kunci API akses, kata sandi, dan token. OAuth Menggunakan sebuah layanan yang dibuat khusus untuk menyimpan, mengelola, dan merotasi kredensial ini akan membantu Anda untuk mengurangi kemungkinan peretasan kredensial.

Hasil yang diinginkan: Menerapkan sebuah mekanisme untuk mengelola kredensial aplikasi dengan aman yang akan mencapai tujuan-tujuan berikut:

- Melakukan identifikasi atas rahasia apa yang diperlukan untuk beban kerja.
- Mengurangi jumlah kredensial jangka panjang yang diperlukan dan menggunakan kredensial jangka pendek, jika memungkinkan, sebagai gantinya.
- Menetapkan penyimpanan yang aman dan rotasi otomatis kredensial jangka panjang yang tersisa.
- Melakukan audit terhadap akses ke rahasia yang ada di beban kerja.
- Melakukan pemantauan berkelanjutan untuk memverifikasi bahwa tidak ada rahasia yang disematkan di kode sumber selama proses pengembangan.
- Mengurangi kemungkinan terungkapnya kredensial secara tidak sengaja.

Anti-pola umum:

- Tidak melakukan rotasi kredensial.
- Menyimpan kredensial jangka panjang dalam kode sumber atau file konfigurasi.
- Menyimpan kredensial diam tanpa dienkripsi.

Manfaat menjalankan praktik terbaik ini:

- Rahasia yang disimpan diamankan dengan enkripsi saat diam mau pun bergerak.
- Akses ke kredensil terjaga keamanannya melalui API (anggap saja sebagai mesin penjual otomatis kredensi).
- Akses ke kredensial (baca dan tulis) diaudit dan dicatat log-nya.
- Pemisahan masalah: rotasi kredensial dilakukan oleh komponen terpisah, yang dapat dipisahkan dari bagian arsitektur lainnya.
- Rahasia didistribusikan secara otomatis ke komponen-komponen perangkat lunak sesuai permintaan dan rotasi dilakukan di sebuah lokasi pusat.
- Akses ke kredensial dapat dikontrol dengan sangat ketat dan terperinci.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Di masa lalu, kredensil yang digunakan untuk mengautentikasi ke database, pihak ketigaAPIs, token, dan rahasia lainnya mungkin telah disematkan dalam kode sumber atau dalam file lingkungan. AWS menyediakan beberapa mekanisme untuk menyimpan kredensil ini dengan aman, memutarnya secara otomatis, dan mengaudit penggunaannya.

Cara terbaik yang bisa Anda lakukan untuk mengelola rahasia adalah dengan mengikuti panduan penghapusan, penggantian, dan rotasi. Kredensial yang paling aman adalah kredensial yang tidak perlu Anda simpan, kelola, atau tangani. Jika ada kredensial yang sudah tidak Anda gunakan untuk menjalankan beban kerja, maka Anda dapat menghapusnya.

Apabila kredensial masih diperlukan untuk menjalankan beban kerja dengan benar, maka kredensial jangka panjangnya mungkin bisa diganti dengan kredensial sementara atau kredensial jangka pendek. Misalnya, alih-alih mengkodekan kunci akses AWS rahasia, pertimbangkan untuk mengganti kredensi jangka panjang itu dengan kredensi sementara menggunakan peran. IAM

Beberapa rahasia yang sudah lama ada mungkin tidak dapat dihapus atau diganti. Rahasia-rahasia ini dapat disimpan dalam sebuah layanan seperti AWS Secrets Manager, di mana mereka dapat disimpan secara terpusat, dikelola, dan dirotasi secara teratur.

Pengauditan kode sumber dan file konfigurasi beban kerja dapat menunjukkan berbagai jenis kredensial. Tabel berikut merangkum bermacam-macam strategi yang bisa digunakan untuk menangani berbagai jenis kredensial umum:

| Tipe kredensial | Deskripsi | Strategi yang disarankan |
| --- | --- | --- |
| Kunci akses IAM | AWS IAMakses dan kunci rahasia yang digunakan untuk mengambil IAM peran di dalam beban kerja | Ganti: Gunakan IAMperan yang ditetapkan ke instance komputasi (seperti Amazon EC2 atau AWS Lambda) sebagai gantinya. Untuk interoperabilitas dengan pihak ketiga yang memerluka n akses ke sumber daya di Anda Akun AWS, tanyakan apakah mereka mendukung akses AWS lintas akun. Untuk aplikasi seluler, sebaiknya Anda menggunakan kredensia l sementara melalui kolam identitas Amazon Cognito (identitas terfederasi). Untuk beban kerja yang berjalan di luar AWS, pertimbangkan |

| Tipe kredensial | Deskripsi | Strategi yang disarankan |
| --- | --- | --- |
| | | IAMRoles Anywhere atau AWS Systems Manager Hybrid Activations. |
| SSHkunci | Kunci pribadi Secure Shell yang digunakan untuk masuk ke EC2 instance Linux, secara manual atau sebagai bagian dari proses otomatis | Ganti: Gunakan AWS Systems Manager atau EC2Instance Connect untuk menyediak an akses terprogram dan manusiawi ke EC2 instance yang menggunakan IAM peran. |
| Kredensial aplikasi dan basis data | Kata sandi – string teks biasa | Rotasi: Simpan kredensia l di AWS Secrets Manager dan buat rotasi otomatis jika memungkinkan. |
| Kredensi Database Admin Amazon RDS dan Aurora | Kata sandi – string teks biasa | Ganti: Gunakan integrasi Secrets Manager dengan Amazon RDS atau Amazon Aurora. Selain itu, beberapa tipe RDS database dapat menggunakan IAM peran alih-alih kata sandi untuk beberapa kasus penggunaan (untuk detail lebih lanjut, lihat otentikasi IAM database). |
| OAuthtoken | Token rahasia – string teks biasa | Rotasi: Simpan token di AWS Secrets Manager dan konfigurasikan rotasi otomatis. |
| APItoken dan kunci | Token rahasia – string teks biasa | Rotasi: Simpan token di AWS Secrets Manager dan buat rotasi otomatis jika memungkinkan. |

Anti-pola yang umum adalah menyematkan kunci IAM akses di dalam kode sumber, file konfigurasi, atau aplikasi seluler. Ketika kunci IAM akses diperlukan untuk berkomunikasi dengan AWS layanan, gunakan kredensil keamanan sementara (jangka pendek). Kredensi jangka pendek ini dapat diberikan melalui IAMperan untuk EC2 instance, peran eksekusi untuk fungsi Lambda, peran IAMCognito untuk akses pengguna seluler, dan kebijakan IoT Core untuk perangkat IoT. Saat berinteraksi dengan pihak ketiga, lebih suka mendelegasikan akses ke IAM peran dengan akses yang diperlukan ke sumber daya akun Anda daripada mengonfigurasi IAM pengguna dan mengirimkan kunci akses rahasia kepada pihak ketiga untuk pengguna tersebut.

Ada banyak kasus di mana beban kerja membutuhkan penyimpanan rahasia yang diperlukan untuk berinteraksi dengan layanan dan sumber daya lainnya. AWS Secrets Managerdibuat dengan tujuan untuk mengelola kredensil ini dengan aman, serta penyimpanan, penggunaan, dan rotasi API token, kata sandi, dan kredensil lainnya.

AWS Secrets Manager menyediakan lima kemampuan utama untuk memastikan penyimpanan yang aman dan penanganan kredensi sensitif: enkripsi saat istirahat, enkripsidalam perjalanan, audit komprehensif, kontrol akses berbutir halus, dan rotasi kredenal yang dapat diperluas. Layanan manajemen rahasia lainnya dari Partner AWS atau Anda juga dapat menggunakan solusi-solusi yang dikembangkan secara lokal yang dapat memberikan kemampuan dan jaminan serupa.

Langkah-langkah implementasi

1. Identifikasi jalur kode yang berisi kredensi hard-code menggunakan alat otomatis seperti Amazon. CodeGuru
   a. Gunakan Amazon CodeGuru untuk memindai repositori kode Anda. Setelah peninjauan selesai, filter CodeGuru untuk menemukan baris kode yang bermasalah. `Type=Secrets`
2. Identifikasi kredensial yang dapat dihapus atau diganti.
   a. Identifikasi kredensial yang sudah tidak diperlukan, dan kemudian tandai untuk dihapus.
   b. Untuk Kunci AWS Rahasia yang disematkan dalam kode sumber, gantilah dengan IAM peran yang terkait dengan sumber daya yang diperlukan. Jika bagian dari beban kerja Anda berada di luar AWS tetapi memerlukan IAM kredensil untuk mengakses AWS sumber daya, pertimbangkan IAMPeran Anywhere atau AWS Systems Manager Hybrid Activations.
3. Untuk rahasia lama lainnya dari pihak ketiga yang memerlukan penggunaan strategi rotasi, integrasikan Secrets Manager ke dalam kode Anda untuk mengambil rahasia pihak ketiga pada waktu proses runtime.
   a. CodeGuru Konsol dapat secara otomatis membuat rahasia di Secrets Manager menggunakan kredenal yang ditemukan.

   b. Integrasikan pengambilan rahasia dari Secrets Manager ke dalam kode aplikasi Anda.
      i. Fungsi Lambda nirserver dapat menggunakan ekstensi Lambda yang bersifat agnostik bahasa.
      ii. Untuk EC2 instance atau container, AWS berikan contoh kode sisi klien untuk mengambil rahasia dari Secrets Manager dalam beberapa bahasa pemrograman populer.
4. Lakukan peninjauan terhadap basis kode Anda secara berkala dan lakukan pemindaian kembali untuk memverifikasi bahwa tidak ada rahasia baru yang ditambahkan ke kode.
   a. Pertimbangkan untuk menggunakan alat-alat seperti git-secret untuk mencegah memberikan rahasia baru ke repositori kode sumber Anda.
5. Pantau aktivitas Secrets Manager untuk mengetahui adanya indikasi penggunaan yang tidak terduga, akses rahasia yang tidak semestinya, atau upaya untuk menghapus rahasia.
6. Kurangi akses manusia ke kredensial. Batasi akses untuk membaca, menulis, dan memodifikasi kredensil ke IAM peran yang didedikasikan untuk tujuan ini, dan hanya menyediakan akses untuk mengambil peran ke sebagian kecil pengguna operasional.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP02 Gunakan kredenal sementara
- SEC02-BP05 Audit dan putar kredensional secara berkala

Dokumen terkait:

- Memulai dengan AWS Secrets Manager
- Penyedia Identitas dan Federasi
- Amazon CodeGuru Memperkenalkan Detektor Rahasia
- Bagaimana AWS Secrets Manager menggunakan AWS Key Management Service
- Enkripsi dan dekripsi rahasia di Secrets Manager
- Entri blog Secrets Manager
- Amazon RDS mengumumkan integrasi dengan AWS Secrets Manager

Video terkait:

- Praktik Terbaik untuk Mengelola, Mengambil, dan Merotasi Secret dalam Skala Besar
- Temukan Rahasia Hard-Coded Menggunakan Detektor Rahasia Amazon CodeGuru
- Mengamankan Rahasia untuk Beban Kerja Hybrid Menggunakan AWS Secrets Manager

Lokakarya terkait:

- Menyimpan, mengambil, dan mengelola kredensi sensitif di AWS Secrets Manager
- AWS Systems Manager Aktivasi Hibrida

SEC02-BP04 Mengandalkan penyedia identitas terpusat

Untuk identitas tenaga kerja (karyawan dan kontraktor), andalkan penyedia identitas yang memungkinkan Anda mengelola identitas di sebuah tempat yang tersentralisasi. Hal ini akan mempermudah Anda dalam melakukan pengelolaan akses di beberapa aplikasi dan sistem, karena Anda membuat, menetapkan, mengelola, mencabut, dan mengaudit akses dari satu lokasi.

Hasil yang diinginkan: Anda memiliki penyedia identitas terpusat tempat Anda mengelola pengguna tenaga kerja secara terpusat, kebijakan otentikasi (seperti memerlukan otentikasi multi-faktor (MFA)), dan otorisasi ke sistem dan aplikasi (seperti menetapkan akses berdasarkan keanggotaan atau atribut grup pengguna). Pengguna tenaga kerja Anda masuk ke penyedia identitas pusat dan melakukan penggabungan (federasi) (masuk tunggal) ke aplikasi internal dan eksternal, sehingga pengguna tidak perlu mengingat lebih dari satu kredensial. Penyedia identitas Anda terintegrasi dengan sistem sumber daya manusia (SDM) Anda sehingga perubahan personel secara otomatis akan disinkronkan ke penyedia identitas Anda. Misalnya, jika seseorang meninggalkan organisasi Anda, Anda dapat secara otomatis mencabut akses ke aplikasi dan sistem federasi (termasuk). AWS Anda telah mengaktifkan pencatatan log audit mendetail di penyedia identitas Anda dan memantau log tersebut untuk mendeteksi perilaku pengguna yang tidak biasa.

Anti-pola umum:

- Anda tidak menggunakan federasi dan masuk tunggal. Pengguna tenaga kerja Anda membuat akun dan kredensial pengguna terpisah di beberapa aplikasi dan sistem.
- Anda belum melakukan otomatisasi siklus hidup identitas untuk pengguna tenaga kerja, seperti dengan mengintegrasikan penyedia identitas Anda dengan sistem SDM Anda. Saat pengguna keluar dari organisasi atau beralih jabatan, Anda harus mengikuti proses manual untuk menghapus atau memperbarui catatan mereka di beberapa aplikasi dan sistem.

Manfaat menerapkan praktik terbaik ini: Dengan menggunakan penyedia identitas yang terpusat, Anda memiliki satu tempat untuk mengelola identitas dan kebijakan pengguna tenaga kerja, kemampuan untuk menetapkan akses aplikasi kepada pengguna dan grup, dan kemampuan untuk memantau aktivitas masuk pengguna. Dengan melakukan integrasi dengan sistem sumber daya manusia (SDM), ketika ada seorang pengguna beralih jabatan, perubahan ini akan disinkronkan dengan penyedia identitas yang secara otomatis memperbarui aplikasi dan izin yang ditetapkan. Ketika ada pengguna yang keluar dari organisasi Anda, maka identitas mereka pun secara otomatis dinonaktifkan di penyedia identitas, sehingga akses mereka ke aplikasi dan sistem gabungan dicabut.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Panduan untuk pengguna tenaga kerja yang mengakses AWS

Pengguna tenaga kerja seperti karyawan dan kontraktor di organisasi Anda mungkin memerlukan akses untuk AWS menggunakan AWS Management Console atau AWS Command Line Interface (AWS CLI) untuk menjalankan fungsi pekerjaan mereka. Anda dapat memberikan AWS akses ke pengguna tenaga kerja Anda dengan menggabungkan dari penyedia identitas terpusat Anda ke dua AWS tingkat: federasi langsung ke masing-masing Akun AWS atau federasi ke beberapa akun di organisasi Anda.AWS

- Untuk menyatukan pengguna tenaga kerja Anda secara langsung dengan masing-masing pengguna Akun AWS, Anda dapat menggunakan penyedia identitas terpusat untuk bergabung di akun itu. AWS Identity and Access Management Fleksibilitas IAM memungkinkan Anda mengaktifkan SAML2.0 terpisah atau Penyedia Identitas Open ID Connect (OIDC) untuk masing-masing Akun AWS dan menggunakan atribut pengguna gabungan untuk kontrol akses. Pengguna tenaga kerja Anda akan menggunakan browser web mereka untuk masuk ke penyedia identitas dengan memberikan kredensialnya (seperti kata sandi dan kode MFA token). Penyedia identitas mengeluarkan SAML pernyataan ke browser mereka yang dikirimkan ke AWS Management Console login URL untuk memungkinkan pengguna untuk masuk tunggal AWS Management Console dengan mengasumsikan Peran. IAM Pengguna Anda juga dapat memperoleh AWS API kredensi sementara untuk digunakan di AWS CLIatau AWS SDKsdari AWS STSdengan mengambil IAM peran menggunakan SAML pernyataan dari penyedia identitas.
- Untuk menggabungkan pengguna tenaga kerja Anda dengan beberapa akun di AWS organisasi Anda, Anda dapat menggunakannya AWS IAM Identity Centeruntuk mengelola akses secara terpusat bagi pengguna dan aplikasi tenaga kerja Anda. Akun AWS Anda mengaktifkan Pusat Identitas untuk organisasi Anda dan mengonfigurasi sumber identitas Anda. IAMIdentity Center

menyediakan direktori sumber identitas default yang dapat Anda gunakan untuk mengelola pengguna dan grup Anda. Atau, Anda dapat memilih sumber identitas eksternal dengan menghubungkan ke penyedia identitas eksternal Anda menggunakan SAML 2.0 dan secara otomatis menyediakan pengguna dan grup menggunakanSCIM, atau menghubungkan ke Direktori Microsoft AD Anda menggunakan. AWS Directory Service Setelah sumber identitas dikonfigurasi, Anda dapat menetapkan akses ke pengguna dan grup Akun AWS dengan menentukan kebijakan hak istimewa paling sedikit di set izin Anda. Pengguna tenaga kerja Anda dapat melakukan autentikasi melalui penyedia identitas pusat Anda untuk masuk ke portal akses AWS dan melakukan masuk tunggal ke aplikasi cloud Akun AWS dan yang ditetapkan untuknya. Pengguna Anda dapat mengonfigurasi AWS CLI v2 untuk mengautentikasi dengan Identity Center dan mendapatkan kredensional untuk menjalankan perintah. AWS CLI Identity Center juga memungkinkan akses masuk tunggal ke AWS aplikasi seperti portal Amazon SageMaker Studio dan AWS IoT Sitewise Monitor.

Setelah Anda mengikuti panduan sebelumnya, pengguna tenaga kerja Anda tidak perlu lagi menggunakan IAM pengguna dan grup untuk operasi normal saat mengelola beban kerja. AWS Sebagai gantinya, pengguna dan grup Anda dikelola di luar AWS dan pengguna dapat mengakses AWS sumber daya sebagai identitas gabungan. Identitas gabungan (terfederasi) menggunakan grup yang ditentukan oleh penyedia identitas terpusat Anda. Anda harus mengidentifikasi dan menghapus IAM grup, IAM pengguna, dan kredensi pengguna yang berumur panjang (kata sandi dan kunci akses) yang tidak lagi diperlukan di situs Anda. Akun AWSAnda dapat menemukan kredensi yang tidak digunakan menggunakan laporan IAM kredensi, menghapus IAMpengguna yang sesuai, dan menghapus grup. IAM Anda dapat menerapkan Kebijakan Kontrol Layanan (SCP) ke organisasi Anda yang membantu mencegah pembuatan IAM pengguna dan grup baru, menegakkan akses AWS tersebut melalui identitas gabungan.

Panduan untuk para pengguna aplikasi Anda

Anda dapat mengelola identitas pengguna aplikasi Anda, seperti aplikasi seluler, menggunakan Amazon Cognito sebagai penyedia identitas terpusat Anda. Amazon Cognito mengaktifkan autentikasi, otorisasi, dan pengelolaan pengguna, baik untuk aplikasi web mau pun aplikasi seluler Anda. Amazon Cognito menyediakan toko identitas yang menskalakan jutaan pengguna, mendukung federasi identitas sosial dan organisasi, serta menawarkan fitur-fitur keamanan canggih untuk membantu Anda melindungi para pengguna dan bisnis Anda. Anda dapat mengintegrasikan aplikasi web atau seluler kustom Anda dengan Amazon Cognito untuk menambahkan autentikasi pengguna dan kontrol akses ke aplikasi Anda dalam hitungan menit. Dibangun di atas standar identitas terbuka

seperti SAML dan Open ID Connect (OIDC), Amazon Cognito mendukung berbagai peraturan kepatuhan dan terintegrasi dengan sumber daya pengembangan frontend dan backend.

Langkah-langkah implementasi

Langkah-langkah untuk pengguna tenaga kerja yang mengakses AWS

- Gabungkan pengguna tenaga kerja Anda untuk AWS menggunakan penyedia identitas terpusat menggunakan salah satu pendekatan berikut:
  - Gunakan Pusat IAM Identitas untuk mengaktifkan sistem masuk tunggal ke beberapa Akun AWS di AWS organisasi Anda dengan berfederasi dengan penyedia identitas Anda.
  - Gunakan IAM untuk menghubungkan penyedia identitas Anda secara langsung ke masing-masing Akun AWS, memungkinkan akses berbutir halus gabungan.
- Identifikasi dan hapus IAM pengguna dan grup yang digantikan oleh identitas federasi.

Langkah-langkah untuk pengguna aplikasi Anda

- Gunakan Amazon Cognito sebagai penyedia identitas terpusat menuju aplikasi Anda.
- Integrasikan aplikasi kustom Anda dengan Amazon Cognito menggunakan OpenID Connect dan. OAuth Anda dapat mengembangkan aplikasi kustom menggunakan pustaka Amplify yang menyediakan antarmuka sederhana untuk diintegrasikan dengan berbagai AWS layanan, seperti Amazon Cognito untuk otentikasi.

Sumber daya

Praktik terbaik Well-Architected terkait:

- SEC02-BP06 Mempekerjakan grup dan atribut pengguna
- SEC03-BP02 Berikan akses hak istimewa paling sedikit
- SEC03-BP06 Mengelola akses berdasarkan siklus hidup

Dokumen terkait:

- Federasi identitas di AWS
- Praktik terbaik keamanan di IAM
- AWS Identity and Access Management Praktik terbaik

- Memulai administrasi delegasi Pusat IAM Identitas
- Cara menggunakan kebijakan yang dikelola pelanggan di Pusat IAM Identitas untuk kasus penggunaan lanjutan
- AWS CLI v2: Penyedia kredensi Pusat IAM Identitas

Video terkait:

- AWS Re: inforce 2022 - AWS Identity and Access Management () IAM menyelam dalam
- AWS re:invent 2022 - Sederhanakan akses tenaga kerja Anda yang ada dengan Identity Center IAM
- AWS Re: Invent 2018: Menguasai Identitas di Setiap Lapisan Kue

Contoh terkait:

- Workshop: Menggunakan AWS IAM Identity Center untuk mencapai manajemen identitas yang kuat
- Lokakarya: Identitas nirserver

Alat terkait:

- AWS Mitra Kompetensi Keamanan: Identity and Access Management
- AWS IAM Identity Center

SEC02-BP05 Audit dan putar kredensional secara berkala

Lakukan audit dan rotasi kredensial secara rutin membatasi seberapa lama kredensial dapat digunakan untuk mengakses sumber daya Anda. Kredensial jangka panjang dapat menimbulkan banyak risiko, tetapi risiko-risiko ini dapat dikurangi dengan secara rutin melakukan rotasi terhadap kredensial jangka panjang.

Hasil yang diinginkan: Mengimplementasikan rotasi kredensial untuk membantu Anda mengurangi risiko yang terkait dengan penggunaan kredensial jangka panjang. Melakukan audit dan perbaikan secara rutin untuk penggunaan yang tidak mematuhi kebijakan-kebijakan rotasi kredensial.

Anti-pola umum:

- Tidak melakukan audit penggunaan kredensial.

- Menggunakan kredensial jangka panjang saat tidak diperlukan.
- Menggunakan kredensial jangka panjang dan tidak melakukan rotasi kredensial secara rutin.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Bila Anda tidak dapat mengandalkan kredensi sementara dan memerlukan kredensil jangka panjang, kredensi audit untuk memverifikasi bahwa kontrol yang ditentukan seperti otentikasi multi-faktor (MFA) diberlakukan, diputar secara teratur, dan memiliki tingkat akses yang sesuai.

Validasi berkala, sebaiknya menggunakan sebuah alat otomatis, diperlukan untuk memverifikasi bahwa kontrol yang tepat sudah diberlakukan. Untuk identitas manusia, Anda harus mewajibkan pengguna untuk mengubah kata sandi mereka secara rutin dan memensiunkan kunci akses yang ditukar dengan kredensial sementara. Saat Anda berpindah dari AWS Identity and Access Management (IAM) pengguna ke identitas terpusat, Anda dapat membuat laporan kredensi untuk mengaudit pengguna Anda.

Kami juga menyarankan Anda menegakkan dan memantau MFA di penyedia identitas Anda. Anda dapat mengatur Aturan AWS Config, atau menggunakan Standar AWS Security Hub Keamanan, untuk memantau apakah pengguna telah mengonfigurasiMFA. Pertimbangkan untuk menggunakan IAM Peran Di Mana Saja untuk memberikan kredensi sementara untuk identitas mesin. Dalam situasi ketika menggunakan IAM peran dan kredensi sementara tidak dimungkinkan, sering mengaudit dan memutar kunci akses diperlukan.

Langkah-langkah implementasi

- Audit kredensi secara teratur: Mengaudit identitas yang dikonfigurasi di penyedia identitas Anda dan IAM membantu memverifikasi bahwa hanya identitas resmi yang memiliki akses ke beban kerja Anda. Identitas tersebut dapat mencakup, tetapi tidak terbatas pada, IAM pengguna, pengguna, AWS IAM Identity Center pengguna Active Directory, atau pengguna di penyedia identitas hulu yang berbeda. Misalnya, hapus orang-orang yang keluar dari organisasi, dan hapus pula peran lintas akun yang tidak lagi diperlukan. Memiliki proses untuk mengaudit izin secara berkala ke layanan yang diakses oleh IAM entitas. Tindakan ini akan membantu Anda mengidentifikasi kebijakan-kebijakan yang perlu diubah untuk menghapus izin-izin yang tidak digunakan. Gunakan laporan kredensi dan AWS Identity and Access Management Access Analyzeruntuk mengaudit IAM kredensi dan izin. Anda dapat menggunakan Amazon CloudWatch untuk mengatur alarm untuk API panggilan tertentu yang dipanggil dalam AWS lingkungan Anda.

Amazon juga GuardDuty dapat mengingatkan Anda tentang aktivitas tak terduga, yang mungkin menunjukkan akses yang terlalu permisif atau akses yang tidak diinginkan ke kredensil. IAM

- Putar kredensil secara teratur: Ketika Anda tidak dapat menggunakan kredensil sementara, putar kunci IAM akses jangka panjang secara teratur (maksimum setiap 90 hari). Tindakan ini akan membatasi waktu penggunaan kredensial untuk mengakses sumber daya Anda jika ada kunci akses yang bocor tanpa sepengetahuan Anda. Untuk informasi tentang memutar kunci akses bagi IAM pengguna, lihat Memutar kunci akses.
- Meninjau IAM izin: Untuk meningkatkan keamanan Anda Akun AWS, tinjau dan pantau setiap IAM kebijakan Anda secara teratur. Pastikan bahwa kebijakan tersebut memenuhi prinsip hak akses paling rendah.
- Pertimbangkan untuk mengotomatiskan pembuatan dan pembaruan IAM sumber daya: Pusat IAM Identitas mengotomatiskan banyak IAM tugas, seperti manajemen peran dan kebijakan. Atau, AWS CloudFormation dapat digunakan untuk mengotomatisasi penyebaran IAM sumber daya, termasuk peran dan kebijakan, untuk mengurangi kemungkinan kesalahan manusia karena template dapat diverifikasi dan dikontrol versi.
- Gunakan IAM Peran Di Mana Saja untuk mengganti IAM pengguna dengan identitas mesin: IAM Peran Di Mana Saja memungkinkan Anda menggunakan peran di area yang biasanya tidak dapat Anda lakukan, seperti server di lokasi. IAMRoles Anywhere menggunakan sertifikat X.509 tepercaya untuk mengautentikasi AWS dan menerima kredensi sementara. Menggunakan IAM Peran Di Mana Saja menghindari kebutuhan untuk memutar kredensil ini, karena kredensil jangka panjang tidak lagi disimpan di lingkungan lokal Anda. Perlu diketahui bahwa Anda harus memantau dan merotasi sertifikat X.509 sebelum memasuki masa kedaluwarsa.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP02 Gunakan kredenal sementara
- SEC02-BP03 Menyimpan dan menggunakan rahasia dengan aman

Dokumen terkait:

- Memulai dengan AWS Secrets Manager
- IAMPraktik Terbaik
- Penyedia Identitas dan Federasi

- Solusi Partner Keamanan: Akses dan Kontrol Akses
- Kredensial Keamanan Sementara
- Mendapatkan laporan kredensi untuk Anda Akun AWS

Video terkait:

- Praktik Terbaik untuk Mengelola, Mengambil, dan Merotasi Secret dalam Skala Besar
- Mengelola izin pengguna dalam skala besar dengan AWS IAM Identity Center
- Menguasai identitas di setiap lapisan beban kerja

Contoh terkait:

- Well-Architected Lab - Pembersihan Pengguna Otomatis IAM
- Well-Architected Lab - Penyebaran IAM Otomatis Grup dan Peran

SEC02-BP06 Mempekerjakan grup dan atribut pengguna

Penentuan izin sesuai dengan grup pengguna dan atribut akan membantu Anda mengurangi jumlah dan kompleksitas kebijakan, sehingga prinsip hak akses paling rendah dapat Anda wujudkan dengan lebih sederhana.  Anda dapat menggunakan grup pengguna untuk mengelola izin bagi banyak orang di satu tempat berdasarkan fungsi yang mereka lakukan di dalam organisasi Anda.  Atribut, seperti departemen atau lokasi, dapat menyediakan lapisan tambahan cakupan izin ketika ada orang yang melakukan fungsi serupa, tetapi untuk subset sumber daya yang berbeda.

Hasil yang diinginkan: Anda dapat menerapkan perubahan-perubahan izin berdasarkan fungsi untuk semua pengguna yang menjalankan fungsi tersebut.  Keanggotaan grup dan atribut mengatur izin pengguna, sehingga hal itu akan mengurangi kebutuhan untuk mengelola izin di tingkat masing-masing pengguna.  Grup dan atribut yang Anda tentukan di penyedia identitas (IdP) Anda disebarkan secara otomatis ke lingkungan AWS Anda.

Anti-pola umum:

- Mengelola izin untuk pengguna individual dan menduplikasinya kepada banyak pengguna.
- Menentukan grup pada tingkat yang terlalu tinggi, sehingga memberikan izin yang terlalu luas.
- Menentukan grup pada tingkat yang terlalu terperinci, sehingga menghasilkan duplikasi dan kebingungan terkait keanggotaan.

- Menggunakan grup dengan izin duplikat di seluruh subset sumber daya ketika atribut dapat digunakan sebagai gantinya.
- Tidak mengelola grup, atribut, dan keanggotaan melalui penyedia identitas standar yang terintegrasi dengan lingkungan AWS Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS izin didefinisikan dalam dokumen yang disebut kebijakan yang terkait dengan prinsipal, seperti pengguna, grup, peran, atau sumber daya. Untuk tenaga kerja Anda, hal ini akan memungkinkan Anda untuk menentukan grup berdasarkan fungsi yang dilakukan pengguna untuk organisasi Anda, bukan berdasarkan sumber daya yang diakses. Misalnya, `WebAppDeveloper` grup mungkin memiliki kebijakan yang dilampirkan untuk mengonfigurasi layanan seperti Amazon CloudFront dalam akun pengembangan. `AutomationDeveloper`Grup mungkin memiliki beberapa CloudFront izin yang sama dengan `WebAppDeveloper` grup. Izin ini dapat direkam dalam kebijakan terpisah dan dikaitkan dengan kedua grup, daripada memiliki pengguna dari kedua fungsi milik dari sebuah grup `CloudFrontAccess`.

Selain grup, Anda dapat menggunakan atribut untuk akses cakupan yang lebih jauh. Misalnya, Anda mungkin memiliki atribut `Project` untuk pengguna yang ada di grup `WebAppDeveloper` Anda untuk cakupan akses ke sumber daya khusus untuk proyek mereka. Dengan teknik ini, tidak diperlukan grup yang berbeda untuk pengembang aplikasi yang bekerja di proyek yang berbeda jika izin mereka sama. Cara Anda merujuk ke atribut dalam kebijakan izin didasarkan pada sumbernya, apakah atribut tersebut didefinisikan sebagai bagian dari protokol federasi Anda (sepertiSAML,, atauSCIM)OIDC, sebagai SAML pernyataan khusus, atau ditetapkan dalam Pusat IAM Identitas.

Langkah-langkah implementasi

1. Tetapkan di mana Anda akan menentukan grup dan atribut.
   a. Mengikuti panduan diSEC02-BP04 Mengandalkan penyedia identitas terpusat, Anda dapat menentukan apakah Anda perlu menentukan grup dan atribut dalam penyedia identitas Anda, dalam Pusat IAM Identitas, atau menggunakan grup IAM pengguna di akun tertentu.
2. Tentukan grup.
   a. Tentukan grup Anda berdasarkan fungsi dan cakupan akses yang diperlukan.
   b. Jika mendefinisikan dalam Pusat IAM Identitas, buat grup dan kaitkan tingkat akses yang diinginkan menggunakan set izin.

   c. Jika mendefinisikan dalam penyedia identitas eksternal, tentukan apakah penyedia mendukung SCIM protokol dan pertimbangkan untuk mengaktifkan penyediaan otomatis dalam Pusat Identitas. IAM  Kemampuan ini menyinkronkan pembuatan, keanggotaan, dan penghapusan grup antara penyedia Anda dan IAM Pusat Identitas.

3. Tentukan atribut.

   a. Jika menggunakan penyedia identitas eksternal, protokol SCIM dan SAML 2.0 menyediakan atribut tertentu secara default.  Atribut tambahan dapat didefinisikan dan diteruskan menggunakan SAML pernyataan menggunakan nama `https://aws.amazon.com/SAML/Attributes/PrincipalTag` atribut.

   b. Jika mendefinisikan dalam IAM Identity Center, aktifkan fitur access control (ABAC) berbasis atribut dan tentukan atribut yang diinginkan.

4. Tentukan cakupan izin berdasarkan grup dan atribut.

   a. Sebaiknya Anda menyertakan kondisi dalam kebijakan izin Anda yang membandingkan atribut pengguna utama Anda dengan atribut sumber daya yang diakses.  Misalnya, Anda dapat menentukan kondisi untuk mengizinkan akses ke sumber daya hanya jika nilai kunci kondisi `PrincipalTag` cocok dengan nilai kunci `ResourceTag` yang memiliki nama yang sama.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP04 Mengandalkan penyedia identitas terpusat
- SEC03-BP02 Berikan akses hak istimewa paling sedikit
- COST02-BP04 Melaksanakan kelompok dan peran

Dokumen terkait:

- IAMPraktik Terbaik
- Mengelola Identitas di Pusat IAM Identitas
- ABACUntuk apa AWS?
- ABACDi Pusat IAM Identitas

Video terkait:

- Mengelola izin pengguna dalam skala besar dengan AWS IAM Identity Center
- Menguasai identitas di setiap lapisan beban kerja

## SEC3. Bagaimana cara mengelola izin untuk manusia dan mesin?

Kelola izin untuk mengontrol akses ke orang dan identitas mesin yang memerlukan akses AWS dan beban kerja Anda. Izin akan mengontrol siapa yang dapat mengakses hal tertentu, beserta kondisinya.

Praktik terbaik

- SEC03-BP01 Tentukan persyaratan akses
- SEC03-BP02 Berikan akses hak istimewa paling sedikit
- SEC03-BP03 Menetapkan proses akses darurat
- SEC03-BP04 Kurangi izin terus menerus
- SEC03-BP05 Tentukan pagar pembatas izin untuk organisasi Anda
- SEC03-BP06 Mengelola akses berdasarkan siklus hidup
- SEC03-BP07 Menganalisis akses publik dan lintas akun
- SEC03-BP08 Bagikan sumber daya dengan aman di dalam organisasi Anda
- SEC03-BP09 Berbagi sumber daya secara aman dengan pihak ketiga

### SEC03-BP01 Tentukan persyaratan akses

Tiap-tiap komponen atau sumber daya beban kerja Anda perlu diakses oleh administrator, pengguna akhir, atau komponen-komponen lainnya. Penetapan harus jelas tentang siapa atau apa yang harus memiliki akses ke tiap-tiap komponen, pilih tipe identitas dan metode autentikasi serta otorisasi yang sesuai.

Anti-pola umum:

- Melakukan hard-coding atau menyimpan rahasia di dalam aplikasi Anda.
- Memberikan izin kustom untuk masing-masing pengguna.
- Menggunakan kredensial yang sudah berumur panjang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Tiap-tiap komponen atau sumber daya beban kerja Anda perlu diakses oleh administrator, pengguna akhir, atau komponen-komponen lainnya. Penetapan harus jelas tentang siapa atau apa yang harus memiliki akses ke tiap-tiap komponen, pilih tipe identitas dan metode autentikasi serta otorisasi yang sesuai.

Akses reguler ke Akun AWS dalam organisasi harus disediakan menggunakan akses federasi atau penyedia identitas terpusat. Anda juga harus memusatkan manajemen identitas Anda dan memastikan bahwa ada praktik yang mapan untuk mengintegrasikan AWS akses ke siklus hidup akses karyawan Anda. Misalnya, saat ada seorang karyawan yang berganti peran pekerjaan dengan level akses berbeda, maka keanggotaan grupnya juga harus berubah agar sesuai dengan persyaratan akses baru yang berlaku padanya.

Saat Anda menetapkan persyaratan akses untuk identitas selain manusia, Anda harus menentukan aplikasi dan komponen mana yang memerlukan akses dan bagaimana izin diberikan. Menggunakan IAM peran yang dibangun dengan model akses hak istimewa paling sedikit adalah pendekatan yang disarankan. AWS Kebijakan terkelola menyediakan kebijakan yang telah ditentukan sebelumnya IAM yang mencakup sebagian besar kasus penggunaan umum.

AWS layanan, seperti AWS Secrets Managerdan AWS Systems Manager Parameter Store, dapat membantu memisahkan rahasia dari aplikasi atau beban kerja dengan aman jika tidak layak untuk menggunakan peran. IAM Di Secrets Manager, Anda dapat membuat rotasi otomatis untuk kredensial Anda. Anda dapat menggunakan Systems Manager untuk mereferensikan parameter dalam skrip, perintah, SSM dokumen, konfigurasi, dan alur kerja otomatisasi dengan menggunakan nama unik yang Anda tentukan saat membuat parameter.

Anda dapat menggunakan AWS Identity and Access Management Peran Di Mana Saja untuk mendapatkan kredensi keamanan sementara IAM untuk beban kerja yang berjalan di luar. AWS Beban kerja Anda dapat menggunakan IAMkebijakan dan IAMperan yang sama yang Anda gunakan dengan AWS aplikasi untuk mengakses AWS sumber daya.

Jika memungkinkan, Anda sebaiknya menggunakan kredensial sementara jangka pendek, bukan kredensial statis jangka panjang. Untuk skenario di mana Anda memerlukan pengguna dengan akses yang terprogram dan kredensial jangka panjang, gunakan informasi kunci akses yang terakhir digunakan untuk merotasi dan menghapus kunci akses.

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

| Pengguna mana yang membutuhkan akses programatis? | Untuk | Oleh |
| --- | --- | --- |
| Identitas tenaga kerja<br><br>(Pengguna dikelola di Pusat IAM Identitas) | Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs | Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.<br><br>• Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna.<br>• Untuk AWS SDKs, alat, dan AWS APIs, lihat otentikasi Pusat IAM Identitas di Panduan Referensi Alat AWS SDKs dan Alat. |
| IAM | Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs | Mengikuti petunjuk dalam Menggunakan kredensi sementara dengan AWS sumber daya di IAMPanduan Pengguna. |
| IAM | (Tidak direkomendasikan) Gunakan kredensi jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs | Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.<br><br>• Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensi IAM pengguna di Panduan Pengguna.AWS Command Line Interface |

| Pengguna mana yang membutuhkan akses programatis? | Untuk | Oleh |
| --- | --- | --- |
| | | • Untuk AWS SDKs dan alat, lihat Mengautentikasi menggunakan kredensi jangka panjang di Panduan Referensi Alat AWS SDKs dan Alat.<br>• Untuk AWS APIs, lihat Mengelola kunci akses untuk IAM pengguna di Panduan IAM Pengguna. |

Sumber daya

Dokumen terkait:

- Kontrol akses berbasis atribut () ABAC
- AWS IAM Identity Center
- IAMPeran Di Mana Saja
- AWS Kebijakan terkelola untuk Pusat IAM Identitas
- AWS IAMkondisi kebijakan
- IAMkasus penggunaan
- Hapus kredensial yang tidak perlu
- Bekerja dengan Kebijakan
- Cara mengontrol akses ke AWS sumber daya berdasarkan Akun AWS, OU, atau organisasi
- Identifikasi, atur, dan kelola rahasia dengan mudah menggunakan pencarian yang disempurnakan di AWS Secrets Manager

Video terkait:

- Menjadi Master IAM Kebijakan dalam 60 Menit atau Kurang
- Pemisahan Tugas, Hak Akses Paling Rendah, Delegasi, dan CI/CD

- Merampingkan manajemen identitas dan akses untuk inovasi

SEC03-BP02 Berikan akses hak istimewa paling sedikit

Salah satu praktik terbaik yang bisa digunakan adalah memberikan hanya akses yang diperlukan identitas untuk melakukan tindakan tertentu pada sumber daya tertentu dalam kondisi tertentu. Gunakan atribut grup dan identitas untuk menetapkan izin secara dinamis dalam skala besar, bukannya menentukan izin satu per satu untuk setiap pengguna. Misalnya, Anda dapat memberikan kepada sebuah grup pengembang akses untuk mengelola sumber daya untuk proyek mereka saja. Dengan cara ini, jika seorang pengembang keluar dari proyek, maka akses pengembang tersebut secara otomatis dicabut tanpa mengubah kebijakan akses dasar.

Hasil yang diinginkan: Para pengguna seharusnya hanya memiliki izin yang diperlukan untuk melakukan pekerjaannya. Para pengguna seharusnya hanya diberi akses ke lingkungan produksi untuk melakukan tugas tertentu dalam jangka waktu terbatas dan akses harus dicabut setelah tugas tersebut selesai. Izin harus dicabut jika sudah tidak digunakan lagi, termasuk saat pengguna beralih ke proyek atau jabatan kerja lain. Hak akses administrator hanya boleh diberikan kepada sekelompok kecil administrator yang sudah tepercaya. Izin harus ditinjau secara rutin untuk menghindari creep izin. Akun sistem atau mesin seharusnya hanya boleh diberi rangkaian izin paling sedikit yang diperlukan untuk menyelesaikan tugas-tugas mereka.

Anti-pola umum:

- Memberikan izin administrator kepada para pengguna secara default.
- Menggunakan pengguna root untuk day-to-day aktivitas.
- Membuat kebijakan yang terlalu permisif, tetapi tanpa memberikan hak istimewa administrator penuh.
- Tidak meninjau izin untuk memahami apakah izin tersebut memberikan hak akses paling rendah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Prinsip hak akses paling rendah menyatakan bahwa identitas hanya boleh mendapatkan izin untuk melakukan serangkaian tindakan terkecil yang diperlukan untuk memenuhi tugas tertentu. Hal ini akan menyeimbangkan kegunaan, efisiensi, dan keamanan. Pengoperasian berdasarkan prinsip ini akan membantu Anda membatasi akses yang tidak diinginkan dan membantu Anda dalam memantau siapa saja yang memiliki akses ke sumber daya yang mana. IAMpengguna dan peran

tidak memiliki izin secara default. Pengguna root memiliki akses penuh secara default dan harus dikontrol, dipantau, dan digunakan secara ketat hanya untuk tugas-tugas yang memerlukan akses root.

IAMkebijakan digunakan untuk secara eksplisit memberikan izin untuk IAM peran atau sumber daya tertentu. Misalnya, kebijakan berbasis identitas dapat dilampirkan ke IAM grup, sementara bucket S3 dapat dikontrol oleh kebijakan berbasis sumber daya.

Saat membuat IAM kebijakan, Anda dapat menentukan tindakan layanan, sumber daya, dan kondisi yang harus benar AWS untuk mengizinkan atau menolak akses. AWS mendukung berbagai kondisi untuk membantu Anda menjangkau akses ke bawah. Misalnya, dengan menggunakan kunci `PrincipalOrgID` kondisi, Anda dapat menolak tindakan jika pemohon bukan bagian dari Organisasi Anda AWS .

Anda juga dapat mengontrol permintaan yang dibuat AWS layanan atas nama Anda, seperti AWS CloudFormation membuat AWS Lambda fungsi, menggunakan tombol `CalledVia` kondisi. Anda harus melapisi jenis kebijakan yang berbeda untuk menetapkan defense-in-depth dan membatasi izin keseluruhan pengguna Anda. Anda juga bisa membatasi izin yang dapat diberikan beserta kondisinya. Misalnya, Anda dapat mengizinkan tim aplikasi membuat IAM kebijakan mereka sendiri untuk sistem yang mereka bangun, tetapi juga harus menerapkan Batas Izin untuk membatasi izin maksimum yang dapat diterima sistem.

Langkah-langkah implementasi

- Menerapkan kebijakan hak istimewa terkecil: Tetapkan kebijakan akses dengan hak istimewa paling sedikit ke IAM grup dan peran untuk mencerminkan peran atau fungsi pengguna yang telah Anda tetapkan.
  - Kebijakan dasar API penggunaan: Salah satu cara untuk menentukan izin yang diperlukan adalah dengan meninjau AWS CloudTrail log. Ulasan ini memungkinkan Anda membuat izin yang disesuaikan dengan tindakan yang sebenarnya dilakukan pengguna. AWSIAMAccess Analyzer dapat secara otomatis menghasilkan IAM kebijakan berdasarkan aktivitas. Anda dapat menggunakan IAM Access Advisor di tingkat organisasi atau akun untuk melacak informasi terakhir yang diakses untuk kebijakan tertentu.
- Pertimbangkan untuk menggunakan kebijakan terkelola AWS untuk fungsi pekerjaan. Saat mulai membuat kebijakan izin berbutir halus, mungkin sulit untuk mengetahui dari mana harus memulai. AWS telah mengelola kebijakan untuk peran pekerjaan umum, misalnya penagihan, administrator database, dan ilmuwan data. Kebijakan ini dapat membantu Anda mempersempit akses yang dimiliki pengguna sekaligus menentukan cara menerapkan kebijakan hak akses paling rendah.

- Hapus izin yang tidak perlu: Hapus izin-izin yang tidak diperlukan dan pangkas kembali kebijakan yang terlalu permisif. IAMPembuatan kebijakan Access Analyzer dapat membantu menyempurnakan kebijakan izin.
- Pastikan bahwa para pengguna memiliki akses terbatas ke lingkungan produksi: Pengguna hanya boleh memiliki akses ke lingkungan produksi yang memiliki kasus penggunaan yang valid. Setelah pengguna menyelesaikan tugas-tugas tertentu yang memerlukan akses produksi, akses harus dicabut. Pembatasan akses ke lingkungan produksi akan membantu Anda mencegah kejadian tak terduga yang memengaruhi produksi dan memperkecil cakupan dampak akses yang tidak diharapkan.
- Pertimbangkan batasan izin: Batas izin adalah fitur untuk menggunakan kebijakan terkelola yang menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas. IAM Batas izin entitas mengizinkannya untuk melakukan hanya tindakan yang diizinkan oleh kebijakan berbasis identitas dan batas izinnya.
- Pertimbangkan tag sumber daya untuk izin: Model kontrol akses berbasis atribut yang menggunakan tag sumber daya akan memungkinkan Anda untuk memberikan akses berdasarkan tujuan sumber daya, pemilik, lingkungan, atau kriteria lainnya. Misalnya, Anda dapat menggunakan tanda sumber daya untuk membedakan lingkungan pengembangan dan produksi. Dengan tanda ini, Anda dapat membatasi pengembang agar hanya dapat mengakses lingkungan pengembangan. Dengan memadukan kebijakan pemberian tanda dan izin, Anda dapat memiliki akses sumber daya yang mendetail tanpa harus menentukan kebijakan kustom dan rumit untuk setiap fungsi tugas.
- Gunakan kebijakan kontrol layanan untuk AWS Organizations. Kebijakan kontrol layanan secara terpusat mengontrol izin maksimum yang tersedia bagi akun anggota yang ada di organisasi Anda. Fungsi penting dari kebijakan kontrol layanan adalah untuk memungkinkan Anda membatasi izin pengguna root di dalam akun anggota. Juga pertimbangkan untuk menggunakan AWS Control Tower, yang menyediakan kontrol terkelola preskriptif yang memperkaya. AWS Organizations Anda juga dapat menentukan kontrol Anda sendiri di dalam Control Tower.
- Menetapkan kebijakan siklus hidup pengguna untuk organisasi Anda: Kebijakan siklus hidup pengguna menentukan tugas yang akan dilakukan saat pengguna di-onboard AWS, mengubah peran atau cakupan pekerjaan, atau tidak lagi memerlukan akses ke. AWS Peninjauan izin harus dilakukan dalam setiap langkah dalam siklus hidup pengguna untuk memverifikasi bahwa izin dibatasi dengan sesuai dan untuk menghindari creep izin.
- Tetapkan jadwal reguler untuk meninjau izin dan menghapus izin yang tidak diperlukan: Anda harus secara teratur meninjau akses pengguna untuk memverifikasi bahwa pengguna tidak

memiliki akses yang terlalu permisif. AWS Configdan IAM Access Analyzer dapat membantu saat mengaudit izin pengguna.

- Tetapkan matriks peran pekerjaan: Matriks peran pekerjaan memvisualisasikan berbagai peran dan tingkat akses yang diperlukan dalam AWS jejak Anda. Dengan menggunakan sebuah matriks peran kerja, Anda dapat menentukan dan memisahkan izin berdasarkan tanggung jawab pengguna di dalam organisasi. Gunakan grup alih-alih menerapkan izin langsung ke masing-masing pengguna atau peran.

Sumber daya

Dokumen terkait:

- Berikan hak akses paling rendah
- Batas izin untuk entitas IAM
- Teknik untuk menulis kebijakan hak istimewa IAM paling sedikit
- IAMAccess Analyzer mempermudah penerapan izin hak istimewa paling sedikit dengan membuat IAM kebijakan berdasarkan aktivitas akses
- Delegasikan manajemen izin kepada pengembang dengan menggunakan batas IAM izin
- Menyempurnakan Izin dengan menggunakan informasi yang terakhir kali diakses
- IAMjenis kebijakan dan kapan menggunakannya
- Menguji IAM kebijakan dengan simulator IAM kebijakan
- Pagar pembatas di AWS Control Tower
- Arsitektur Zero Trust: Sebuah perspektif AWS
- Bagaimana menerapkan prinsip hak istimewa paling sedikit dengan CloudFormation StackSets
- Kontrol akses berbasis atribut () ABAC
- Mengurangi cakupan kebijakan dengan melihat aktivitas pengguna
- Lihat akses peran
- Gunakan Pemberian Tag untuk Mengatur Lingkungan Anda dan Mendorong Akuntabilitas
- Strategi Pemberian Tag AWS
- Pemberian tag pada sumber daya AWS

Video terkait:

- Manajemen izin generasi berikutnya

- Zero Trust: Sebuah AWS perspektif

Contoh terkait:

- Lab: batas IAM izin yang mendelegasikan pembuatan peran
- Lab: kontrol akses berbasis IAM tag untuk EC2

SEC03-BP03 Menetapkan proses akses darurat

Buat proses yang memungkinkan akses darurat ke beban kerja Anda jika terjadi masalah pada penyedia identitas tersentralisasi Anda.

Anda harus merancang desain proses untuk berbagai mode kegagalan yang dapat mengakibatkan terjadinya sebuah peristiwa darurat. Misalnya, dalam keadaan normal, pengguna tenaga kerja Anda bergabung ke cloud menggunakan penyedia identitas terpusat (SEC02-BP04) untuk mengelola beban kerja mereka. Namun, jika penyedia identitas tersentralisasi Anda gagal, atau konfigurasi untuk federasi di cloud diubah, maka pengguna tenaga kerja Anda mungkin tidak dapat melakukan federasi ke cloud. Proses akses darurat akan memungkinkan administrator yang berwenang untuk mengakses sumber daya cloud Anda melalui cara alternatif (seperti bentuk federasi alternatif atau akses pengguna langsung) untuk memperbaiki masalah dengan konfigurasi federasi atau beban kerja Anda. Proses akses darurat tersebut digunakan sampai mekanisme federasi normal berhasil dipulihkan.

Hasil yang diinginkan:

- Anda telah menentukan dan mendokumentasikan mode kegagalan yang terhitung sebagai sebuah keadaan darurat: pertimbangkan keadaan normal Anda dan sistem yang diandalkan oleh para pengguna untuk mengelola beban kerja mereka. Pertimbangkan bagaimana masing-masing dependensi ini dapat gagal dan menyebabkan terjadinya sebuah keadaan darurat. Anda mungkin akan menemukan pertanyaan dan praktik-praktik terbaik dalam Pilar Keandalan yang berguna untuk mengidentifikasi mode kegagalan dan merancang arsitektur sistem yang lebih tangguh untuk meminimalkan kemungkinan terjadinya kegagalan.
- Anda telah mendokumentasikan langkah-langkah yang harus diikuti untuk mengonfirmasi bahwa kegagalan dianggap sebagai keadaan darurat. Misalnya, Anda dapat meminta administrator identitas Anda untuk memeriksa status penyedia identitas utama dan siaga Anda dan, jika keduanya tidak tersedia, maka Anda harus mengumumkan peristiwa darurat untuk kegagalan penyedia identitas.

- Anda telah menentukan sebuah proses akses darurat khusus untuk masing-masing jenis mode darurat atau kegagalan. Pengkhususan ini dapat mengurangi godaan di pihak pengguna Anda untuk terlalu sering menggunakan sebuah proses umum untuk semua jenis keadaan darurat. Proses akses darurat Anda menggambarkan keadaan di mana masing-masing proses harus digunakan dan, sebaliknya, situasi di mana proses tidak boleh digunakan dan menunjuk ke proses alternatif yang mungkin berlaku.
- Proses Anda didokumentasikan dengan baik dengan instruksi yang mendetail dan playbook yang dapat diikuti dengan cepat dan dengan efisien. Ingatlah bahwa sebuah peristiwa darurat dapat menjadi saat-saat yang memusingkan bagi para pengguna Anda dan mereka sedang berada di bawah tekanan waktu yang ekstrem, jadi buatlah desain proses yang sesederhana mungkin.

Anti-pola umum:

- Anda tidak memiliki proses akses darurat yang didokumentasikan dengan baik dan teruji dengan baik. Pengguna Anda tidak siap untuk menghadapi sebuah keadaan darurat dan mengikuti proses yang diimprovisasi ketika ada sebuah peristiwa darurat yang muncul.
- Proses akses darurat Anda bergantung pada sistem yang sama (seperti penyedia identitas tersentralisasi) dengan mekanisme akses normal Anda. Ini artinya, kegagalan sistem tersebut dapat memengaruhi mekanisme akses normal dan darurat Anda dan akan mengganggu kemampuan Anda untuk pulih dari kegagalan tersebut.
- Proses akses darurat Anda digunakan dalam situasi yang tidak darurat. Misalnya, pengguna Anda sering kali menyalahgunakan proses akses darurat karena mereka merasa lebih mudah melakukan perubahan secara langsung daripada mengirimkan perubahan melalui sebuah pipeline.
- Proses akses darurat Anda tidak menghasilkan log yang memadai untuk mengaudit proses tersebut, atau log tersebut tidak dipantau untuk mendapatkan peringatan mengenai adanya potensi penyalahgunaan proses.

Manfaat menjalankan praktik terbaik ini:

- Dengan memiliki proses akses darurat yang didokumentasikan dengan baik dan teruji dengan baik, Anda dapat mengurangi waktu yang dibutuhkan pengguna untuk merespons dan menyelesaikan sebuah peristiwa darurat. Hal ini dapat menghasilkan lebih sedikit waktu henti dan ketersediaan yang lebih tinggi untuk layanan-layanan yang Anda berikan kepada pelanggan Anda.
- Anda dapat melacak setiap permintaan akses darurat dan mendeteksi serta memberikan peringatan mengenai adanya upaya penyalahgunaan proses untuk peristiwa yang tidak darurat.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Bagian ini memberikan panduan untuk membuat proses akses darurat untuk beberapa mode kegagalan yang terkait dengan beban kerja yang digunakan AWS, dimulai dengan panduan umum yang berlaku untuk semua mode kegagalan dan diikuti oleh panduan khusus berdasarkan jenis mode kegagalan.

Panduan umum untuk semua mode kegagalan

Pertimbangkan hal berikut saat Anda membuat desain proses akses darurat untuk sebuah mode kegagalan:

- Dokumentasikan kondisi awal (pre-conditions) dan asumsi mengenai proses tersebut: kapan proses tersebut harus digunakan dan kapan proses tersebut tidak boleh digunakan. Penting untuk memiliki detail mode kegagalan dan mendokumentasikan asumsi, seperti status keadaan sistem-sistem terkait lainnya. Misalnya, proses untuk Mode Kegagalan 2 mengasumsikan bahwa penyedia identitas tersedia, tetapi konfigurasi AWS dimodifikasi atau telah kedaluwarsa.
- Pra-buat sumber daya yang dibutuhkan oleh proses akses darurat (SEC10-BP05). Misalnya, pra-buat akses darurat Akun AWS dengan IAM pengguna dan peran, dan IAM peran lintas akun di semua akun beban kerja. Hal ini akan memastikan bahwa semua sumber daya ini siap dan tersedia ketika ada sebuah peristiwa darurat yang terjadi. Dengan membuat sumber daya sebelumnya, Anda tidak memiliki ketergantungan pada bidang AWS kontrol APIs (digunakan untuk membuat dan memodifikasi AWS sumber daya) yang mungkin tidak tersedia dalam keadaan darurat. Selanjutnya, dengan membuat IAM sumber daya sebelumnya, Anda tidak perlu memperhitungkan potensi penundaan karena konsistensi akhirnya.
- Sertakan proses akses darurat sebagai bagian dari rencana manajemen insiden Anda (SEC10-BP02). Dokumentasikan bagaimana peristiwa darurat dilacak dan dikomunikasikan kepada orang lain yang ada dalam organisasi Anda, seperti tim sejawat, pimpinan Anda, dan, jika ada, secara eksternal kepada para pelanggan dan partner bisnis Anda.
- Tentukan proses permintaan akses darurat yang ada dalam sistem alur kerja permintaan layanan yang ada, jika Anda memilikinya. Biasanya, sistem alur kerja semacam ini akan memungkinkan Anda untuk membuat formulir penerimaan informasi untuk mengumpulkan informasi tentang permintaan, melacak permintaan melalui setiap tahap alur kerja, dan menambahkan langkah-langkah persetujuan otomatis dan manual. Hubungkan setiap permintaan dengan sebuah peristiwa darurat terkait yang dilacak dalam sistem manajemen insiden Anda. Dengan memiliki sistem yang

seragam untuk akses darurat, Anda dapat melacak permintaan tersebut dalam satu sistem tunggal, menganalisis tren penggunaan, dan meningkatkan kualitas proses Anda.

- Pastikan bahwa proses akses darurat Anda hanya dapat dimulai oleh pengguna yang berwenang dan itu memerlukan persetujuan dari rekan sejawat atau manajemen pengguna yang sesuai. Proses persetujuan harus beroperasi secara efektif baik di dalam mau pun di luar jam kerja. Tentukan bagaimana permintaan persetujuan mengizinkan pemberi persetujuan sekunder jika pemberi persetujuan utama tidak tersedia dan bagaimana permintaan itu ditingkatkan ke rantai manajemen Anda hingga mendapatkan persetujuan.
- Pastikan bahwa proses tersebut menghasilkan log dan peristiwa audit yang mendetail, baik untuk upaya yang berhasil maupun upaya yang gagal mendapatkan akses darurat. Pantau proses permintaan serta mekanisme akses darurat untuk mendeteksi adanya penyalahgunaan atau akses yang tidak sah. Korelasikan aktivitas dengan peristiwa darurat yang sedang berlangsung dari sistem manajemen insiden Anda dan munculkan peringatan ketika ada tindakan yang terjadi di luar periode waktu yang diharapkan. Misalnya, Anda harus memantau dan memberikan peringatan mengenai aktivitas dalam akses darurat Akun AWS, karena akun tersebut tidak boleh digunakan dalam operasi normal.
- Lakukan pengujian terhadap proses akses darurat secara berkala untuk memverifikasi bahwa langkah-langkahnya sudah jelas dan memberikan tingkat akses yang benar dengan cepat dan efisien. Proses akses darurat Anda harus diuji sebagai bagian dari simulasi respons insiden (SEC10-BP07) dan tes pemulihan bencana (-BP03). REL13

Mode Kegagalan 1: Penyedia identitas yang digunakan untuk federasi AWS tidak tersedia

Seperti yang dijelaskan dalam SEC02-BP04 Mengandalkan penyedia identitas terpusat, kami sarankan untuk mengandalkan penyedia identitas terpusat untuk menyatukan pengguna tenaga kerja Anda untuk memberikan akses ke. Akun AWS Anda dapat mengfederasi ke beberapa Akun AWS di AWS organisasi Anda menggunakan Pusat IAM Identitas, atau Anda dapat federasi ke individu Akun AWS menggunakan. IAM Dalam kedua kasus tersebut, pengguna tenaga kerja melakukan autentikasi dengan penyedia identitas tersentralisasi Anda sebelum diarahkan ke titik akhir masuk AWS ke masuk tunggal.

Apabila penyedia identitas tersentralisasi Anda tidak tersedia, pengguna tenaga kerja Anda tidak dapat melakukan federasi ke Akun AWS atau mengelola beban kerja mereka. Dalam acara darurat ini, Anda dapat menyediakan proses akses darurat untuk sekelompok kecil administrator untuk mengakses Akun AWS untuk melakukan tugas-tugas penting yang tidak dapat menunggu sampai penyedia identitas terpusat Anda kembali online. Misalnya, penyedia identitas Anda tidak tersedia

selama 4 jam dan selama periode tersebut Anda perlu mengubah batas atas grup EC2 Auto Scaling Amazon di akun Produksi untuk menangani lonjakan lalu lintas pelanggan yang tidak terduga. Administrator darurat Anda harus mengikuti proses akses darurat untuk mendapatkan akses ke produksi tertentu Akun AWS dan membuat perubahan yang diperlukan.

Proses akses darurat bergantung pada akses darurat yang dibuat sebelumnya Akun AWS yang digunakan semata-mata untuk akses darurat dan memiliki AWS sumber daya (seperti IAM peran dan IAM pengguna) untuk mendukung proses akses darurat. Selama operasi normal, tidak ada yang boleh mengakses akun akses darurat tersebut dan Anda harus melakukan pemantauan atas dan memberikan peringatan mengenai terjadinya penyalahgunaan akun ini (untuk lebih jelasnya, lihat bagian panduan umum sebelumnya).

Akun akses darurat memiliki IAM peran akses darurat dengan izin untuk mengambil peran lintas akun di akun Akun AWS yang memerlukan akses darurat. IAMPeran ini telah dibuat sebelumnya dan dikonfigurasi dengan kebijakan kepercayaan yang mempercayai IAM peran akun darurat.

Proses akses darurat dapat menggunakan salah satu pendekatan berikut ini:

- Anda dapat membuat serangkaian IAMpengguna untuk administrator darurat Anda di akun akses darurat dengan kata sandi dan MFA token kuat yang terkait. IAMPengguna ini memiliki izin untuk mengambil IAM peran yang kemudian memungkinkan akses lintas akun ke Akun AWS tempat akses darurat diperlukan. Kami sarankan Anda untuk membuat pengguna sesedikit mungkin dan menetapkan masing-masing pengguna ke satu administrator darurat. Selama keadaan darurat, pengguna administrator darurat masuk ke akun akses darurat menggunakan kata sandi dan kode MFA token mereka, beralih ke IAM peran akses darurat di akun darurat, dan akhirnya beralih ke IAM peran akses darurat di akun beban kerja untuk melakukan tindakan perubahan darurat. Keuntungan dari pendekatan ini adalah bahwa setiap IAM pengguna ditugaskan ke satu administrator darurat dan Anda dapat mengetahui pengguna mana yang masuk dengan meninjau peristiwa. CloudTrail Kerugiannya adalah Anda harus mempertahankan banyak IAM pengguna dengan kata sandi dan MFA token berumur panjang yang terkait.
- Anda dapat menggunakan pengguna Akun AWS root akses darurat untuk masuk ke akun akses darurat, mengambil IAM peran untuk akses darurat, dan mengambil peran lintas akun di akun beban kerja. Kami merekomendasikan pengaturan kata sandi yang kuat dan beberapa MFA token untuk pengguna root. Kami juga merekomendasikan untuk menyimpan kata sandi dan MFA token di brankas kredensi perusahaan yang aman yang memberlakukan otentikasi dan otorisasi yang kuat. Anda harus mengamankan faktor pengaturan ulang kata sandi dan MFA token: setel alamat email akun ke daftar distribusi email yang dipantau oleh administrator keamanan cloud Anda, dan nomor telepon akun ke nomor telepon bersama yang juga dipantau oleh administrator keamanan.

Keunggulan pendekatan ini adalah bahwa hanya ada satu set kredensial pengguna root yang harus dikelola. Kelemahan pendekatan ini adalah karena ini merupakan pengguna bersama, maka ada beberapa administrator yang memiliki kemampuan untuk masuk sebagai pengguna root. Anda harus melakukan audit terhadap peristiwa log brankas korporasi Anda untuk mengidentifikasi administrator mana yang menggunakan kata sandi pengguna root.

Mode Kegagalan 2: Konfigurasi penyedia identitas AWS dimodifikasi atau telah kedaluwarsa

Untuk memungkinkan pengguna tenaga kerja Anda bergabung Akun AWS, Anda dapat mengonfigurasi Pusat IAM Identitas dengan penyedia identitas eksternal atau membuat Penyedia Identitas (IAMSEC02-BP04). Biasanya, Anda mengonfigurasinya dengan mengimpor XML dokumen SAML meta-data yang disediakan oleh penyedia identitas Anda. XMLDokumen meta-data mencakup sertifikat X.509 yang sesuai dengan kunci pribadi yang digunakan penyedia identitas untuk menandatangani pernyataannya. SAML

Konfigurasi di AWS sisi -ini dapat diubah atau dihapus secara tidak sengaja oleh administrator. Dalam skenario lain, sertifikat X.509 yang diimpor ke AWS mungkin kedaluwarsa dan meta-data baru XML dengan sertifikat baru belum diimpor ke. AWS Kedua skenario dapat merusak federasi AWS untuk pengguna tenaga kerja Anda, yang mengakibatkan keadaan darurat.

Dalam keadaan darurat seperti itu, Anda dapat memberikan akses kepada administrator identitas Anda AWS untuk memperbaiki masalah federasi. Misalnya, administrator identitas Anda menggunakan proses akses darurat untuk masuk ke akses darurat Akun AWS, beralih ke peran di akun administrator Pusat Identitas, dan memperbarui konfigurasi penyedia identitas eksternal dengan mengimpor XML dokumen SAML meta-data terbaru dari penyedia identitas Anda untuk mengaktifkan kembali federasi. Setelah federasi selesai diperbaiki, pengguna tenaga kerja Anda kemudian melanjutkan penggunaan proses operasi normal untuk melakukan federasi ke akun beban kerja mereka.

Anda dapat mengikuti pendekatan-pendekatan yang diuraikan dalam Mode Kegagalan 1 sebelumnya untuk membuat sebuah proses akses darurat. Anda dapat memberikan hak akses paling rendah kepada administrator identitas Anda sehingga hanya bisa mengakses akun administrator Pusat Identitas dan melakukan tindakan pada Pusat Identitas di akun tersebut.

Mode Kegagalan 3: Gangguan Pusat Identitas

Jika terjadi Wilayah AWS gangguan atau Pusat IAM Identitas, kami sarankan Anda menyiapkan konfigurasi yang dapat Anda gunakan untuk menyediakan akses sementara ke AWS Management Console.

Proses akses darurat menggunakan federasi langsung dari penyedia identitas Anda ke IAM dalam akun darurat. Untuk mendapatkan detail tentang proses dan pertimbangan desain, silakan lihat Mengatur akses darurat ke AWS Management Console.

Langkah-langkah implementasi

Langkah-langkah umum untuk semua mode kegagalan

- Buat yang Akun AWS didedikasikan untuk proses akses darurat. Pra-buat IAM sumber daya yang dibutuhkan dalam akun seperti IAM peran atau IAM pengguna, dan Penyedia IAM Identitas opsional. Selain itu, pra-buat IAM peran lintas akun dalam beban kerja dengan hubungan kepercayaan Akun AWS dengan IAM peran yang sesuai di akun akses darurat. Anda dapat menggunakan AWS CloudFormation StackSets dengan AWS Organizations untuk membuat sumber daya tersebut di akun anggota di organisasi Anda.

- Buat kebijakan kontrol AWS Organizations layanan (SCPs) untuk menolak penghapusan dan modifikasi IAM peran lintas akun di anggota. Akun AWS
- CloudTrail Aktifkan akses darurat Akun AWS dan kirim peristiwa jejak ke bucket S3 pusat di koleksi Akun AWS log Anda. Jika Anda menggunakannya AWS Control Tower untuk mengatur dan mengatur lingkungan AWS multi-akun, maka setiap akun yang Anda buat menggunakan AWS Control Tower atau mendaftar AWS Control Tower telah CloudTrail diaktifkan secara default dan dikirim ke bucket S3 dalam arsip log khusus. Akun AWS
- Pantau aktivitas akun akses darurat dengan membuat EventBridge aturan yang cocok saat login konsol dan API aktivitas berdasarkan IAM peran darurat. Kirimkan notifikasi ke pusat operasi keamanan Anda ketika ada aktivitas yang terjadi di luar peristiwa darurat yang sedang berlangsung yang terlacak dalam sistem manajemen insiden Anda.

Langkah-langkah tambahan untuk Mode Kegagalan 1: Penyedia identitas yang digunakan untuk federasi tidak AWS tersedia dan Mode Kegagalan 2: Konfigurasi penyedia identitas AWS dimodifikasi atau telah kedaluwarsa

- Buatlah sumber daya di awal tergantung pada mekanisme yang Anda pilih untuk akses darurat:
  - Menggunakan IAM pengguna: pra-buat IAM pengguna dengan kata sandi yang kuat dan MFA perangkat terkait.
  - Menggunakan pengguna root akun darurat: konfigurasikan pengguna root dengan kata sandi yang kuat dan simpan kata sandi tersebut di dalam brankas kredensial korporasi Anda. Kaitkan

beberapa MFA perangkat fisik dengan pengguna root dan simpan perangkat di lokasi yang dapat diakses dengan cepat oleh anggota tim administrator darurat Anda.

Langkah-langkah tambahan untuk Mode Kegagalan 3: Gangguan pusat identitas

- Sebagaimana dijelaskan dalam Mengatur akses darurat ke AWS Management Console, dalam akses darurat Akun AWS, buat Penyedia IAM Identitas untuk mengaktifkan SAML federasi langsung dari penyedia identitas Anda.
- Buat grup operasi darurat di IdP Anda tanpa anggota.
- Buat IAM peran yang sesuai dengan grup operasi darurat di akun akses darurat.

Sumber daya

Praktik terbaik Well-Architected terkait:

- SEC02-BP04 Mengandalkan penyedia identitas terpusat
- SEC03-BP02 Berikan akses hak istimewa paling sedikit
- SEC10-BP02 Mengembangkan rencana manajemen insiden
- SEC10-BP07 Jalankan hari permainan

Dokumen terkait:

- Mengatur akses darurat ke AWS Management Console
- Mengaktifkan SAML 2.0 pengguna federasi untuk mengakses AWS Management Console
- Akses pecah kaca

Video terkait:

- AWS re:invent 2022 - Sederhanakan akses tenaga kerja Anda yang ada dengan Identity Center IAM
- AWS Re: inforce 2022 - AWS Identity and Access Management () IAM menyelam dalam

Contoh terkait:

- Peran Pecah Kaca AWS

- Kerangka kerja playbook pelanggan AWS
- Contoh playbook respons insiden AWS

SEC03-BP04 Kurangi izin terus menerus

Jika tim Anda telah menentukan akses yang diperlukan, maka Anda harus menghapus izin-izin yang tidak diperlukan dan menetapkan proses peninjauan untuk mendapatkan izin hak akses paling rendah. Lakukan pemantauan secara terus-menerus dan hapus identitas serta izin-izin yang tidak diperlukan, baik untuk akses manusia maupun mesin.

Hasil yang diinginkan: Kebijakan izin harus mematuhi prinsip hak akses paling rendah. Setelah penetapan tugas dan peran pekerjaan sudah menjadi lebih baik, kebijakan izin Anda perlu ditinjau untuk menghapus izin-izin yang tidak perlu. Pendekatan ini mempersempit cakupan dampak akibat terjadinya kebocoran kredensial secara tidak sengaja, atau karena diakses tanpa otorisasi.

Anti-pola umum:

- Memberikan izin administrator kepada para pengguna secara default.
- Membuat kebijakan yang terlalu permisif, tetapi tanpa memberikan hak istimewa administrator penuh.
- Menyimpan kebijakan izin meski sudah tidak diperlukan lagi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Setelah tim dan proyek mulai, kebijakan izin permisif mungkin digunakan untuk menumbuhkan banyak inovasi dan ketangkasan. Misalnya, dalam lingkungan pengembangan atau pengujian, pengembang dapat diberikan akses ke serangkaian AWS layanan yang luas. Sebaiknya evaluasi akses secara terus-menerus dan batasi akses hanya untuk layanan dan tindakan layanan yang diperlukan untuk menyelesaikan tugas saat ini. Sebaiknya evaluasi ini dilakukan untuk identitas manusia maupun mesin. Identitas mesin, kadang-kadang disebut akun sistem atau layanan, adalah identitas yang memberikan AWS akses ke aplikasi atau server. Akses ini penting, terutama dalam sebuah lingkungan produksi, yang apabila izinnya terlalu permisif, maka dampaknya bisa begitu luas dan berpotensi mengekspos data konsumen.

AWS menyediakan beberapa metode untuk membantu mengidentifikasi pengguna, peran, izin, dan kredensil yang tidak digunakan. AWS juga dapat membantu menganalisis aktivitas akses

IAM pengguna dan peran, termasuk kunci akses terkait, dan akses ke AWS sumber daya seperti objek di bucket Amazon S3. AWS Identity and Access Management Access Analyzer pembuatan kebijakan dapat membantu Anda dalam membuat kebijakan izin terbatas berdasarkan layanan aktual dan tindakan yang berinteraksi dengan prinsipal. Kontrol akses berbasis atribut (ABAC) dapat membantu menyederhanakan pengelolaan izin, karena Anda dapat memberikan izin kepada pengguna menggunakan atribut mereka alih-alih melampirkan kebijakan izin secara langsung ke setiap pengguna.

Langkah-langkah implementasi

- Penggunaan AWS Identity and Access Management Access Analyzerhttps://docs.aws.amazon.com/IAM/latest/UserGuide/what-is-access-analyzer.html: IAM Access Analyzer membantu mengidentifikasi sumber daya di organisasi dan akun Anda, seperti bucket atau IAM peran Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) atau peran yang dibagikan dengan entitas eksternal.
- Gunakan pembuatan kebijakan IAM Access Analyzer: Pembuatan kebijakan IAM Access Analyzer membantu Anda membuat kebijakan izin berbutir halus berdasarkan aktivitas akses IAM pengguna atau peran.
- Menentukan kerangka waktu dan kebijakan penggunaan yang dapat diterima untuk IAM pengguna dan peran: Gunakan stempel waktu yang terakhir diakses untuk mengidentifikasi pengguna dan peran yang tidak digunakan dan menghapusnya. Tinjau layanan dan tindakan informasi yang terakhir diakses untuk mengidentifikasi dan masukkan izin dalam cakupan untuk pengguna dan peran tertentu. Misalnya, Anda dapat menggunakan informasi yang terakhir diakses untuk mengidentifikasi tindakan Amazon S3 tertentu yang diperlukan oleh peran aplikasi dan membatasi akses hanya untuk tindakan-tindakan tersebut. Fitur informasi yang diakses terakhir tersedia di AWS Management Console dan secara terprogram memungkinkan Anda untuk memasukkannya ke dalam alur kerja infrastruktur dan alat otomatis Anda.
- Pertimbangkan untuk mencatat peristiwa data di AWS CloudTrail: Secara default, CloudTrail tidak mencatat peristiwa data seperti aktivitas tingkat objek Amazon S3 (misalnya, `GetObject` dan) atau aktivitas tabel `DeleteObject` Amazon DynamoDB (misalnya, dan). `PutItem` `DeleteItem` Pertimbangkan untuk mengaktifkan pencatatan log untuk peristiwa ini sehingga Anda bisa menentukan pengguna dan peran apa yang perlu mengakses objek Amazon S3 dan item tabel DynamoDB tertentu.

Sumber daya

Dokumen terkait:

- Berikan hak akses paling rendah
- Hapus kredensial yang tidak perlu
- Apa itu AWS CloudTrail?
- Bekerja dengan Kebijakan
- Pencatatan log dan pemantauan DynamoDB
- Menggunakan pencatatan CloudTrail peristiwa untuk bucket dan objek Amazon S3
- Mendapatkan laporan kredensi untuk Anda Akun AWS

Video terkait:

- Menjadi Master IAM Kebijakan dalam 60 Menit atau Kurang
- Pemisahan Tugas, Hak Akses Paling Rendah, Delegasi, dan CI/CD
- AWS Re: inforce 2022 - AWS Identity and Access Management () IAM menyelam dalam

SEC03-BP05 Tentukan pagar pembatas izin untuk organisasi Anda

Gunakan pagar pembatas izin untuk mengurangi cakupan izin-izin yang tersedia yang dapat diberikan kepada pengguna utama. Rantai evaluasi kebijakan izin mencakup pagar pembatas yang digunakan untuk menentukan izin efektif pengguna utama saat membuat keputusan otorisasi. Anda dapat menentukan pagar pembatas dengan menggunakan pendekatan berbasis lapisan. Terapkan beberapa pagar pembatas secara meluas di seluruh organisasi Anda dan terapkan pagar pembatas lainnya secara terperinci ke sesi akses sementara.

Hasil yang diinginkan: Anda memiliki isolasi lingkungan yang jelas menggunakan Akun AWS terpisah. Kebijakan kontrol layanan (SCPs) digunakan untuk menentukan pagar pembatas izin di seluruh organisasi. Pagar pembatas yang lebih meluas ditetapkan pada tingkat hierarki yang paling dekat dengan root organisasi Anda, dan pagar pembatas yang lebih ketat ditetapkan lebih dekat ke tingkat akun individual. Jika didukung, kebijakan sumber daya akan menentukan kondisi yang harus dipenuhi oleh pengguna utama untuk mendapatkan akses ke sebuah sumber daya. Kebijakan sumber daya juga mengurangi cakupan dari serangkaian tindakan yang diizinkan, jika sesuai. Batasan izin diterapkan pada pengguna utama yang mengelola izin beban kerja, dengan mendelegasikan manajemen izin kepada pemilik beban kerja individual.

Anti-pola umum:

- Membuat anggota Akun AWS dalam AWS Organisasi, tetapi tidak menggunakan SCPs untuk membatasi penggunaan dan izin yang tersedia untuk kredensialnya.
- Menetapkan izin berdasarkan hak akses paling rendah, tetapi tidak menerapkan pagar pembatas pada kumpulan izin maksimum yang dapat diberikan.
- Mengandalkan dasar penolakan implisit AWS IAM untuk membatasi izin, percaya bahwa kebijakan tidak akan memberikan izin izin eksplisit yang tidak diinginkan.
- Menjalankan beberapa lingkungan beban kerja secara bersamaan Akun AWS, lalu mengandalkan mekanisme sepertiVPCs, tag, atau kebijakan sumber daya untuk menegakkan batasan izin.

Manfaat menerapkan praktik terbaik ini: Pagar pembatas izin akan membantu Anda untuk membangun keyakinan bahwa izin yang tidak diinginkan tidak dapat diberikan, bahkan ketika kebijakan izin mencoba melakukannya. Hal ini dapat menyederhanakan penentuan dan pengelolaan izin dengan mengurangi cakupan maksimum izin yang perlu dipertimbangkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sebaiknya Anda gunakan pendekatan berbasis lapisan untuk menentukan pagar pembatas izin bagi organisasi Anda. Pendekatan ini secara sistematis akan mengurangi izin maksimum yang mungkin digunakan saat lapisan tambahan diterapkan. Hal ini akan membantu Anda memberikan akses berdasarkan prinsip hak akses paling rendah, sehingga itu akan mengurangi risiko akses yang tidak diinginkan karena terjadinya kesalahan konfigurasi kebijakan.

Langkah pertama untuk membuat pagar pembatas izin adalah mengisolasi beban kerja dan lingkungan Anda ke dalam Akun AWS terpisah. Prinsipal dari satu akun tidak dapat mengakses sumber daya di akun lain tanpa izin eksplisit untuk melakukannya, bahkan ketika kedua akun berada di organisasi yang sama atau di bawah unit AWS organisasi yang sama (OU). Anda dapat menggunakan OUs untuk mengelompokkan akun yang ingin Anda kelola sebagai satu unit.

Langkah selanjutnya adalah mengurangi izin maksimum yang dapat Anda berikan kepada pengguna utama yang ada dalam akun anggota di organisasi Anda. Anda dapat menggunakan kebijakan kontrol layanan (SCPs) untuk tujuan ini, yang dapat Anda terapkan pada OU atau akun. SCPsdapat menerapkan kontrol akses umum, seperti membatasi akses ke spesifik Wilayah AWS, membantu mencegah sumber daya dihapus, atau menonaktifkan tindakan layanan yang berpotensi berisiko. SCPsyang Anda terapkan ke root organisasi Anda hanya memengaruhi akun anggotanya, bukan akun manajemen. SCPshanya mengatur kepala sekolah dalam organisasi Anda. Anda SCPs tidak mengatur kepala sekolah di luar organisasi Anda yang mengakses sumber daya Anda.

Langkah selanjutnya adalah menggunakan kebijakan IAM sumber daya untuk mencakup tindakan yang tersedia yang dapat Anda ambil pada sumber daya yang mereka atur, bersama dengan kondisi apa pun yang harus dipenuhi oleh kepala pelaksana. Ini bisa seluas memungkinkan semua tindakan selama prinsipal adalah bagian dari organisasi Anda (menggunakan kunci PrincipalOrgId kondisi), atau sedetail hanya mengizinkan tindakan tertentu dengan IAM peran tertentu. Anda dapat mengambil pendekatan serupa dengan kondisi dalam kebijakan kepercayaan IAM peran. Jika kebijakan kepercayaan sumber daya atau peran secara eksplisit menyebutkan prinsipal di akun yang sama dengan peran atau sumber daya yang diatur, prinsipal tersebut tidak memerlukan IAM kebijakan terlampir yang memberikan izin yang sama. Jika prinsipal berada di akun yang berbeda dari sumber daya, maka kepala sekolah memang memerlukan IAM kebijakan terlampir yang memberikan izin tersebut.

Sering kali, sebuah tim beban kerja ingin mengelola izin yang dibutuhkan beban oleh kerja mereka. Ini mungkin mengharuskan mereka untuk membuat IAM peran baru dan kebijakan izin. Anda dapat menangkap cakupan maksimum izin yang diizinkan tim untuk diberikan dalam batas IAM izin, dan mengaitkan dokumen ini ke IAM peran yang kemudian dapat digunakan tim untuk mengelola IAM peran dan izin mereka. Pendekatan ini dapat memberi mereka kemampuan untuk menyelesaikan pekerjaan mereka sambil mengurangi risiko memiliki IAM akses administratif.

Langkah yang lebih terperinci adalah menerapkan teknik manajemen akses istimewa (PAM) dan manajemen akses tinggi sementara (TEAM). Salah satu contohnya PAM adalah meminta kepala sekolah untuk melakukan otentikasi multi-faktor sebelum mengambil tindakan istimewa. Untuk informasi selengkapnya, lihat Mengonfigurasi akses MFA yang dilindungi API. TEAMmembutuhkan solusi yang mengelola persetujuan dan jangka waktu bahwa kepala sekolah diizinkan untuk memiliki akses yang lebih tinggi. Salah satu pendekatannya adalah untuk sementara menambahkan prinsipal ke kebijakan kepercayaan peran untuk IAM peran yang memiliki akses tinggi. Pendekatan lain adalah, dalam operasi normal, mengurangi izin yang diberikan kepada kepala sekolah dengan IAM peran menggunakan kebijakan sesi, dan kemudian mencabut sementara pembatasan ini selama jendela waktu yang disetujui. Untuk mempelajari lebih lanjut tentang solusi yang divalidasi oleh AWS dan mitra terpilih, lihat Akses yang ditingkatkan sementara.

Langkah-langkah implementasi

1. Isolasikan beban kerja dan lingkungan Anda ke dalam Akun AWS terpisah.
2. Gunakan SCPs untuk mengurangi set izin maksimum yang dapat diberikan kepada kepala sekolah dalam akun anggota organisasi Anda.
   a. Kami menyarankan Anda mengambil pendekatan daftar izin untuk menulis Anda SCPs yang menyangkal semua tindakan kecuali yang Anda izinkan, dan kondisi di mana mereka diizinkan.

Mulailah dengan menentukan Sumber Daya yang ingin Anda kontrol, dan kemudian atur Efek ke Tolak. Gunakan NotAction elemen untuk menolak semua tindakan kecuali yang Anda tentukan. Gabungkan ini dengan NotLike Kondisi untuk menentukan kapan tindakan ini diizinkan, jika berlaku, seperti StringNotLike dan ArnNotLike.

b. Lihat Contoh kebijakan kontrol layanan.

3. Gunakan kebijakan IAM sumber daya untuk mengurangi cakupan dan menentukan kondisi untuk tindakan yang diizinkan pada sumber daya. Gunakan kondisi dalam kebijakan kepercayaan IAM peran untuk membuat batasan pada asumsi peran.
4. Tetapkan batasan IAM izin untuk IAM peran yang kemudian dapat digunakan tim beban kerja untuk mengelola IAM peran dan izin beban kerja mereka sendiri.
5. Evaluasi PAM dan TEAM solusi berdasarkan kebutuhan Anda.

Sumber daya

Dokumen terkait:

- Perimeter data pada AWS
- Tetapkan pagar pembatas izin dengan menggunakan perimeter data
- Logika evaluasi kebijakan

Contoh terkait:

- Contoh kebijakan kontrol layanan

Alat terkait:

- Solusi AWS : Manajemen Akses Tinggi Sementara
- Solusi mitra keamanan yang divalidasi untuk TEAM

SEC03-BP06 Mengelola akses berdasarkan siklus hidup

Lakukan pemantauan dan penyesuaian terhadap izin-izin yang diberikan kepada para pengguna utama Anda (pengguna, peran, dan grup) di sepanjang siklus hidupnya dalam organisasi Anda. Sesuaikan keanggotaan grup jika pengguna berganti peran, dan hapus akses saat seorang pengguna meninggalkan organisasi.

Hasil yang diinginkan: Anda melakukan pemantauan dan menyesuaikan izin sepanjang siklus hidup pengguna utama yang ada dalam organisasi, mengurangi risiko hak akses istimewa yang tidak perlu. Anda memberikan akses yang sesuai saat membuat pengguna. Anda mengubah akses saat tanggung jawab pengguna berubah, dan Anda menghapus akses ketika pengguna tersebut tidak lagi aktif atau telah meninggalkan organisasi. Anda mengelola perubahan yang terjadi pada pengguna, peran, dan grup secara terpusat. Anda menggunakan otomatisasi untuk menyebarkan perubahan ke AWS lingkungan Anda.

Anti-pola umum:

- Anda memberikan hak akses yang berlebihan atau luas ke identitas di awal, yang melebihi hak akses yang diperlukan untuk pertama kali.
- Anda tidak meninjau dan menyesuaikan hak akses saat peran dan tanggung jawab identitas berubah dari waktu ke waktu.
- Anda membiarkan identitas yang tidak aktif atau identitas yang dihentikan dengan hak istimewa akses yang aktif. Hal ini akan meningkatkan risiko akses yang tidak sah.
- Anda tidak melakukan otomatisasi terhadap manajemen siklus hidup identitas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Kelola dan sesuaikan secara cermat hak akses yang Anda berikan kepada identitas (seperti pengguna, peran, grup) di sepanjang siklus hidup mereka. Siklus hidup ini mencakup fase onboarding awal, perubahan peran dan tanggung jawab yang berkelanjutan, dan akhirnya offboarding atau penghentian. Lakukan pengelolaan akses secara proaktif berdasarkan tahap siklus hidup untuk mempertahankan tingkat akses yang sesuai. Patuhi prinsip hak akses paling rendah untuk mengurangi munculnya risiko hak akses yang berlebihan atau tidak perlu.

Anda dapat mengelola siklus hidup IAM pengguna secara langsung di dalam Akun AWS, atau melalui federasi dari penyedia identitas tenaga kerja Anda ke AWS IAM Pusat Identitas. Untuk IAM pengguna, Anda dapat membuat, memodifikasi, dan menghapus pengguna dan izin terkait mereka di dalam Akun AWS. Untuk pengguna federasi, Anda dapat menggunakan Pusat IAM Identitas untuk mengelola siklus hidup mereka dengan menyinkronkan informasi pengguna dan grup dari penyedia identitas organisasi Anda menggunakan protokol System for Cross-domain Identity Management (). SCIM

SCIMadalah protokol standar terbuka untuk penyediaan otomatis dan deprovisioning identitas pengguna di berbagai sistem. Dengan mengintegrasikan penyedia identitas Anda dengan Pusat IAM Identitas menggunakanSCIM, Anda dapat secara otomatis menyinkronkan informasi pengguna dan grup, membantu memvalidasi bahwa hak akses diberikan, dimodifikasi, atau dicabut berdasarkan perubahan dalam sumber identitas otoritatif organisasi Anda.

Ketika peran dan tanggung jawab karyawan berubah dalam organisasi Anda, sesuaikan hak akses mereka sesuai keperluan. Anda dapat menggunakan set izin Pusat IAM Identitas untuk menentukan peran atau tanggung jawab pekerjaan yang berbeda dan mengaitkannya dengan IAM kebijakan dan izin yang sesuai. Saat peran seorang karyawan berubah, Anda dapat memperbarui kumpulan izin yang ditetapkan padanya untuk menyesuaikan dengan tanggung jawab baru mereka. Lakukan verifikasi bahwa mereka memiliki akses yang diperlukan dan sekaligus mematuhi prinsip hak akses paling rendah.

Langkah-langkah implementasi

1. Tentukan dan dokumentasikan proses siklus hidup manajemen akses, termasuk prosedur-prosedur yang harus dilakukan untuk memberikan akses awal, peninjauan berkala, dan offboarding.
2. Menerapkan batasan IAM peran, grup, dan izin untuk mengelola akses secara kolektif dan menegakkan tingkat akses maksimum yang diizinkan.
3. Integrasikan dengan penyedia identitas federasi (seperti Microsoft Active Directory, Okta, Ping Identity) sebagai sumber otoritatif untuk informasi pengguna dan grup menggunakan IAM Identity Center.
4. Gunakan SCIM protokol untuk menyinkronkan informasi pengguna dan grup dari penyedia identitas ke Identity Store Pusat IAM Identitas.
5. Buat set izin di Pusat IAM Identitas yang mewakili peran atau tanggung jawab pekerjaan yang berbeda dalam organisasi Anda. Tentukan IAM kebijakan dan izin yang sesuai untuk setiap set izin.
6. Implementasikan peninjauan akses secara rutin, pencabutan akses yang cepat, dan peningkatan berkelanjutan terhadap proses siklus hidup manajemen akses.
7. Berikan pelatihan dan pengetahuan kepada karyawan tentang praktik terbaik manajemen akses.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP04 Mengandalkan penyedia identitas terpusat

Dokumen terkait:

- Kelola sumber identitas Anda
- Mengelola identitas di Pusat IAM Identitas
- Menggunakan AWS Identity and Access Management Access Analyzer
- IAMPembuatan kebijakan Access Analyzer

Video terkait:

- AWS re:Inforce 2023 - Kelola akses sementara yang ditinggikan dengan Identity Center AWS IAM
- AWS re:invent 2022 - Sederhanakan akses tenaga kerja Anda yang ada dengan Identity Center IAM
- AWS re:invent 2022 - Memanfaatkan kekuatan IAM kebijakan & mengendalikan izin dengan Access Analyzer

SEC03-BP07 Menganalisis akses publik dan lintas akun

Pantau secara terus-menerus temuan yang menyoroti akses lintas akun dan publik. Kurangi akses publik dan akses lintas akun hanya ke sumber daya yang memerlukan akses ini.

Hasil yang diinginkan: Ketahui AWS sumber daya Anda mana yang dibagikan dan dengan siapa. Lakukan pemantauan dan audit secara terus-menerus terhadap sumber daya bersama untuk memastikan bahwa sumber daya tersebut hanya dibagikan kepada pengguna utama yang sah.

Anti-pola umum:

- Tidak menyimpan sebuah inventaris sumber daya bersama.
- Tidak mengikuti sebuah proses persetujuan akses lintas akun atau publik ke sumber daya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Jika akun Anda masuk AWS Organizations, Anda dapat memberikan akses ke sumber daya ke seluruh organisasi, unit organisasi tertentu, atau akun individual. Jika akun Anda bukan merupakan

anggota suatu organisasi, maka Anda dapat berbagi sumber daya dengan akun individu. Anda dapat memberikan akses lintas akun langsung menggunakan kebijakan berbasis sumber daya — misalnya, kebijakan bucket Amazon Simple Storage Service (Amazon S3) — atau dengan mengizinkan prinsipal di akun lain untuk berperan dalam akun Anda. IAM Saat menggunakan kebijakan sumber daya, verifikasi bahwa akses tersebut hanya diberikan kepada pengguna utama yang sah. Tentukan sebuah proses untuk menyetujui semua sumber daya yang diperlukan untuk tersedia secara publik.

AWS Identity and Access Management Access Analyzer menggunakan keamanan yang dapat dibuktikan untuk mengidentifikasi semua jalur akses ke sumber daya dari luar akun tersebut. Keamanan tersebut akan meninjau kebijakan sumber daya secara terus-menerus, dan melaporkan temuan akses lintas akun dan publik untuk memudahkan Anda dalam menganalisis potensi akses yang meluas. Pertimbangkan untuk mengonfigurasi IAM Access Analyzer AWS Organizations untuk memverifikasi bahwa Anda memiliki visibilitas ke semua akun Anda. IAMAccess Analyzer juga memungkinkan Anda untuk melihat pratinjau temuan sebelum menerapkan izin sumber daya. Hal ini akan memungkinkan Anda untuk memvalidasi bahwa perubahan kebijakan hanya memberikan akses lintas akun dan publik yang benar-benar diinginkan ke sumber daya Anda. Saat merancang untuk akses multi-akun, Anda dapat menggunakan kebijakan kepercayaan untuk mengontrol dalam kasus apa sebuah peran bisa diambil. Misalnya, Anda dapat menggunakan kunci kondisi `PrincipalOrgId` untuk menolak upaya pengambilan peran dari luar AWS Organizations Anda.

AWS Config dapat melaporkan sumber daya yang salah konfigurasi, dan melalui pemeriksaan AWS Config kebijakan, dapat mendeteksi sumber daya yang memiliki akses publik yang dikonfigurasi. Layanan seperti AWS Control Towerdan AWS Security Hubmenyederhanakan penyebaran kontrol detektif dan pagar pembatas AWS Organizations untuk mengidentifikasi dan memulihkan sumber daya yang terpapar publik. Misalnya, AWS Control Tower memiliki pagar pembatas terkelola yang dapat mendeteksi jika ada EBSsnapshot Amazon yang dapat dipulihkan oleh. Akun AWS

Langkah-langkah implementasi

- Pertimbangkan AWS Config untuk menggunakan for AWS Organizations: AWS Config memungkinkan Anda mengumpulkan temuan dari beberapa akun dalam akun administrator yang didelegasikan AWS Organizations ke akun administrator. Ini memberikan tampilan yang komprehensif, dan memungkinkan Anda untuk menyebarkan Aturan AWS Config di seluruh akun untuk mendeteksi sumber daya yang dapat diakses publik.
- Mengonfigurasi AWS Identity and Access Management Access Analyzer IAM Access Analyzer membantu Anda mengidentifikasi sumber daya di organisasi dan akun Anda, seperti bucket Amazon S3 IAM atau peran yang dibagikan dengan entitas eksternal.

- Gunakan remediasi otomatis AWS Config untuk menanggapi perubahan konfigurasi akses publik bucket Amazon S3: Anda dapat secara otomatis mengaktifkan pengaturan blokir akses publik untuk bucket Amazon S3.
- Terapkan pemantauan dan peringatan untuk mengidentifikasi apakah bucket Amazon S3 telah menjadi publik: Anda harus memiliki pemantauan dan peringatan yang tersedia untuk mengidentifikasi kapan Blokir Akses Publik Amazon S3 dimatikan, dan apakah bucket Amazon S3 sudah publik atau tidak. Selain itu, jika Anda menggunakan AWS Organizations, Anda dapat membuat kebijakan kontrol layanan yang mencegah perubahan pada kebijakan akses publik Amazon S3. AWS Trusted Advisor memeriksa bucket Amazon S3 yang memiliki izin akses terbuka. Izin bucket yang memberikan, mengunggah, atau menghapus akses ke semua orang akan menciptakan potensi masalah keamanan dengan mengizinkan siapa pun untuk menambahkan, mengubah, atau menghapus item yang dalam sebuah bucket. Trusted Advisor Pemeriksaan memeriksa izin bucket eksplisit dan kebijakan bucket terkait yang mungkin mengganti izin bucket. Anda juga dapat menggunakan AWS Config untuk memantau bucket Amazon S3 Anda untuk akses publik. Untuk informasi selengkapnya, lihat Cara Menggunakan AWS Config untuk Memantau dan Menanggapi Bucket Amazon S3 yang Memungkinkan Akses Publik. Saat meninjau akses, sebaiknya Anda mempertimbangkan jenis data apa yang terkandung dalam bucket Amazon S3. Amazon Macie membantu menemukan dan melindungi data sensitif, seperti PIIPHI, dan kredensil, seperti pribadi atau kunci. AWS

Sumber daya

Dokumen terkait:

- Menggunakan AWS Identity and Access Management Access Analyzer
- AWS Control Tower kontrol perpustakaan
- AWS Standar Praktik Terbaik Keamanan Dasar
- Aturan Terkelola AWS Config
- Referensi pemeriksaan AWS Trusted Advisor
- Memantau hasil AWS Trusted Advisor pemeriksaan dengan Amazon EventBridge
- Mengelola AWS Config Aturan di Semua Akun di Organisasi Anda
- AWS Config dan AWS Organizations
- Jadikan Anda AMI tersedia untuk umum untuk digunakan di Amazon EC2

Video terkait:

- Praktik Terbaik untuk mengamankan lingkungan multiakun Anda
- Menyelam Jauh ke dalam IAM Access Analyzer

SEC03-BP08 Bagikan sumber daya dengan aman di dalam organisasi Anda

Seiring dengan meningkatnya jumlah beban kerja, Anda mungkin perlu membagikan akses ke sumber daya dalam beban kerja tersebut atau berulang kali menyediakan sumber daya itu di seluruh akun. Anda mungkin memiliki konsep untuk membagi lingkungan Anda dalam beberapa kelompok, seperti lingkungan pengembangan, pengujian, dan lingkungan produksi. Namun demikian, konsep pemisahan ini tidak akan membatasi Anda untuk berbagi secara aman. Dengan membagikan komponen-komponen yang tumpang tindih, Anda dapat mengurangi overhead operasional dan memungkinkan pengalaman yang konsisten tanpa harus menebak-nebak mengenai apa yang terlewatkan sekaligus membuat sumber daya yang sama berulang kali.

Hasil yang diinginkan: Mengurangi akses yang tidak diinginkan sekecil hingga sekecil mungkin dengan menggunakan metode yang aman untuk berbagi sumber daya dalam organisasi Anda, dan membantu inisiatif pencegahan kehilangan data Anda. Mengurangi overhead operasional daripada mengelola komponen satu per satu, akan mengurangi kesalahan yang diakibatkan oleh pembuatan komponen yang sama secara manual berulang kali, serta meningkatkan skalabilitas beban kerja. Anda dapat memperoleh manfaat dari pengurangan waktu hingga resolusi di skenario kegagalan multi-titik, dan meningkatkan keyakinan Anda dalam menentukan kapan sebuah komponen tidak diperlukan lagi. Untuk panduan preskriptif tentang cara melakukan analisis sumber daya bersama secara eksternal, silakan lihat SEC03-BP07 Menganalisis akses publik dan lintas akun.

Anti-pola umum:

- Tidak ada proses untuk melakukan pemantauan secara terus-menerus dan dan memberikan peringatan otomatis mengenai pembagian secara eksternal yang tidak terduga.
- Tidak ada acuan terkait apa yang boleh dan tidak boleh dibagikan.
- Kebijakan terbuka luas secara default, bukannya berbagi secara eksplisit ketika diperlukan.
- Membuat sumber daya dasar yang tumpang tindih secara manual, saat diperlukan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Rancang arsitektur pola dan kontrol akses Anda untuk mengelola penggunaan sumber daya yang dibagikan secara aman dan hanya dengan entitas-entitas yang tepercaya. Lakukan pemantauan

terhadap sumber daya yang dibagikan dan peninjauan terhadap akses sumber daya yang dibagikan secara terus-menerus serta tetap waspada terhadap adanya pembagian yang tidak terduga atau tidak tepat. Tinjau Analisis akses publik dan akses lintas akun untuk membantu Anda menetapkan tata kelola untuk mengurangi akses eksternal hanya ke sumber daya yang memerlukannya, dan untuk membuat proses untuk melakukan pemantauan secara terus menerus dan memberi peringatan secara otomatis.

Berbagi lintas akun di dalamnya AWS Organizations didukung oleh sejumlah AWS layanan, seperti AWS Security Hub, Amazon GuardDuty, dan AWS Backup. Layanan-layanan ini akan memungkinkan data untuk dibagikan ke akun pusat, dapat diakses dari akun pusat, atau mengelola sumber daya dan data dari akun pusat. Misalnya, AWS Security Hub dapat mentransfer temuan dari akun individu ke akun pusat tempat Anda dapat melihat semua temuan. AWS Backup dapat mengambil cadangan untuk sumber daya dan membagikannya di seluruh akun. Anda dapat menggunakan AWS Resource Access Manager(AWS RAM) untuk berbagi sumber daya umum lainnya, seperti VPCsubnet dan lampiran Transit Gateway AWS Network Firewall, atau pipeline Amazon SageMaker .

Untuk membatasi akun Anda agar hanya berbagi sumber daya dalam organisasi Anda, gunakan kebijakan kontrol layanan (SCPs) untuk mencegah akses ke prinsipal eksternal. Saat berbagi sumber daya, gabungkan kontrol berbasis identitas dan kontrol jaringan untuk membuat perimeter data bagi organisasi Anda guna membantu Anda melindungi dari akses yang tidak diinginkan. Perimeter data adalah kumpulan pagar pembatas preventif untuk membantu Anda memverifikasi bahwa hanya identitas yang Anda percaya saja yang mengakses sumber daya tepercaya dari jaringan yang dikenal. Kontrol ini akan menetapkan batas yang sesuai terkait sumber daya apa yang dapat dibagikan, serta mencegah dibagikannya atau bocornya sumber daya yang tidak semestinya terjadi. Misalnya, sebagai bagian dari perimeter data, Anda dapat menggunakan kebijakan VPC titik akhir dan `AWS:PrincipalOrgId` kondisi untuk memastikan identitas yang mengakses bucket Amazon S3 milik organisasi Anda. Penting untuk dicatat bahwa SCPstidak berlaku untuk peran terkait layanan atau prinsip AWS layanan.

Saat menggunakan Amazon S3, matikan ACLs bucket Amazon S3 Anda dan IAM gunakan kebijakan untuk menentukan kontrol akses. Untuk membatasi akses ke Amazon S3 asal dari CloudFrontAmazon, bermigrasi dari origin access identity OAI () ke origin access control OAC () yang mendukung fitur tambahan termasuk enkripsi sisi server dengan. AWS Key Management Service

Dalam beberapa kasus, Anda mungkin ingin mengizinkan pembagian sumber daya ke luar organisasi Anda atau memberikan pihak ketiga akses ke sumber daya Anda. Untuk panduan preskriptif tentang cara mengelola izin untuk berbagi sumber daya secara eksternal, lihat Manajemen izin.

Langkah-langkah implementasi

1. Gunakan AWS Organizations.

   AWS Organizations adalah layanan manajemen akun yang memungkinkan Anda untuk mengkonsolidasikan beberapa Akun AWS ke dalam organisasi yang Anda buat dan kelola secara terpusat. Anda dapat mengelompokkan akun Anda ke dalam unit organisasi (OUs) dan melampirkan kebijakan yang berbeda ke setiap OU untuk membantu Anda memenuhi kebutuhan anggaran, keamanan, dan kepatuhan Anda. Anda juga dapat mengontrol bagaimana kecerdasan AWS buatan (AI) dan layanan pembelajaran mesin (ML) dapat mengumpulkan dan menyimpan data, dan menggunakan manajemen multi-akun dari AWS layanan yang terintegrasi dengan Organizations.

2. Integrasikan AWS Organizations dengan AWS layanan.

   Saat Anda menggunakan AWS layanan untuk melakukan tugas atas nama Anda di akun anggota organisasi Anda, AWS Organizations buat peran IAM terkait layanan (SLR) untuk layanan tersebut di setiap akun anggota. Anda harus mengelola akses tepercaya menggunakan AWS APIs,, atau AWS CLI. AWS Management Console Untuk panduan preskriptif tentang mengaktifkan akses tepercaya, lihat Menggunakan AWS Organizations dengan AWS layanan dan AWS layanan lain yang dapat Anda gunakan dengan Organizations.

3. Membuat perimeter data.

   AWS Perimeter biasanya direpresentasikan sebagai organisasi yang dikelola oleh AWS Organizations. Seiring dengan jaringan dan sistem lokal, mengakses AWS sumber daya adalah apa yang banyak dianggap sebagai perimeter My. AWS Perimeter bertujuan untuk memverifikasi bahwa akses diizinkan jika identitasnya dipercaya, sumber dayanya dipercaya, dan jaringannya dikenal.

   a. Menentukan dan mengimplementasikan perimeter.

      Ikuti langkah-langkah yang dijelaskan dalam implementasi Perimeter dalam Membangun Perimeter pada AWS whitepaper untuk setiap kondisi otorisasi. Untuk panduan preskriptif tentang cara melindungi lapisan jaringan, lihat Melindungi jaringan.

   b. Tetap pantau dan waspada.

      AWS Identity and Access Management Access Analyzer dapat membantu Anda mengidentifikasi sumber daya di akun dan organisasi Anda yang dibagi dengan entitas eksternal. Anda dapat mengintegrasikan IAMAccess Analyzer AWS Security Hub untuk mengirim dan mengumpulkan temuan untuk sumber daya dari IAM Access Analyzer ke Security Hub untuk membantu menganalisis postur keamanan lingkungan Anda. Untuk

mengintegrasikan, aktifkan IAM Access Analyzer dan Security Hub di setiap Wilayah di setiap akun. Anda juga dapat menggunakan Aturan AWS Config untuk mengaudit konfigurasi dan memperingatkan pihak yang sesuai menggunakan AWS Chatbot dengan AWS Security Hub. Anda kemudian dapat menggunakan dokumen Otomasi AWS Systems Manager untuk memulihkan sumber daya yang tidak sesuai.

c. Untuk panduan preskriptif tentang pemantauan dan peringatan berkelanjutan tentang sumber daya yang dibagikan secara eksternal, lihat Menganalisis akses publik dan akses lintas akun.

4. Gunakan berbagi sumber daya dalam AWS layanan dan batasi yang sesuai.

   Banyak AWS layanan memungkinkan Anda berbagi sumber daya dengan akun lain, atau menargetkan sumber daya di akun lain, seperti Amazon Machine Images (AMIs) dan AWS Resource Access Manager (AWS RAM). Batasi `ModifyImageAttribute` API untuk menentukan akun tepercaya untuk berbagi AMI dengan. Tentukan `ram:RequestedAllowsExternalPrincipals` kondisi saat menggunakan AWS RAM untuk membatasi berbagi ke organisasi Anda saja, untuk membantu mencegah akses dari identitas yang tidak tepercaya. Untuk panduan dan pertimbangan preskriptif, lihat Berbagi sumber daya dan target eksternal.

5. Gunakan AWS RAM untuk berbagi dengan aman di akun atau dengan yang lain Akun AWS.

   AWS RAM akan membantu Anda secara aman membagikan sumber daya yang Anda buat kepada peran dan pengguna di akun Anda, serta dengan Akun AWS lainnya. Dalam lingkungan multi-akun, AWS RAM memungkinkan Anda membuat sumber daya sekali dan membagikannya dengan akun lain. Pendekatan ini membantu mengurangi overhead operasional Anda sekaligus memberikan konsistensi, visibilitas, dan auditabilitas melalui integrasi dengan CloudWatch Amazon AWS CloudTrail dan, yang tidak Anda terima saat menggunakan akses lintas akun.

   Jika Anda memiliki sumber daya yang Anda bagikan sebelumnya menggunakan kebijakan berbasis sumber daya, Anda dapat menggunakan `PromoteResourceShareCreatedFromPolicy`APIatau yang setara untuk mempromosikan pembagian sumber daya ke pembagian sumber daya penuh. AWS RAM

   Dalam beberapa kasus, Anda mungkin memerlukan beberapa langkah tambahan yang harus dilakukan untuk berbagi sumber daya. Misalnya, untuk membagikan snapshot terenkripsi, Anda perlu membagikan kunci. AWS KMS

Sumber daya

Praktik-praktik terbaik terkait:

- SEC03-BP07 Menganalisis akses publik dan lintas akun
- SEC03-BP09 Berbagi sumber daya secara aman dengan pihak ketiga
- SEC05-BP01 Buat lapisan jaringan

Dokumen terkait:

- Pemilik bucket yang memberikan izin lintas akun untuk objek yang bukan miliknya
- Cara menggunakan Kebijakan Kepercayaan dengan IAM
- Membangun Perimeter Data di AWS
- Cara menggunakan ID eksternal saat memberikan akses pihak ketiga ke sumber daya Anda AWS
- AWS Layanan yang dapat Anda gunakan dengan AWS Organizations
- Membuat perimeter data pada AWS: Izinkan hanya identitas tepercaya untuk mengakses data perusahaan

Video terkait:

- Akses Terperinci dengan AWS Resource Access Manager
- Mengamankan perimeter data Anda dengan titik akhir VPC
- Membangun perimeter data pada AWS

Alat terkait:

- Contoh Kebijakan Perimeter Data

SEC03-BP09 Berbagi sumber daya secara aman dengan pihak ketiga

Keamanan lingkungan cloud tidak berhenti di organisasi Anda. Organisasi Anda mungkin menggunakan pihak ketiga untuk mengelola sebagian data Anda. Manajemen izin untuk sistem yang dikelola pihak ketiga harus mengikuti praktik just-in-time akses menggunakan prinsip hak istimewa paling sedikit dengan kredensi sementara. Melalui kerja sama dengan pihak ketiga, Anda dapat mengurangi cakupan dampak dan risiko yang mungkin dimunculkan oleh akses yang tidak diinginkan.

Hasil yang diinginkan: Kredensi jangka panjang AWS Identity and Access Management (IAM), kunci IAM akses, dan kunci rahasia yang terkait dengan pengguna dapat digunakan oleh siapa saja selama kredensialnya valid dan aktif. Menggunakan IAM peran dan kredensi sementara membantu Anda meningkatkan sikap keamanan Anda secara keseluruhan dengan mengurangi upaya untuk mempertahankan kredensi jangka panjang, termasuk manajemen dan overhead operasional dari detail sensitif tersebut. Dengan menggunakan pengenal unik universal (UUID) untuk ID eksternal dalam kebijakan IAM kepercayaan, dan menjaga kebijakan yang melekat pada IAM peran di bawah kendali Anda, Anda dapat mengaudit dan memverifikasi bahwa akses yang diberikan kepada pihak ketiga tidak terlalu permisif. IAM Untuk panduan preskriptif tentang cara melakukan analisis sumber daya bersama secara eksternal, silakan lihat SEC03-BP07 Menganalisis akses publik dan lintas akun.

Anti-pola umum:

- Menggunakan kebijakan IAM kepercayaan default tanpa syarat apa pun.
- Menggunakan IAM kredensi jangka panjang dan kunci akses.
- Menggunakan kembali eksternalIDs.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Anda mungkin ingin mengizinkan berbagi sumber daya di luar AWS Organizations atau memberikan akses pihak ketiga ke akun Anda. Misalnya, pihak ketiga mungkin menyediakan sebuah solusi pemantauan yang perlu mengakses sumber daya yang ada di akun Anda. Dalam kasus tersebut, buat peran IAM lintas akun dengan hanya hak istimewa yang dibutuhkan oleh pihak ketiga. Selain itu, tentukan kebijakan kepercayaan dengan menggunakan kondisi ID eksternal. Saat menggunakan ID eksternal, Anda atau pihak ketiga dapat membuat sebuah ID unik untuk setiap pelanggan, pihak ketiga, atau penghunian. Setelah dibuat, ID unik tersebut tidak boleh dikontrol oleh siapa pun selain Anda. Pihak ketiga harus mengimplementasikan sebuah proses untuk memberikan ID eksternal melalui cara yang aman, dapat diaudit, dan diproduksi kembali.

Anda juga dapat menggunakan IAMRoles Anywhere untuk mengelola IAM peran untuk aplikasi di luar penggunaan AWS itu AWS APIs.

Hapus peran tersebut jika pihak ketiga sudah tidak perlu mengakses lingkungan Anda. Hindari menyediakan kredensial jangka panjang kepada pihak ketiga. Pertahankan kesadaran akan AWS layanan lain yang mendukung berbagi. Misalnya, AWS Well-Architected Tool memungkinkan berbagi

beban kerja dengan orang lain Akun AWS, dan AWS Resource Access Managermembantu Anda berbagi AWS sumber daya yang Anda miliki dengan aman dengan akun lain.

Langkah-langkah implementasi

1. Gunakan peran lintas akun untuk memberikan akses kepada akun eksternal.

   Peran lintas akun akan mengurangi jumlah informasi sensitif yang disimpan oleh akun eksternal dan pihak ketiga yang diperlukan untuk melayani pelanggan mereka. Peran lintas akun memungkinkan Anda memberikan akses ke AWS sumber daya di akun Anda dengan aman kepada pihak ketiga, seperti AWS Partner s atau akun lain di organisasi Anda, sambil mempertahankan kemampuan untuk mengelola dan mengaudit akses tersebut.

   Pihak ketiga mungkin memberikan layanan kepada Anda dari sebuah infrastruktur hibrida atau menarik data ke lokasi di luar situs. IAMRoles Anywhere membantu Anda mengizinkan beban kerja pihak ketiga untuk berinteraksi secara aman dengan AWS beban kerja Anda dan selanjutnya mengurangi kebutuhan akan kredensi jangka panjang.

   Anda tidak boleh menggunakan kredensial jangka panjang, atau kunci akses yang dikaitkan dengan pengguna, untuk menyediakan akses kepada akun eksternal. Sebaiknya gunakan peran lintas akun untuk memberikan akses lintas akun sebagai gantinya.
2. Gunakan ID eksternal dengan pihak ketiga.

   Menggunakan ID eksternal memungkinkan Anda untuk menentukan siapa yang dapat mengambil peran dalam kebijakan IAM kepercayaan. Kebijakan kepercayaan mungkin mengharuskan pengguna yang mengambil peran menegaskan persyaratan dan target operasi. Itu ini juga memberikan cara bagi pemilik akun untuk mengizinkan peran tersebut untuk diasumsikan hanya dalam keadaan tertentu. Fungsi utama dari ID eksternal adalah mengatasi dan mencegah masalah deputi yang membingungkan.

   Gunakan ID eksternal jika Anda adalah Akun AWS pemilik dan Anda telah mengonfigurasi peran untuk pihak ketiga yang mengakses orang lain Akun AWS selain milik Anda, atau ketika Anda berada dalam posisi mengambil peran atas nama pelanggan yang berbeda. Bekerja dengan pihak ketiga Anda atau AWS Partner untuk menetapkan kondisi ID eksternal untuk disertakan dalam kebijakan IAM kepercayaan.
3. Gunakan eksternal IDs yang unik secara universal.

   Menerapkan proses yang menghasilkan nilai unik acak untuk ID eksternal, seperti pengidentifikasi unik universal ()UUID. Pihak ketiga yang menggunakan kembali eksternal IDs di seluruh

pelanggan yang berbeda tidak mengatasi masalah wakil yang membingungkan, karena pelanggan A mungkin dapat melihat data pelanggan B dengan menggunakan peran ARN pelanggan B bersama dengan ID eksternal yang digandakan. Dalam lingkungan multi-tenant, di mana pihak ketiga mendukung beberapa pelanggan dengan berbeda Akun AWS, pihak ketiga harus menggunakan ID unik yang berbeda sebagai ID eksternal untuk masing-masing. Akun AWS Pihak ketiga bertanggung jawab untuk mendeteksi duplikat eksternal IDs dan memetakan setiap pelanggan dengan aman ke ID eksternal masing-masing. Pihak ketiga harus menguji untuk memverifikasi bahwa pihaknya hanya dapat mengambil peran saat menentukan ID eksternal. Pihak ketiga harus menahan diri dari menyimpan peran pelanggan ARN dan ID eksternal sampai ID eksternal diperlukan.

ID eksternal bukan sesuatu yang rahasia, tetapi ID Eksternal tidak boleh berupa nilai yang mudah ditebak, seperti nomor telepon, nama, atau ID akun. Buatlah ID eksternal menjadi bidang hanya baca sehingga ID eksternal tersebut tidak dapat diubah untuk tujuan meniru penyiapan.

Anda atau pihak ketiga dapat membuat ID eksternal. Bentuklah sebuah proses untuk menentukan siapa yang bertanggung jawab dalam pembuatan ID. Siapa pun entitas pembuat ID eksternalnya, pihak ketiga menjaga keunikan dan formatnya tetap konsisten untuk semua pelanggan.

4. Menghilangkan kredensial jangka panjang yang disediakan pelanggan.

   Menghentikan penggunaan kredensi jangka panjang dan gunakan peran lintas akun atau Peran Di Mana Saja. IAM Jika Anda harus menggunakan kredensial jangka panjang, buatlah sebuah rencana untuk bermigrasi ke akses berbasis peran. Untuk mendapatkan detail tentang cara mengelola kunci, silakan lihat Manajemen Identitas. Juga bekerja dengan Akun AWS tim Anda dan pihak ketiga untuk membuat runbook mitigasi risiko. Untuk panduan preskriptif mengenai cara menanggapi dan mengurangi potensi dampak insiden keamanan, silakan lihat Respons insiden.

5. Verifikasi bahwa pengaturan memiliki panduan preskriptif atau otomatis.

   Kebijakan yang dibuat untuk akses lintas akun di akun Anda harus mengikuti prinsip hak akses paling rendah. Pihak ketiga harus menyediakan dokumen kebijakan peran atau mekanisme penyiapan otomatis yang menggunakan AWS CloudFormation templat atau yang setara untuk Anda. Hal ini akan mengurangi adanya potensi kesalahan yang bisa terjadi pada pembuatan kebijakan manual dan menyediakan jejak yang dapat diaudit. Untuk informasi selengkapnya tentang penggunaan AWS CloudFormation templat untuk membuat peran lintas akun, lihat Peran Lintas Akun.

   Pihak ketiga harus menyediakan sebuah mekanisme penyiapan otomatis yang dapat diaudit. Namun, dengan dokumen kebijakan peran yang menguraikan akses yang diperlukan, Anda harus

mengotomatiskan penyiapan peran tersebut. Menggunakan AWS CloudFormation template atau yang setara, Anda harus memantau perubahan dengan deteksi drift sebagai bagian dari praktik audit.

6. Akun untuk perubahan.

   Struktur akun Anda, kebutuhan Anda terhadap pihak ketiga, atau penawaran layanan yang disediakan dapat berubah. Anda harus mengantisipasi perubahan dan kegagalan, dan membuat rencana yang sesuai dengan orang, proses, dan teknologi yang tepat. Lakukan audit tingkat akses yang Anda berikan secara berkala, dan terapkan metode deteksi yang akan memberikan Anda peringatan tentang perubahan yang tidak terduga. Memantau dan mengaudit penggunaan peran dan datastore eksternal. IDs Anda harus bersiap untuk mencabut akses pihak ketiga, baik untuk sementara atau secara permanen, jika terjadi perubahan atau pola akses yang tidak terduga. Selain itu, ukur dampak atas operasi pencabutan Anda, termasuk waktu yang diperlukan untuk melakukannya, orang yang terlibat, biaya, dan dampaknya terhadap sumber daya lainnya.

   Untuk panduan preskriptif mengenai metode deteksi, silakan lihat Praktik terbaik deteksi.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP02 Gunakan kredenal sementara
- SEC03-BP05 Tentukan pagar pembatas izin untuk organisasi Anda
- SEC03-BP06 Mengelola akses berdasarkan siklus hidup
- SEC03-BP07 Menganalisis akses publik dan lintas akun
- SEC04 Deteksi

Dokumen terkait:

- Pemilik bucket yang memberikan izin lintas akun untuk objek yang bukan miliknya
- Cara menggunakan kebijakan kepercayaan dengan IAM peran
- Mendelegasikan akses di seluruh Akun AWS menggunakan peran IAM
- Bagaimana cara mengakses sumber daya di Akun AWS penggunaan lainIAM?
- Praktik terbaik keamanan di IAM
- Logika evaluasi kebijakan lintas akun

- Cara menggunakan ID eksternal saat memberikan akses ke AWS sumber daya Anda kepada pihak ketiga
- Mengumpulkan Informasi dari AWS CloudFormation Sumber Daya yang Dibuat di Akun Eksternal dengan Sumber Daya Kustom
- Menggunakan ID Eksternal dengan Aman untuk Mengakses AWS Akun yang Dimiliki oleh Orang Lain
- IAMPerluas peran ke beban kerja di luar IAM dengan IAM Peran Di Mana Saja

Video terkait:

- Bagaimana cara mengizinkan pengguna atau peran dalam Akun AWS akses terpisah ke saya Akun AWS?
- AWS Re:Invent 2018: Menjadi Master IAM Kebijakan dalam 60 Menit atau Kurang
- AWS Pusat Pengetahuan Langsung: Praktik IAM Terbaik dan Keputusan Desain

Contoh terkait:

- Well-Architected Lab - Asumsi peran lintas IAM akun Lambda (Level 300)
- Mengonfigurasikan akses lintas akun ke Amazon DynamoDB
- AWS STS Alat Kueri Jaringan

## Deteksi

Pertanyaan

- SEC4. Bagaimana cara mendeteksi dan menyelidiki peristiwa keamanan?

### SEC4. Bagaimana cara mendeteksi dan menyelidiki peristiwa keamanan?

Catat dan analisis peristiwa dari log dan metrik untuk mendapatkan visibilitas. Ambil tindakan atas peristiwa keamanan dan potensi ancaman untuk membantu mengamankan beban kerja Anda.

Praktik terbaik

- SEC04-BP01 Mengkonfigurasi layanan dan pencatatan aplikasi
- SEC04-BP02 Tangkap log, temuan, dan metrik di lokasi standar

- SEC04-BP03 Menghubungkan dan memperkaya peringatan keamanan
- SEC04-BP04 Memulai remediasi untuk sumber daya yang tidak sesuai

SEC04-BP01 Mengkonfigurasi layanan dan pencatatan aplikasi

Pertahankan log peristiwa keamanan dari layanan dan aplikasi. Ini adalah prinsip dasar keamanan untuk audit, investigasi, dan kasus penggunaan operasional, dan persyaratan keamanan umum yang didorong oleh standar, kebijakan, dan prosedur tata kelola, risiko, dan kepatuhan (GRC).

Hasil yang diinginkan: Organisasi harus dapat secara andal dan konsisten mengambil log peristiwa keamanan dari AWS layanan dan aplikasi secara tepat waktu ketika diperlukan untuk memenuhi proses atau kewajiban internal, seperti respons insiden keamanan. Sebaiknya pusatkan log untuk mendapatkan hasil operasional yang lebih baik.

Anti-pola umum:

- Log disimpan tanpa batas waktu yang jelas atau dihapus terlalu cepat.
- Semua orang dapat mengakses log.
- Sepenuhnya menggunakan proses manual untuk tata kelola dan penggunaan log.
- Menyimpan setiap jenis log untuk berjaga-jaga jika diperlukan.
- Memeriksa integritas log hanya jika diperlukan.

Manfaat menetapkan praktik terbaik ini: Menerapkan mekanisme analisis akar penyebab (RCA) untuk insiden keamanan dan sumber bukti untuk kewajiban tata kelola, risiko, dan kepatuhan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Selama penyelidikan keamanan atau kasus penggunaan lain berdasarkan kebutuhan Anda, Anda harus dapat meninjau log yang relevan untuk mencatat dan memahami seluruh cakupan serta lini masa insiden. Log juga diperlukan untuk pembuatan peringatan yang mengindikasikan bahwa tindakan-tindakan tertentu telah terjadi. Sangat penting bagi Anda untuk memilih, menyalakan, menyimpan, dan menyiapkan mekanisme kueri dan pengambilan serta pembuatan peringatan.

Langkah-langkah implementasi

- Pilih dan gunakan sumber log. Sebelum melakukan penyelidikan keamanan, Anda perlu mengambil log yang relevan untuk merekonstruksi aktivitas secara surut di Akun AWS. Pilih sumber log yang relevan dengan beban kerja Anda.

  Kriteria pemilihan sumber log harus didasarkan pada kasus penggunaan yang diperlukan oleh bisnis Anda. Buat jejak untuk setiap Akun AWS penggunaan AWS CloudTrail atau AWS Organizations jejak, dan konfigurasikan bucket Amazon S3 untuknya.

  AWS CloudTrail adalah layanan logging yang melacak API panggilan yang dilakukan terhadap aktivitas AWS layanan Akun AWS penangkapan. Ini diaktifkan secara default dengan retensi acara manajemen selama 90 hari yang dapat diambil melalui riwayat CloudTrail Acara menggunakan, file AWS Management Console, atau file AWS CLI. AWS SDK Untuk retensi dan visibilitas peristiwa data yang lebih lama, buat CloudTrail jejak dan kaitkan dengan bucket Amazon S3, dan opsional dengan grup log Amazon CloudWatch . Atau, Anda dapat membuat CloudTrail Danau, yang menyimpan CloudTrail log hingga tujuh tahun dan menyediakan fasilitas kueri SQL berbasis

  AWS merekomendasikan agar pelanggan menggunakan pengaktifan VPC lalu lintas jaringan dan DNS log menggunakan VPCFlow Logs dan log kueri penyelesai Amazon Route 53, masing-masing, dan mengalirkannya ke bucket Amazon S3 atau grup log. CloudWatch Anda dapat membuat VPC log aliran untukVPC, subnet, atau antarmuka jaringan. Untuk VPC Flow Logs, Anda dapat selektif tentang bagaimana dan di mana Anda menggunakan Flow Logs untuk mengurangi biaya.

  AWS CloudTrail Log, Log VPC Aliran, dan log kueri resolver Route 53 adalah sumber pencatatan dasar untuk mendukung penyelidikan keamanan. AWS Anda juga dapat menggunakan Amazon Security Lake untuk mengumpulkan, menormalkan, dan menyimpan data log ini dalam format Apache Parquet dan Open Cybersecurity Schema Framework (OCSF), yang siap untuk kueri. Security Lake juga mendukung AWS log dan log lain dari sumber pihak ketiga.

  AWS layanan dapat menghasilkan log yang tidak diambil oleh sumber log dasar, seperti log Elastic Load Balancing, log, AWS WAF log AWS Config perekam, temuan Amazon, log audit GuardDuty Amazon Elastic Kubernetes Service (Amazon), dan sistem operasi EKS instans Amazon dan log aplikasi. EC2 Untuk melihat daftar lengkap opsi pencatatan dan pemantauan, silakan lihat Lampiran A: Definisi kemampuan cloud — Pencatatan Log dan Peristiwa yang ada di Panduan Respons Insiden Keamanan AWS.
- Kemampuan pencatatan penelitian untuk setiap AWS layanan dan aplikasi: Setiap AWS layanan dan aplikasi memberi Anda opsi untuk penyimpanan log, yang masing-masing dengan kemampuan retensi dan siklus hidupnya sendiri. Dua layanan penyimpanan log yang paling umum adalah

Amazon Simple Storage Service (Amazon S3) dan Amazon. CloudWatch Untuk periode retensi yang panjang, sebaiknya gunakan Amazon S3 untuk mendapatkan efektivitas biaya dan kemampuan siklus hidup yang fleksibel. Jika opsi logging utama adalah Amazon CloudWatch Logs, sebagai opsi, Anda harus mempertimbangkan pengarsipan log yang jarang diakses ke Amazon S3.

- Pilih penyimpanan log: Pilihan penyimpanan log umumnya terkait dengan alat kueri yang Anda gunakan, kemampuan retensi, pemahaman, dan biaya. Opsi utama untuk penyimpanan log adalah bucket Amazon S3 atau grup CloudWatch Log.

  Bucket Amazon S3 menyediakan penyimpanan yang tahan lama dan hemat biaya, dengan kebijakan siklus hidup opsional. Log yang disimpan di dalam bucket Amazon S3 dapat dikueri secara native dengan menggunakan layanan-layanan seperti Amazon Athena.

  Grup CloudWatch log menyediakan penyimpanan yang tahan lama dan fasilitas kueri bawaan melalui Wawasan CloudWatch Log.
- Identifikasi penyimpanan log yang sesuai: Saat menggunakan bucket Amazon S3 atau grup CloudWatch log untuk menyimpan log, Anda harus menetapkan siklus hidup yang memadai untuk setiap sumber log guna mengoptimalkan biaya penyimpanan dan pengambilan. Pada umumnya, para pelanggan memiliki waktu antara tiga bulan hingga satu tahun untuk melakukan kueri log, dengan periode retensi hingga tujuh tahun. Pilihan ketersediaan dan retensi harus selaras dengan persyaratan keamanan Anda serta gabungan mandat hukum, peraturan, dan bisnis.
- Gunakan pencatatan untuk setiap AWS layanan dan aplikasi dengan kebijakan retensi dan siklus hidup yang tepat: Untuk setiap AWS layanan atau aplikasi di organisasi Anda, cari panduan konfigurasi pencatatan khusus:
  - Konfigurasikan AWS CloudTrail Trail
  - Konfigurasikan VPC Log Aliran
  - Konfigurasikan Amazon GuardDuty Finding Export
  - Konfigurasikan AWS Config perekaman
  - Konfigurasikan ACL lalu lintas AWS WAF web
  - Konfigurasikan log lalu lintas AWS Network Firewall jaringan
  - Konfigurasikan log akses Penyeimbangan Beban Elastis
  - Konfigurasikan log kueri Amazon Route 53 Resolver
  - Konfigurasikan RDS log Amazon
  - Konfigurasikan log Pesawat EKS Kontrol Amazon

  - Mengonfigurasi CloudWatch agen Amazon untuk EC2 instans Amazon dan server lokal
- Pilih dan terapkan mekanisme kueri untuk log: Untuk kueri log, Anda dapat menggunakan Wawasan CloudWatch Log untuk data yang disimpan dalam grup CloudWatch log, serta Amazon Athena dan Amazon OpenSearch Service untuk data yang disimpan di Amazon S3. Anda juga dapat menggunakan alat kueri pihak ketiga seperti informasi keamanan dan layanan manajemen acara (SIEM).

  Proses untuk memilih alat kueri log harus mempertimbangkan aspek orang, proses, dan teknologi dalam operasi keamanan Anda. Pilihlah sebuah alat yang memenuhi persyaratan operasional, bisnis, dan keamanan, serta dapat diakses dan dipelihara dalam jangka panjang. Perlu diingat bahwa alat kueri log bekerja secara optimal ketika jumlah log yang akan dipindai tidak melebihi batas alat. Tidak jarang terdapat beberapa alat kueri karena adanya kendala biaya atau teknis.

  Misalnya, Anda mungkin menggunakan informasi keamanan pihak ketiga dan alat manajemen peristiwa (SIEM) untuk melakukan kueri selama 90 hari terakhir data, tetapi gunakan Athena untuk melakukan kueri di luar 90 hari karena biaya penyerapan log dari a. SIEM Terlepas dari implementasi, pastikan pendekatan Anda akan meminimalkan jumlah alat yang diperlukan untuk memaksimalkan efisiensi operasional, khususnya selama penyelidikan peristiwa keamanan.
- Gunakan log untuk peringatan: AWS menyediakan peringatan melalui beberapa layanan keamanan:
  - AWS Configmemantau dan mencatat konfigurasi AWS sumber daya Anda dan memungkinkan Anda mengotomatiskan evaluasi dan remediasi terhadap konfigurasi yang diinginkan.
  - Amazon GuardDuty adalah layanan deteksi ancaman yang terus memantau aktivitas berbahaya dan perilaku tidak sah untuk melindungi Anda Akun AWS dan beban kerja. GuardDuty menyerap, mengumpulkan, dan menganalisis informasi dari sumber, seperti peristiwa AWS CloudTrail manajemen dan data, DNS log, Log VPC Aliran, dan log Audit AmazonEKS. GuardDuty menarik aliran data independen langsung dari CloudTrail, Log VPC Aliran, log DNS kueri, dan Amazon. EKS Anda tidak perlu mengelola kebijakan bucket Amazon S3 atau mengubah cara Anda mengumpulkan dan menyimpan log. Sebaiknya tetap simpan dan mempertahankan log tersebut untuk tujuan penyelidikan dan kepatuhan.
  - AWS Security Hubmenyediakan satu tempat yang mengumpulkan, mengatur, dan memprioritaskan peringatan keamanan atau temuan Anda dari beberapa AWS layanan dan produk pihak ketiga opsional untuk memberi Anda pandangan komprehensif tentang peringatan keamanan dan status kepatuhan.

Anda juga dapat menggunakan mesin pembuat peringatan kustom untuk peringatan keamanan yang tidak dicakup oleh layanan-layanan ini atau untuk peringatan tertentu yang relevan dengan lingkungan Anda. Untuk informasi tentang membuat peringatan dan deteksi ini, lihat Deteksi di Panduan Respons Insiden AWS Keamanan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC04-BP02 Tangkap log, temuan, dan metrik di lokasi standar
- SEC07-BP04 Tentukan manajemen siklus hidup data yang dapat diskalakan
- SEC10-BP06 Alat pra-penerapan

Dokumen terkait:

- AWS Panduan Respons Insiden Keamanan
- Memulai dengan Danau Keamanan Amazon
- Memulai: Amazon CloudWatch Logs
- Solusi Partner Keamanan: Pencatatan Log dan Pemantauan

Video terkait:

- AWS Re:invent 2022 - Memperkenalkan Danau Keamanan Amazon

Contoh terkait:

- Pembantu Log Enabler untuk AWS
- AWS Security Hub Temuan Ekspor Sejarah

Alat terkait:

- Snowflake untuk Keamanan Siber

SEC04-BP02 Tangkap log, temuan, dan metrik di lokasi standar

Tim keamanan mengandalkan log dan temuan untuk melakukan analisis terhadap peristiwa yang mungkin mengindikasikan aktivitas yang tidak sah atau perubahan yang tidak disengaja. Untuk menyederhanakan analisis ini, lakukan perekaman log dan temuan keamanan di lokasi yang terstandardisasi.  Hal ini membuat titik data yang menjadi perhatian tersedia untuk korelasi dan dapat menyederhanakan integrasi alat.

Hasil yang diinginkan: Anda memiliki pendekatan standar untuk mengumpulkan, menganalisis, dan memvisualisasikan data log, temuan, dan metrik. Tim keamanan dapat secara efisien mengorelasikan, menganalisis, dan memvisualisasikan data keamanan di seluruh sistem yang berbeda untuk menemukan kemungkinan adanya potensi peristiwa keamanan dan mengidentifikasi anomali. Informasi keamanan dan manajemen peristiwa (SIEM) sistem atau mekanisme lainnya terintegrasi untuk menanyakan dan menganalisis data log untuk respons, pelacakan, dan eskalasi peristiwa keamanan yang tepat waktu.

Anti-pola umum:

- Tim secara mandiri memiliki dan mengelola pencatatan log dan pengumpulan metrik yang tidak konsisten dengan strategi pencatatan log organisasi.
- Tim tidak memiliki kontrol akses yang memadai untuk membatasi visibilitas dan perubahan terhadap data yang dikumpulkan.
- Tim tidak mengatur log, temuan, dan metrik keamanan mereka sebagai bagian dari kebijakan klasifikasi data.
- Tim mengabaikan persyaratan kedaulatan dan pelokalan data saat melakukan konfigurasi terhadap pengumpulan data.

Manfaat menerapkan praktik terbaik ini: Solusi pencatatan log standar untuk mengumpulkan dan menanyakan data dan peristiwa log akan meningkatkan wawasan yang diperoleh dari informasi yang dikandungnya. Konfigurasi siklus hidup otomatis untuk data log yang dikumpulkan dapat mengurangi biaya yang ditimbulkan oleh penyimpanan log. Anda dapat membuat kontrol akses terperinci untuk informasi log yang dikumpulkan sesuai dengan sensitivitas data dan pola akses yang dibutuhkan oleh tim-tim Anda. Anda dapat mengintegrasikan peralatan untuk mengorelasikan, memvisualisasikan, dan memperoleh wawasan dari data.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Pertumbuhan AWS penggunaan dalam suatu organisasi menghasilkan semakin banyak beban kerja dan lingkungan yang didistribusikan. Karena masing-masing beban kerja dan lingkungan ini menghasilkan data tentang aktivitas di dalamnya, pencatatan dan penyimpanan data ini secara lokal akan menimbulkan kesulitan untuk operasi keamanan. Tim keamanan menggunakan alat seperti informasi keamanan dan sistem manajemen acara (SIEM) untuk mengumpulkan data dari sumber terdistribusi dan menjalani alur kerja korelasi, analisis, dan respons. Ini memerlukan pengelolaan serangkaian izin yang kompleks untuk mengakses berbagai sumber data dan overhead tambahan dalam mengoperasikan proses ekstraksi, transformasi, dan pemuatan ()ETL.

Untuk mengatasi tantangan ini, pertimbangkan untuk menggabungkan semua sumber data log keamanan yang relevan ke dalam akun Arsip Log seperti yang dijelaskan dalam Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun. Hal ini mencakup semua data terkait keamanan dari beban kerja dan log yang dihasilkan layanan AWS , seperti, AWS CloudTrail, AWS WAF, Penyeimbangan Beban Elastis, dan Amazon Route 53. Ada beberapa manfaat untuk menangkap data ini di lokasi standar secara terpisah Akun AWS dengan izin lintas akun yang tepat. Praktik ini membantu mencegah log dimanipulasi dalam beban kerja dan lingkungan yang mengalami kebocoran keamanan, menyediakan satu titik integrasi untuk alat tambahan, dan menawarkan model yang lebih sederhana untuk mengonfigurasi retensi data dan siklus hidup data.  Evaluasi dampak kedaulatan data, cakupan kepatuhan, dan peraturan lainnya untuk menentukan apakah beberapa lokasi penyimpanan data dan periode retensi data keamanan diperlukan.

Untuk memudahkan perekaman dan standardisasi log dan temuan, lakukan evaluasi terhadap Danau Keamanan Amazon di akun Arsip Log Anda. Anda dapat mengonfigurasi Security Lake untuk secara otomatis menyerap data dari sumber umum seperti CloudTrail, Route 53, Amazon EKS, dan VPCFlow Logs. Anda juga dapat mengonfigurasi AWS Security Hub sebagai sumber data ke Security Lake, memungkinkan Anda mengkorelasikan temuan dari AWS layanan lain, seperti Amazon GuardDuty dan Amazon Inspector, dengan data log Anda.  Anda juga dapat menggunakan integrasi sumber data pihak ketiga, atau mengonfigurasi sumber data kustom. Semua integrasi menstandarisasi data Anda ke dalam format Open Cybersecurity Schema Framework (OCSF), dan disimpan dalam bucket Amazon S3 sebagai file Parket, sehingga tidak perlu diproses. ETL

Menyimpan data keamanan di lokasi standar memberikan kemampuan analitik tingkat lanjut.AWS merekomendasikan Anda menggunakan alat untuk analitik keamanan yang beroperasi di AWS lingkungan ke akun Perkakas Keamanan yang terpisah dari akun Arsip Log Anda.  Pendekatan ini akan memungkinkan Anda untuk mengimplementasikan kontrol secara mendalam guna melindungi integritas dan ketersediaan log serta proses manajemen log, terpisah dari alat yang

mengaksesnya. Pertimbangkan untuk menggunakan layanan-layanan, seperti Amazon Athena, untuk menjalankan kueri sesuai permintaan yang menghubungkan beberapa sumber data. Anda juga dapat mengintegrasikan alat visualisasi, seperti Amazon QuickSight. Solusi yang didukung AI makin banyak tersedia dan dapat melakukan berbagai fungsi, misalnya menerjemahkan temuan ke dalam ringkasan yang dapat dibaca manusia dan interaksi bahasa yang alami. Solusi ini sering kali lebih mudah diintegrasikan dengan memiliki lokasi penyimpanan data terstandardisasi untuk melakukan kueri.

Langkah-langkah implementasi

1. Buat akun Arsip Log dan Peralatan Keamanan
    a. Menggunakan AWS Organizations, buat akun Log Archive dan Security Tooling di bawah unit organisasi keamanan. Jika Anda menggunakan AWS Control Tower untuk mengelola organisasi Anda, akun Log Archive dan Security Tooling dibuat untuk Anda secara otomatis. Konfigurasikan peran dan izin untuk mengakses dan mengelola akun ini sesuai kebutuhan.
2. Konfigurasikan lokasi data keamanan terstandardisasi Anda
    a. Tentukan strategi Anda untuk membuat lokasi data keamanan terstandardisasi. Anda dapat mencapai ini melalui opsi seperti pendekatan arsitektur danau data umum, produk data pihak ketiga, atau Amazon Security Lake.AWS merekomendasikan agar Anda mengambil data keamanan dari Wilayah AWS yang ikut serta untuk akun Anda, bahkan ketika tidak digunakan secara aktif.
3. Konfigurasikan publikasi sumber data ke lokasi terstandardisasi Anda
    a. Identifikasi sumber untuk data keamanan Anda dan konfigurasikan untuk dipublikasikan ke lokasi standar Anda. Evaluasi opsi untuk mengekspor data secara otomatis dalam format yang diinginkan sebagai lawan dari opsi di mana ETL proses perlu dikembangkan. Dengan Amazon Security Lake, Anda dapat mengumpulkan data dari AWS sumber yang didukung dan sistem pihak ketiga yang terintegrasi.
4. Konfigurasikan alat untuk mengakses lokasi terstandardisasi Anda
    a. Konfigurasikan alat seperti Amazon Athena QuickSight, Amazon, atau solusi pihak ketiga untuk memiliki akses yang diperlukan ke lokasi standar Anda. Konfigurasikan alat-alat ini agar dapat beroperasi di luar akun Alat Keamanan dengan akses baca lintas akun ke akun Arsip Log, jika diperlukan. Buat pelanggan di Danau Keamanan Amazon untuk menyediakan akses alat ini ke data Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP01 Memisahkan beban kerja menggunakan akun
- SEC07-BP04 Tentukan manajemen siklus hidup data
- SEC08-BP04 Menegakkan kontrol akses
- OPS08-BP02 Menganalisis log beban kerja

Dokumen terkait:

- AWS Whitepaper: Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun
- AWS Panduan Preskriptif: Arsitektur Referensi AWS Keamanan ()AWS SRA
- Panduan Preskriptif AWS : Panduan pencatatan log dan pemantauan untuk pemilik aplikasi

Contoh terkait:

- Menggabungkan, mencari, dan memvisualisasikan data log dari sumber terdistribusi dengan Amazon Athena dan Amazon QuickSight
- Cara memvisualisasikan temuan Amazon Security Lake dengan Amazon QuickSight
- Hasilkan wawasan bertenaga AI untuk Amazon Security Lake menggunakan Amazon SageMaker Studio dan Amazon Bedrock
- Identifikasi anomali keamanan siber dalam data Amazon Security Lake Anda menggunakan Amazon SageMaker
- Menelan, mengubah, dan mengirimkan acara yang diterbitkan oleh Amazon Security Lake ke Amazon Service OpenSearch
- Cara menggunakan AWS Security Hub dan OpenSearch Layanan Amazon untuk SIEM

Alat terkait:

- Danau Keamanan Amazon
- Integrasi Mitra Danau Keamanan Amazon
- Kerangka Kerja Skema Keamanan Siber Terbuka () OCSF
- Amazon Athena

- Amazon QuickSight
- Amazon Bedrock

SEC04-BP03 Menghubungkan dan memperkaya peringatan keamanan

Aktivitas tak terduga dapat menghasilkan beberapa peringatan keamanan yang dibuat oleh sumber yang berbeda, yang membutuhkan korelasi dan pengayaan lebih lanjut untuk memahami konteksnya secara lengkap. Menerapkan korelasi otomatis dan pengayaan peringatan keamanan untuk membantu mencapai identifikasi dan respons insiden yang lebih akurat.

Hasil yang diinginkan: Karena aktivitas menghasilkan peringatan yang berbeda dalam beban kerja dan lingkungan Anda, maka mekanisme otomatis menghubungkan data dan memperkaya data tersebut dengan informasi tambahan. Langkah-langkah sebelum pemrosesan (pre-processing) ini menyajikan pemahaman yang lebih mendetail tentang peristiwa, sehingga akan membantu penyelidik Anda dalam menentukan tingkat kekritisan peristiwa tersebut dan apakah peristiwa tersebut merupakan insiden yang memerlukan respons formal. Proses ini akan mengurangi beban pada tim pemantauan dan investigasi Anda.

Anti-pola umum:

- Grup orang yang berbeda-beda menyelidiki temuan dan peringatan yang dihasilkan oleh berbagai sistem, kecuali jika ditentukan lain berdasarkan persyaratan pemisahan tugas.
- Organisasi Anda menyalurkan semua data temuan dan peringatan keamanan ke lokasi terstandardisasi, tetapi mengharuskan penyelidik melakukan korelasi dan pengayaan data secara manual.
- Anda hanya mengandalkan intelijen sistem deteksi ancaman untuk melaporkan temuan dan menetapkan tingkat kekritisan.

Manfaat menerapkan praktik terbaik ini: Korelasi otomatis dan pengayaan peringatan akan membantu Anda dalam mengurangi beban kognitif secara keseluruhan dan persiapan data manual yang diperlukan peneliti Anda. Praktik ini dapat mengurangi waktu yang dibutuhkan untuk menentukan apakah peristiwa tersebut merepresentasikan sebuah insiden dan memulai respons formal. Konteks tambahan juga akan membantu Anda untuk menilai secara akurat tingkat keparahan yang sebenarnya dari suatu peristiwa, karena hal itu bisa jadi lebih tinggi atau lebih rendah dari yang diindikasikan oleh satu peringatan mana pun.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Peringatan keamanan dapat berasal dari berbagai sumber di dalamnya AWS, termasuk:

- Layanan seperti Amazon GuardDuty, AmazonMacie AWS Security Hub, AmazonInspector,, dan Network AWS ConfigAccess AWS Identity and Access Management Access AnalyzerAnalyzer
- Peringatan dari analisis otomatis AWS layanan, infrastruktur, dan log aplikasi, seperti dari Security Analytics for Amazon OpenSearch Service.
- Alarm sebagai respons terhadap perubahan aktivitas penagihan Anda dari sumber seperti Amazon EventBridge, CloudWatchAmazon, atau. AWS Budgets
- Sumber pihak ketiga seperti umpan intelijen ancaman dan Solusi Mitra Keamanan dari AWS Partner Network
- Kontak oleh AWS Trust & Safety atau sumber lain, seperti pelanggan atau karyawan internal.

Dalam bentuknya yang paling mendasar, peringatan berisi informasi tentang siapa (pengguna utama atau identitas) yang melakukan apa (tindakan yang diambil) terhadap apa (sumber daya yang terdampak tindakan). Untuk masing-masing sumber ini, identifikasi apakah Anda dapat membuat pemetaan di seluruh pengidentifikasi untuk identitas, tindakan, dan sumber daya ini sebagai landasan untuk melakukan korelasi. Ini dapat berupa pengintegrasian sumber peringatan dengan alat informasi keamanan dan manajemen peristiwa (SIEM) untuk melakukan korelasi otomatis untuk Anda, membangun jaringan pipa dan pemrosesan data Anda sendiri, atau kombinasi keduanya.

Contoh layanan yang dapat melakukan korelasi untuk Anda adalah Amazon Detective. Detective melakukan konsumsi terus menerus peringatan dari berbagai sumber pihak ketiga AWS dan menggunakan berbagai bentuk kecerdasan untuk menyusun grafik visual hubungan mereka untuk membantu penyelidikan.

Meskipun tingkat kekritisan awal sebuah peringatan dapat membantu dalam menentukan prioritisas, konteks yang mendasari terjadinya peringatan tersebut akan menentukan tingkat kekritisan yang sebenarnya. Sebagai contoh, Amazon GuardDuty dapat mengingatkan bahwa EC2 instans Amazon dalam beban kerja Anda sedang menanyakan nama domain yang tidak terduga. GuardDuty mungkin menetapkan kekritisan rendah untuk peringatan ini sendiri. Namun, korelasi otomatis dengan aktivitas lain di sekitar waktu peringatan mungkin mengungkap bahwa beberapa ratus EC2 contoh digunakan oleh identitas yang sama, yang meningkatkan biaya operasi secara keseluruhan. Dalam acara ini, GuardDuty mungkin mempublikasikan konteks peristiwa yang berkorelasi ini sebagai peringatan keamanan baru dan menyesuaikan kekritisan ke tinggi, yang akan mempercepat tindakan lebih lanjut.

Langkah-langkah implementasi

1. Identifikasi sumber untuk informasi peringatan keamanan. Pahami cara peringatan dari sistem ini merepresentasikan identitas, tindakan, dan sumber daya untuk menentukan di mana korelasi mungkin dilakukan.
2. Tetapkan sebuah mekanisme untuk mencatat peringatan dari sumber yang berbeda-beda. Pertimbangkan layanan seperti Security Hub, EventBridge, dan CloudWatch untuk tujuan ini.
3. Identifikasi sumber untuk korelasi dan pengayaan data. Contoh sumber meliputi CloudTrail, VPC Flow Logs, Amazon Security Lake, dan log infrastruktur dan aplikasi.
4. Integrasikan peringatan Anda dengan sumber korelasi dan pengayaan data untuk membuat konteks peristiwa keamanan yang lebih mendetail dan menetapkan tingkat kekritisannya.
   a. Amazon Detective, SIEM tooling, atau solusi pihak ketiga lainnya dapat melakukan tingkat konsumsi, korelasi, dan pengayaan tertentu secara otomatis.
   b. Anda juga dapat menggunakan AWS layanan untuk membangun layanan Anda sendiri. Misalnya, Anda dapat memanggil AWS Lambda fungsi untuk menjalankan kueri Amazon Athena terhadap AWS CloudTrail atau Amazon Security Lake, dan mempublikasikan hasilnya. EventBridge

Sumber daya

Praktik-praktik terbaik terkait:

- SEC10-BP03 Siapkan kemampuan forensik
- OPS08-BP04 Buat lansiran yang dapat ditindaklanjuti
- REL06-BP03 Kirim pemberitahuan (Pemrosesan waktu nyata dan mengkhawatirkan)

Dokumen terkait:

- Panduan Respons Insiden Keamanan AWS

Contoh terkait:

- Cara memperkaya AWS Security Hub temuan dengan metadata akun
- Cara menggunakan AWS Security Hub dan OpenSearch Layanan Amazon untuk SIEM

Alat terkait:

- Amazon Detective
- Amazon EventBridge
- AWS Lambda
- Amazon Athena

SEC04-BP04 Memulai remediasi untuk sumber daya yang tidak sesuai

Kontrol deteksi Anda mungkin memberikan peringatan tentang sumber daya yang tidak mematuhi persyaratan-persyaratan konfigurasi Anda. Anda dapat mulai melakukan remediasi yang ditentukan secara programatis, baik secara manual maupun otomatis, untuk melakukan remediasi terhadap berbagai sumber daya ini dan membantu meminimalkan dampak-dampak yang mungkin ditimbulkannya. Ketika Anda menentukan remediasi secara programatis, Anda dapat mengambil tindakan yang cepat dan konsisten.

Meskipun otomatisasi dapat meningkatkan operasi keamanan, Anda harus mengimplementasikan dan mengelola otomatisasi dengan hati-hati.  Terapkan mekanisme pengawasan dan kontrol yang tepat untuk memastikan bahwa respons otomatis berjalan efektif, akurat, dan selaras dengan kebijakan organisasi dan tingkat risiko yang dapat diterima.

Hasil yang diinginkan: Anda menentukan standar-standar konfigurasi sumber daya bersama dengan langkah-langkah untuk memulihkan ketika sumber daya terdeteksi tidak sesuai. Jika memungkinkan, Anda sudah harus menentukan remediasi secara programatis sehingga remediasi ini dapat dimulai baik secara manual maupun dengan otomatisasi. Sistem deteksi tersedia untuk mengidentifikasi sumber daya yang tidak mematuhi persyaratan dan memublikasikan peringatan ke alat-alat tersentralisasi yang dipantau oleh personel keamanan Anda. Dengan alat-alat ini, remediasi programatis Anda dapat dijalankan, baik secara manual maupun otomatis. Remediasi otomatis memiliki mekanisme pengawasan dan kontrol yang tepat untuk mengatur penggunaannya.

Anti-pola umum:

- Anda mengimplementasikan otomatisasi, tetapi gagal menguji dan memvalidasi tindakan-tindakan remediasi secara menyeluruh. Hal ini dapat menimbulkan konsekuensi-konsekuensi yang tidak diinginkan, seperti mengganggu operasi bisnis yang sah atau menyebabkan ketidakstabilan terhadap sistem.
- Anda mengoptimalkan waktu dan prosedur respons melalui otomatisasi, tetapi tanpa pemantauan dan mekanisme yang tepat yang memungkinkan intervensi dan penilaian manusia saat diperlukan.

- Anda hanya mengandalkan remediasi, bukannya menggunakan remediasi sebagai salah satu bagian dari program respons dan pemulihan insiden yang lebih luas.

Manfaat menerapkan praktik terbaik ini: Remediasi otomatis dapat merespons kesalahan konfigurasi lebih cepat daripada proses manual, yang akan membantu Anda meminimalkan potensi dampak bisnis dan mengurangi jendela peluang terjadinya penggunaan yang tidak diinginkan. Ketika Anda menentukan remediasi secara terprogram, remediasi akan diterapkan secara konsisten, dan hal ini akan mengurangi risiko kesalahan manusia. Otomatisasi juga dapat menangani volume peringatan yang lebih besar yang muncul secara bersamaan, yang merupakan hal yang sangat penting di lingkungan yang beroperasi dalam skala besar.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Seperti dijelaskan dalam SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan pengendalian, layanan seperti AWS Config dapat membantu Anda memantau konfigurasi sumber daya di akun Anda untuk kepatuhan terhadap kebutuhan Anda.  Ketika sumber daya yang tidak sesuai terdeteksi, sebaiknya Anda mengonfigurasi pengiriman peringatan ke solusi manajemen postur keamanan cloud (CSPM), seperti AWS Security Hub, untuk membantu perbaikan. Solusi ini akan menyediakan sebuah tempat terpusat bagi penyelidik keamanan Anda untuk memantau masalah dan mengambil tindakan korektif.

Meskipun beberapa situasi sumber daya yang tidak mematuhi persyaratan bersifat unik dan memerlukan penilaian manusia untuk diremediasi, namun situasi lainnya memiliki respons standar yang dapat Anda tentukan secara programatis. Misalnya, respons standar terhadap grup VPC keamanan yang salah konfigurasi dapat menghapus aturan yang tidak diizinkan dan memberi tahu pemiliknya. Respons dapat didefinisikan dalam fungsi AWS Lambda, dokumen AWS Systems Manager Automation, atau melalui lingkungan kode lain yang Anda inginkan. Pastikan lingkungan dapat mengautentikasi untuk AWS menggunakan IAM peran dengan jumlah izin paling sedikit yang diperlukan untuk mengambil tindakan korektif.

Setelah Anda menentukan remediasi yang diinginkan, Anda kemudian dapat menentukan cara pilihan Anda untuk memulainya. AWS Config dapat memulai remediasi untuk Anda. Jika Anda menggunakan Security Hub, Anda dapat melakukannya melalui tindakan khusus, yang menerbitkan informasi temuan ke Amazon EventBridge. EventBridge Aturan kemudian dapat memulai remediasi Anda. Anda dapat mengonfigurasi tindakan kustom di Security Hub untuk dijalankan secara otomatis atau manual.

Untuk remediasi terprogram, sebaiknya Anda memiliki log dan audit komprehensif untuk tindakan yang diambil, serta hasilnya. Lakukan peninjauan dan analisis terhadap log ini untuk menilai efektivitas proses yang terjadi secara otomatis, dan mengidentifikasi area perbaikan yang mungkin dilakukan. Tangkap log di CloudWatch Log Amazon dan hasil remediasi sebagai menemukan catatan di Security Hub.

Sebagai titik awal, pertimbangkan Respons Keamanan Otomatis AWS, yang memiliki remediasi pra-bangun untuk menyelesaikan kesalahan konfigurasi keamanan umum.

Langkah-langkah implementasi

1. Analisis dan prioritaskan peringatan.
   a. Mengkonsolidasikan peringatan keamanan dari berbagai AWS layanan ke Security Hub untuk visibilitas, prioritas, dan remediasi terpusat.
2. Kembangkan remediasi.
   a. Gunakan layanan seperti Systems Manager dan AWS Lambda untuk menjalankan remediasi terprogram.
3. Konfigurasikan cara remediasi dimulai.
   a. Menggunakan Systems Manager, tentukan tindakan kustom yang mempublikasikan temuan ke EventBridge. Konfigurasikan tindakan-tindakan ini untuk dimulai secara manual atau otomatis.
   b. Anda juga dapat menggunakan Amazon Simple Notification Service (SNS) untuk mengirim pemberitahuan dan peringatan kepada pemangku kepentingan terkait (seperti tim keamanan atau tim respons insiden) untuk intervensi atau eskalasi manual, jika diperlukan.
4. Tinjau dan analisis log remediasi untuk menentukan efektivitas dan peningkatan.
   a. Kirim output log ke CloudWatch Log. Catat hasil sebagai catatan temuan di Security Hub.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC06-BP03 Mengurangi manajemen manual dan akses interaktif

Dokumen terkait:

- Panduan Respons Insiden Keamanan AWS - Deteksi

Contoh terkait:

- Respon Keamanan Otomatis pada AWS
- Pantau pasangan kunci EC2 instance menggunakan AWS Config
- Buat aturan kustom AWS Config dengan menggunakan kebijakan AWS CloudFormation Guard
- Secara otomatis memulihkan instans dan cluster RDS Amazon DB yang tidak terenkripsi

Alat terkait:

- AWS Otomatisasi Systems Manager
- Respon Keamanan Otomatis pada AWS

## Perlindungan infrastruktur

Pertanyaan

- SEC5. Bagaimana cara melindungi sumber daya jaringan Anda?
- SEC6. Bagaimana cara melindungi sumber daya komputasi Anda?

### SEC5. Bagaimana cara melindungi sumber daya jaringan Anda?

Setiap beban kerja yang memiliki suatu bentuk konektivitas jaringan, baik internet atau jaringan privat, memerlukan beberapa lapisan pertahanan untuk membantu melindungi dari ancaman eksternal dan internal berbasis jaringan.

Praktik terbaik

- SEC05-BP01 Buat lapisan jaringan
- SEC05-BP02 Kontrol arus lalu lintas dalam lapisan jaringan Anda
- SEC05-BP03 Menerapkan perlindungan berbasis inspeksi
- SEC05-BP04 Mengotomatiskan perlindungan jaringan

#### SEC05-BP01 Buat lapisan jaringan

Segmentasikan topologi jaringan Anda ke dalam lapisan yang berbeda-beda berdasarkan pengelompokan logis komponen beban kerja Anda sesuai dengan sensitivitas data dan persyaratan

aksesnya. Bedakan antara komponen yang memerlukan akses masuk dari internet, seperti titik akhir web publik, dan komponen yang hanya membutuhkan akses internal, seperti basis data.

Hasil yang diinginkan: Lapisan jaringan Anda adalah bagian dari defense-in-depth pendekatan integral terhadap keamanan yang melengkapi otentikasi identitas dan strategi otorisasi beban kerja Anda. Lapisan-lapisan diterapkan berdasarkan sensitivitas data dan persyaratan akses, dengan arus lalu lintas dan mekanisme kontrol yang sesuai.

Anti-pola umum:

- Anda membuat semua sumber daya dalam satu VPC atau subnet.
- Anda menyusun konsep lapisan-lapisan jaringan tanpa mempertimbangkan persyaratan sensitivitas data, perilaku komponen, atau fungsionalitas.
- Anda menggunakan VPCs dan subnet sebagai default untuk semua pertimbangan lapisan jaringan, dan Anda tidak mempertimbangkan bagaimana layanan AWS terkelola memengaruhi topologi Anda.

Manfaat menerapkan praktik terbaik ini: Membangun lapisan jaringan adalah langkah pertama dalam membatasi jalur-jalur yang tidak perlu melalui jaringan, terutama jalur-jalur yang mengarah pada sistem dan data kritis. Hal ini akan mempersulit pelaku yang tidak sah untuk mendapatkan akses ke jaringan Anda dan menavigasi ke sumber daya tambahan di dalamnya. Lapisan-lapisan jaringan terpisah (discrete) bermanfaat dalam mengurangi cakupan analisis untuk sistem inspeksi, seperti untuk deteksi intrusi atau pencegahan malware. Hal ini dapat mengurangi potensi positif palsu dan overhead pemrosesan yang tidak perlu.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Saat merancang arsitektur beban kerja, biasanya berbagai komponen dipisahkan menjadi lapisan yang berbeda-beda berdasarkan tanggung jawabnya. Misalnya, sebuah aplikasi web dapat memiliki lapisan presentasi, lapisan aplikasi, dan lapisan data. Anda dapat mengambil pendekatan yang serupa saat merancang desain topologi jaringan Anda. Kontrol jaringan yang mendasarinya dapat membantu Anda memberlakukan persyaratan akses data pada beban kerja Anda. Misalnya, dalam arsitektur aplikasi web tiga tingkat, Anda dapat menyimpan file lapisan presentasi statis Anda di Amazon S3 dan menyajikannya dari jaringan pengiriman konten CDN (), seperti Amazon. CloudFront Lapisan aplikasi dapat memiliki titik akhir publik yang berfungsi Application Load Balancer ALB () di subnet publik VPCAmazon (mirip dengan zona demiliterisasi, DMZ atau), dengan layanan back-

end yang digunakan ke subnet pribadi. Lapisan data tersebut, yang merupakan sumber daya hosting seperti basis data dan sistem file bersama, dapat berada di subnet privat yang berbeda dari sumber daya lapisan aplikasi Anda. Pada setiap batas lapisan ini (CDN, subnet publik, subnet pribadi), Anda dapat menerapkan kontrol yang memungkinkan hanya lalu lintas resmi untuk melintasi batas-batas tersebut.

Serupa dengan pemodelan lapisan jaringan berdasarkan tujuan fungsional komponen beban kerja Anda, pertimbangkan juga sensitivitas data yang diproses. Menggunakan contoh aplikasi web tersebut, meskipun semua layanan beban kerja Anda mungkin berada dalam lapisan aplikasi, layanan yang berbeda dapat memproses data dengan tingkat sensitivitas yang berbeda. Dalam hal ini, membagi lapisan aplikasi menggunakan beberapa subnet pribadi, berbeda VPCs dalam hal yang sama Akun AWS, atau bahkan berbeda VPCs berbeda Akun AWS untuk setiap tingkat sensitivitas data mungkin sesuai dengan persyaratan kontrol Anda.

Pertimbangan lebih lanjut yang harus dilakukan untuk lapisan jaringan adalah konsistensi perilaku dari komponen-komponen yang dimiliki oleh beban kerja Anda. Melanjutkan dengan contoh ini, di lapisan aplikasi, Anda mungkin memiliki layanan yang menerima input dari pengguna akhir atau integrasi sistem eksternal yang secara inheren lebih berisiko daripada input ke layanan lain. Contohnya termasuk unggahan file, skrip kode yang dijalankan, pemindaian email, dan sebagainya. Dengan menempatkan layanan-layanan tersebut di lapisan jaringannya sendiri, akan terbentuk batas isolasi yang lebih kuat di sekitarnya, dan perilaku uniknya dapat dicegah agar tidak menghasilkan peringatan positif palsu dalam sistem inspeksi.

Sebagai bagian dari desain Anda, pertimbangkan bagaimana menggunakan layanan AWS terkelola memengaruhi topologi jaringan Anda. Jelajahi bagaimana layanan seperti Amazon VPC Lattice dapat membantu mempermudah interoperabilitas komponen beban kerja Anda di seluruh lapisan jaringan. Saat menggunakan AWS Lambda, terapkan di VPC subnet Anda kecuali ada alasan khusus untuk tidak melakukannya. Tentukan di mana VPC titik akhir dan AWS PrivateLinkdapat menyederhanakan mematuhi kebijakan keamanan yang membatasi akses ke gateway internet.

Langkah-langkah implementasi

1. Tinjau arsitektur beban kerja Anda. Kelompokkan komponen dan layanan secara logis berdasarkan fungsi yang dijalankan, sensitivitas data yang diproses, dan perilakunya.
2. Untuk komponen-komponen yang merespons permintaan dari internet, pertimbangkan untuk menggunakan penyeimbang beban atau perantara lainnya guna menyediakan titik akhir publik. Jelajahi pengalihan kontrol keamanan dengan menggunakan layanan terkelola, seperti Amazon

API Gateway CloudFront, Elastic Load Balancing, AWS Amplifydan untuk menampung titik akhir publik.

3. Untuk komponen yang berjalan di lingkungan komputasi, seperti EC2 instans Amazon, AWS Fargatecontainer, atau fungsi Lambda, terapkan ini ke subnet pribadi berdasarkan grup Anda dari langkah pertama.
4. Untuk AWS layanan yang dikelola sepenuhnya, seperti Amazon DynamoDB, Amazon Kinesis, atau SQSAmazon, pertimbangkan untuk VPC menggunakan titik akhir sebagai default untuk akses melalui alamat IP pribadi.

Sumber daya

Praktik-praktik terbaik terkait:

- REL02 Rencanakan topologi jaringan Anda
- PERF04-BP01 Memahami bagaimana jaringan memengaruhi kinerja

Video terkait:

- AWS re: invent 2023 - yayasan jaringan AWS

Contoh terkait:

- VPCcontoh
- Akses aplikasi kontainer secara pribadi di Amazon ECS dengan menggunakan AWS Fargate, AWS PrivateLink, dan Network Load Balancer
- Sajikan konten statis dalam bucket Amazon S3 melalui VPC dengan menggunakan Amazon CloudFront

SEC05-BP02 Kontrol arus lalu lintas dalam lapisan jaringan Anda

Dalam lapisan-lapisan jaringan Anda, gunakan segmentasi lebih lanjut untuk membatasi lalu lintas hanya ke arus yang diperlukan untuk masing-masing beban kerja. Pertama, fokus pada pengendalian lalu lintas antara internet atau sistem eksternal lainnya ke beban kerja dan lingkungan Anda (lalu lintas utara-selatan). Setelah itu, lihat arus yang terjadi antara komponen dan sistem yang berbeda (lalu lintas timur-barat).

Hasil yang diinginkan: Anda hanya mengizinkan arus jaringan yang diperlukan untuk komponen beban kerja Anda untuk melakukan komunikasi satu sama lain dan dan klien mereka dan layanan lain yang mereka andalkan. Desain Anda mempertimbangkan hal-hal seperti lalu lintas masuk dan keluar publik dibandingkan dengan privat, klasifikasi data, peraturan regional, dan persyaratan protokol. Jika memungkinkan, Anda lebih menyukai point-to-point aliran melalui pengintip jaringan sebagai bagian dari prinsip desain hak istimewa yang paling tidak.

Anti-pola umum:

- Anda mengambil pendekatan berbasis perimeter untuk keamanan jaringan dan hanya mengontrol arus lalu lintas di batas lapisan-lapisan jaringan Anda.
- Anda menganggap semua lalu lintas yang ada dalam sebuah lapisan jaringan sudah diautentikasi dan diotorisasi.
- Anda dapat menerapkan kontrol untuk salah satu lalu lintas masuk atau lalu lintas keluar, tetapi tidak dapat menerapkan untuk keduanya.
- Anda hanya mengandalkan komponen-komponen beban kerja dan kontrol jaringan untuk melakukan autentikasi dan meberikan otorisasi terhadap lalu lintas.

Manfaat menerapkan praktik terbaik ini: Praktik ini akan membantu Anda dalam mengurangi risiko pergerakan yang tidak sah dalam jaringan Anda dan menambahkan lapisan otorisasi tambahan ke beban kerja Anda. Dengan melakukan kontrol terhadap arus lalu lintas, Anda dapat membatasi cakupan dampak insiden keamanan serta mempercepat deteksi dan respons.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Sementara lapisan jaringan membantu menetapkan batas-batas di sekitar komponen beban kerja Anda yang melayani fungsi, tingkat sensitivitas data, dan perilaku yang serupa, Anda dapat membuat tingkat kontrol lalu lintas yang jauh lebih halus dengan menggunakan teknik untuk mengelompokkan komponen lebih lanjut dalam lapisan-lapisan ini yang mengikuti prinsip hak istimewa paling sedikit. Di dalam AWS, lapisan jaringan terutama didefinisikan menggunakan subnet sesuai dengan rentang alamat IP dalam AmazonVPC. Lapisan juga dapat didefinisikan menggunakan yang berbedaVPCs, seperti untuk mengelompokkan lingkungan microservice berdasarkan domain bisnis. Saat menggunakan beberapaVPCs, mediasi perutean menggunakan file. AWS Transit Gateway Meskipun ini memberikan kontrol lalu lintas pada tingkat Layer 4 (alamat IP dan rentang port) menggunakan grup keamanan dan tabel rute, Anda dapat memperoleh kontrol lebih lanjut

menggunakan layanan tambahan, seperti AWS PrivateLink, Amazon Route 53 Resolver DNS Firewall AWS Network Firewall, dan. AWS WAF

Memahami dan menginventarisasi aliran data dan persyaratan komunikasi beban kerja Anda dalam hal pihak yang memulai koneksi, port, protokol, dan lapisan jaringan. Evaluasi protokol yang tersedia untuk membangun koneksi dan mentransmisikan data ke protokol tertentu yang memenuhi persyaratan perlindungan Anda (misalnya, HTTPS bukan). HTTP Terapkan persyaratan-persyaratan ini di batas-batas jaringan Anda dan dalam masing-masing lapisan. Setelah persyaratan-persyaratan ini diidentifikasi, tinjau opsi-opsi untuk mengizinkan lalu lintas yang diperlukan saja yang dapat mengalir di setiap titik koneksi. Titik awal yang baik adalah menggunakan grup keamanan di dalam AndaVPC, karena mereka dapat dilampirkan ke sumber daya yang menggunakan Antarmuka Jaringan Elastis (ENI), EC2 instans Amazon, ECS tugas Amazon, EKS pod Amazon, atau RDS database Amazon. Tidak seperti firewall Lapisan 4, sebuah grup keamanan dapat memiliki aturan yang mengizinkan lalu lintas dari grup keamanan lain berdasarkan pengidentifikasinya, sehingga hal itu akan meminimalkan pembaruan seiring berubahnya sumber daya dalam grup dari waktu ke waktu. Anda juga dapat memfilter lalu lintas dengan aturan masuk dan keluar dengan menggunakan grup keamanan.

Ketika lalu lintas bergerak di antaraVPCs, itu umum untuk menggunakan VPC peering untuk routing sederhana atau AWS Transit Gateway untuk routing yang kompleks. Dengan pendekatan ini, Anda memfasilitasi arus lalu lintas yang terjadi antara rentang alamat IP jaringan sumber dan tujuan. Namun, jika beban kerja Anda hanya memerlukan arus lalu lintas antara komponen tertentu yang berbedaVPCs, pertimbangkan untuk menggunakan point-to-point AWS PrivateLinkkoneksi. Untuk melakukan hal ini, identifikasi layanan mana yang harus bertindak sebagai produsen dan layanan mana yang harus bertindak sebagai konsumen. Terapkan penyeimbang beban yang kompatibel untuk produsen, nyalakan PrivateLink sesuai, dan kemudian terima permintaan koneksi oleh konsumen. Layanan produsen kemudian diberi alamat IP pribadi dari konsumen VPC yang dapat digunakan konsumen untuk membuat permintaan berikutnya. Pendekatan ini mengurangi kebutuhan untuk melakukan peering jaringan. Sertakan biaya untuk pemrosesan data dan penyeimbangan beban sebagai bagian dari evaluasi PrivateLink.

Sementara kelompok keamanan dan PrivateLink membantu mengontrol aliran antara komponen beban kerja Anda, pertimbangan utama lainnya adalah bagaimana mengontrol DNS domain mana yang diizinkan untuk diakses oleh sumber daya Anda (jika ada). Bergantung pada DHCP konfigurasi AndaVPCs, Anda dapat mempertimbangkan dua AWS layanan berbeda untuk tujuan ini. Sebagian besar pelanggan menggunakan DNS layanan Route 53 Resolver default (juga disebut DNS server Amazon atau AmazonProvidedDNS) yang tersedia VPCs di alamat +2 dari jangkauannyaCIDR. Dengan pendekatan ini, Anda dapat membuat aturan DNS Firewall dan mengaitkannya dengan

aturan Anda VPC yang menentukan tindakan apa yang harus diambil untuk daftar domain yang Anda berikan.

Jika Anda tidak menggunakan Route 53 Resolver, atau jika Anda ingin melengkapi Resolver dengan kemampuan inspeksi dan kontrol aliran yang lebih mendalam selain pemfilteran domain, pertimbangkan untuk melakukan deployment AWS Network Firewall. Layanan ini memeriksa masing-masing paket individual dengan menggunakan aturan stateless atau stateful untuk menentukan apakah akan menolak atau mengizinkan lalu lintas. Anda dapat mengambil pendekatan serupa untuk memfilter lalu lintas web masuk ke titik akhir publik Anda dengan menggunakan AWS WAF. Untuk panduan lebih lanjut tentang layanan ini, lihat SEC05-BP03 Menerapkan perlindungan berbasis inspeksi.

Langkah-langkah implementasi

1. Identifikasi aliran data yang diperlukan antara komponen-komponen beban kerja Anda.
2. Terapkan beberapa kontrol dengan defense-in-depth pendekatan untuk lalu lintas masuk dan keluar, termasuk penggunaan grup keamanan, dan tabel rute.
3. Gunakan firewall untuk menentukan kontrol halus atas lalu lintas jaringan masuk, keluar, dan melintasi Anda, seperti Firewall DNS Resolver Route 53VPCs,, dan. AWS Network Firewall AWS WAF Pertimbangkan untuk menggunakan AWS Firewall Manager untuk mengonfigurasi dan mengelola aturan firewall secara terpusat di seluruh organisasi Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- SEC09-BP02 Menerapkan enkripsi dalam perjalanan

Dokumen terkait:

- Praktik terbaik keamanan untuk Anda VPC
- Tips Optimalisasi Jaringan AWS
- Panduan untuk Keamanan Jaringan di AWS
- VPCAmankan lalu lintas jaringan keluar Anda di AWS Cloud

Alat terkait:

- AWS Firewall Manager

Video terkait:

- AWS Transit Gateway arsitektur referensi untuk banyak VPCs
- Akselerasi dan Perlindungan Aplikasi dengan Amazon CloudFront, AWS WAF, dan AWS Shield
- AWS re:Inforce 2023: Firewall dan tempat meletakkannya

Contoh terkait:

- Lab: CloudFront untuk Aplikasi Web

SEC05-BP03 Menerapkan perlindungan berbasis inspeksi

Siapkan titik inspeksi lalu lintas di antara lapisan jaringan Anda untuk memastikan data bergerak cocok dengan kategori dan pola yang diharapkan.  Lakukan analisis terhadap arus lalu lintas, metadata, dan pola untuk membantu Anda mengidentifikasi, mendeteksi, dan merespons peristiwa dengan lebih efektif.

Hasil yang diinginkan: Lalu lintas yang melintas antara lapisan jaringan Anda diperiksa dan diberi otorisasi.  Keputusan izinkan (allow) dan tolak (deny) didasarkan pada aturan-aturan eksplisit, intelijen ancaman, dan penyimpangan dari perilaku acuan dasar.  Perlindungan menjadi lebih ketat ketika lalu lintas makin mendekati data sensitif.

Anti-pola umum:

- Hanya mengandalkan aturan-aturan firewall berdasarkan port dan protokol. Tidak memanfaatkan sistem cerdas.
- Menulis aturan firewall berdasarkan pola ancaman spesifik saat ini yang masih dapat berubah.
- Hanya memeriksa lalu lintas yang lalu lintasnya bergerak dari subnet privat ke publik, atau dari subnet publik ke Internet.
- Tidak memiliki gambaran acuan dasar tentang lalu lintas jaringan Anda untuk dijadikan pembanding dengan anomali perilaku.

Manfaat menerapkan praktik terbaik ini: Sistem inspeksi akan memungkinkan Anda untuk membuat aturan cerdas, seperti mengizinkan atau menolak lalu lintas hanya ketika kondisi tertentu terjadi dalam data lalu lintas. Manfaatkan set aturan yang dikelola dari AWS dan mitra, berdasarkan intelijen

ancaman terbaru, karena lanskap ancaman berubah seiring waktu. Hal ini akan menurunkan overhead pemeliharaan aturan dan pencarian indikator penyusupan, sehingga dapat mengurangi potensi positif palsu.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Memiliki kontrol halus atas lalu lintas jaringan stateful dan stateless Anda menggunakan AWS Network Firewall, atau Firewall dan Intrusion Prevention Systems (IPS) lainnya yang AWS Marketplace dapat Anda gunakan di belakang Gateway Load Balancer (). GWLBAWS Network Firewall mendukung IPS spesifikasi open source yang kompatibel dengan Suricata untuk membantu melindungi beban kerja Anda.

Baik solusi AWS Network Firewall dan vendor yang menggunakan GWLB dukungan model penyebaran inspeksi inline yang berbeda. Misalnya, Anda dapat melakukan inspeksi per- VPC basis, memusatkan dalam inspeksiVPC, atau menerapkan dalam model hibrida di mana lalu lintas timur-barat mengalir melalui inspeksi VPC dan masuknya Internet diperiksa per-. VPC Pertimbangan lain adalah apakah solusi tersebut mendukung membuka bungkusan Transport Layer Security (TLS), memungkinkan inspeksi paket mendalam untuk arus lalu lintas yang dimulai di kedua arah. Untuk informasi selengkapnya dan detail mendalam tentang konfigurasi ini, lihat panduan Praktik Terbaik AWS Network Firewall.

Jika Anda menggunakan solusi yang melakukan out-of-band inspeksi, seperti analisis pcap data paket dari antarmuka jaringan yang beroperasi dalam mode promiscuous, Anda dapat mengonfigurasi mirroring lalu lintas. VPC Lalu lintas yang di-mirroring diperhitungkan terhadap bandwidth yang tersedia dari antarmuka Anda dan dikenai biaya transfer data yang sama dengan lalu lintas yang tidak di-mirroring. Anda dapat melihat apakah versi virtual dari peralatan ini tersedia di AWS Marketplace, yang dapat mendukung penyebaran sebaris di belakang file. GWLB

Untuk komponen yang bertransaksi melalui protokol HTTP berbasis, lindungi aplikasi Anda dari ancaman umum dengan firewall aplikasi web (). WAF AWS WAFadalah firewall aplikasi web yang memungkinkan Anda memantau dan memblokir HTTP permintaan yang sesuai dengan aturan yang dapat dikonfigurasi sebelum mengirim ke Amazon API Gateway, Amazon CloudFront, AWS AppSync atau Application Load Balancer. Pertimbangkan inspeksi paket mendalam ketika Anda mengevaluasi penerapan firewall aplikasi web Anda, karena beberapa mengharuskan Anda untuk menghentikan TLS sebelum inspeksi lalu lintas. Untuk memulai AWS WAF, Anda dapat menggunakan Peraturan yang Dikelola AWSkombinasi dengan milik Anda sendiri, atau menggunakan integrasi mitra yang ada.

Anda dapat mengelola AWS WAF, AWS Shield Advanced AWS Network Firewall, dan grup VPC keamanan Amazon secara terpusat di seluruh AWS Organisasi Anda. AWS Firewall Manager

Langkah-langkah implementasi

1. Tentukan apakah Anda dapat mencakup aturan inspeksi secara luas, seperti melalui inspeksiVPC, atau jika Anda memerlukan pendekatan per detail yang lebih terperinci. VPC
2. Untuk solusi-solusi inspeksi inline:
   a. Jika menggunakan AWS Network Firewall, buat aturan, kebijakan firewall, dan firewall itu sendiri. Setelah ini dikonfigurasi, Anda dapat merutekan lalu lintas ke titik akhir firewall untuk mengaktifkan inspeksi.
   b. Jika menggunakan alat pihak ketiga dengan Gateway Load Balancer (GWLB), gunakan dan konfigurasikan alat Anda di satu atau beberapa zona ketersediaan. Kemudian, buat, layanan titik akhirGWLB, titik akhir, dan konfigurasikan perutean untuk lalu lintas Anda.
3. Untuk solusi out-of-band inspeksi:
   1. Aktifkan Pencerminan VPC Lalu Lintas pada antarmuka tempat lalu lintas masuk dan keluar harus dicerminkan. Anda dapat menggunakan EventBridge aturan Amazon untuk menjalankan AWS Lambda fungsi guna mengaktifkan pencerminan lalu lintas pada antarmuka saat sumber daya baru dibuat. Arahkan sesi traffic mirroring ke Penyeimbang Beban Jaringan di depan alat Anda yang memproses lalu lintas.
4. Untuk solusi lalu lintas web masuk:
   a. Untuk mengkonfigurasi AWS WAF, mulailah dengan mengkonfigurasi daftar kontrol akses web (webACL). Web ACL adalah kumpulan aturan dengan tindakan default yang diproses secara serial (ALLOWatauDENY) yang menentukan bagaimana Anda WAF menangani lalu lintas. Anda dapat membuat aturan dan grup Anda sendiri atau menggunakan grup aturan AWS terkelola di web AndaACL.
   b. Setelah web Anda ACL dikonfigurasi, kaitkan web ACL dengan AWS sumber daya (seperti Application Load Balancer, API Gateway RESTAPI, atau CloudFront distribusi) untuk mulai melindungi lalu lintas web.

Sumber daya

Dokumen terkait:

- Apa itu Traffic Mirroring?
- Menerapkan inspeksi lalu lintas inline dengan menggunakan peralatan keamanan pihak ketiga

- AWS Network Firewall contoh arsitektur dengan routing
- Arsitektur inspeksi terpusat dengan AWS Gateway Load Balancer dan AWS Transit Gateway

Contoh terkait:

- Praktik terbaik untuk men-deploy Penyeimbang Beban Gateway
- TLSkonfigurasi inspeksi untuk lalu lintas jalan keluar terenkripsi dan AWS Network Firewall

Alat terkait:

- AWS Marketplace IDS/IPS

SEC05-BP04 Mengotomatiskan perlindungan jaringan

Otomatiskan penerapan perlindungan jaringan Anda menggunakan DevOps praktik, seperti infrastruktur sebagai kode (IAc) dan pipeline CI/CD. Praktik-praktik ini dapat membantu Anda melacak perubahan dalam perlindungan jaringan Anda melalui sistem kontrol versi, mengurangi waktu yang diperlukan untuk melakukan deployment perubahan, dan membantu Anda untuk mendeteksi apakah perlindungan jaringan Anda menyimpang dari konfigurasi yang Anda inginkan.

Hasil yang diinginkan: Anda menentukan perlindungan jaringan dengan templat dan memasukkannya ke dalam sistem kontrol versi. Pipeline otomatis dimulai ketika perubahan-perubahan baru dibuat yang mengorkestrasi pengujian dan deployment. Pemeriksaan kebijakan dan pengujian statis lainnya diterapkan untuk memvalidasi perubahan sebelum deployment. Anda melakukan deployment perubahan ke lingkungan pentahapan (staging) untuk memvalidasi bahwa berbagai kontrol beroperasi seperti yang diharapkan. Deployment ke lingkungan produksi Anda juga dilakukan secara otomatis setelah kontrol disetujui.

Anti-pola umum:

- Mengandalkan masing-masing tim beban kerja untuk menentukan tumpukan jaringan lengkap, perlindungan, dan otomatisasinya. Tidak memublikasikan aspek-aspek standar dari tumpukan dan perlindungan jaringan secara terpusat yang akan digunakan oleh tim beban kerja.
- Mengandalkan tim jaringan pusat untuk menentukan semua aspek jaringan, perlindungan, dan otomatisasi. Tidak mendelegasikan aspek-aspek spesifik beban kerja dari tumpukan dan perlindungan jaringan kepada tim beban kerja tersebut.

- Mencapai keseimbangan yang tepat antara sentralisasi dan delegasi antara tim jaringan dan tim beban kerja, tetapi tidak menerapkan standar pengujian dan deployment yang konsisten ke seluruh templat IaC dan pipeline CI/CD Anda. Tidak mencatat konfigurasi yang diperlukan dalam peralatan yang memeriksa kepatuhan templat Anda.

Manfaat menerapkan praktik terbaik ini: Menggunakan templat untuk menentukan perlindungan jaringan Anda akan memungkinkan Anda untuk melacak dan membandingkan perubahan dari waktu ke waktu dengan sistem kontrol versi. Penggunaan otomatisasi untuk menguji dan melakukan deployment perubahan akan menghasilkan standardisasi dan prediktabilitas, sehingga akan meningkatkan peluang keberhasilan deployment dan mengurangi konfigurasi manual yang berulang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sejumlah kontrol perlindungan jaringan yang dijelaskan dalam SEC05-BP02 Mengontrol arus lalu lintas dalam lapisan jaringan Anda dan SEC05-BP03 Menerapkan perlindungan berbasis inspeksi dilengkapi dengan sistem aturan terkelola yang dapat diperbarui secara otomatis berdasarkan intelijen ancaman terbaru. Contoh perlindungan endpoint web Anda termasuk aturan AWS WAF terkelola dan DDoSmitigasi lapisan aplikasi AWS Shield Advanced otomatis. Gunakan grup aturan terkelola AWS Network Firewall untuk tetap up to date dengan daftar domain yang memiliki reputasi rendah dan tanda tangan ancaman juga.

Selain aturan terkelola, kami sarankan Anda menggunakan DevOps praktik untuk mengotomatiskan penerapan sumber daya jaringan, perlindungan, dan aturan yang Anda tentukan. Anda dapat menangkap definisi ini di AWS CloudFormation atau alat infrastruktur sebagai kode (IaC) lain yang Anda pilih, meng-komit-nya ke sistem kontrol versi, dan men-deploynya menggunakan pipeline CI/CD. Gunakan pendekatan ini untuk mendapatkan manfaat tradisional DevOps untuk mengelola kontrol jaringan Anda, seperti rilis yang lebih dapat diprediksi, pengujian otomatis menggunakan alat seperti AWS CloudFormation Guard, dan mendeteksi penyimpangan antara lingkungan yang Anda gunakan dan konfigurasi yang Anda inginkan.

Berdasarkan keputusan yang Anda buat sebagai bagian dari SEC05-BP01 Buat lapisan jaringan, Anda mungkin memiliki pendekatan manajemen pusat untuk membuat VPCs yang didedikasikan untuk arus masuk, keluar, dan inspeksi. Seperti yang dijelaskan dalam Arsitektur Referensi AWS Keamanan (AWS SRA), Anda dapat menentukan ini VPCs dalam akun infrastruktur Jaringan khusus. Anda dapat menggunakan teknik serupa untuk menentukan secara terpusat yang VPCs digunakan oleh beban kerja Anda di akun lain, grup keamanannya, AWS Network Firewall

penerapan, aturan Resolver Route 53 dan konfigurasi DNS Firewall, dan sumber daya jaringan lainnya. Anda dapat berbagi sumber daya ini dengan akun Anda yang lain menggunakan AWS Resource Access Manager. Dengan pendekatan ini, Anda dapat menyederhanakan pengujian dan deployment otomatis atas kontrol jaringan Anda ke akun Jaringan, yang bisa Anda lakukan cukup dengan mengelola satu tujuan. Anda dapat melakukannya dalam model hibrida, yang akan memungkinkan Anda melakukan deployment dan membagikan kontrol tertentu secara terpusat serta mendelegasikan kontrol lainnya ke setiap tim beban kerja dan akun mereka masing-masing.

Langkah-langkah implementasi

1. Tetapkan kepemilikan untuk aspek-aspek jaringan dan perlindungan yang ditentukan secara terpusat, dan yang dapat dipelihara oleh tim beban kerja Anda.
2. Buat lingkungan untuk melakukan pengujian dan deployment perubahan-perubahan yang hendak dibuat pada jaringan Anda dan perlindungannya. Misalnya, gunakan akun Pengujian Jaringan dan akun Produksi Jaringan.
3. Tentukan cara Anda akan menyimpan dan memelihara templat Anda dalam sebuah sistem kontrol versi. Simpan templat pusat dalam sebuah repositori yang berbeda dari repositori beban kerja, sedangkan templat beban kerja dapat disimpan dalam repositori-repositori khusus untuk beban kerja tersebut.
4. Buat pipeline CI/CD untuk melakukan pengujian dan deployment templat. Tentukan pengujian untuk memeriksa adanya kesalahan konfigurasi dan apakah templat tersebut mematuhi standar perusahaan Anda, atau tidak.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP06 Mengotomatiskan penerapan kontrol keamanan standar

Dokumen terkait:

- Arsitektur Referensi Keamanan AWS - Akun jaringan

Contoh terkait:

- Arsitektur Referensi Pipeline Deployment AWS
- NetDevSecOpsuntuk memodernisasi penyebaran jaringan AWS

- Mengintegrasikan tes AWS CloudFormation keamanan dengan AWS Security Hub dan laporan AWS CodeBuild

Alat terkait:

- AWS CloudFormation
- AWS CloudFormation Guard
- cfn_nag

## SEC6. Bagaimana cara melindungi sumber daya komputasi Anda?

Sumber daya komputasi di beban kerja Anda memerlukan beberapa lapisan pertahanan untuk membantu melindungi dari ancaman eksternal dan internal. Sumber daya komputasi mencakup EC2 instance, wadah, AWS Lambda fungsi, layanan basis data, perangkat IoT, dan lainnya.

Praktik terbaik

- SEC06-BP01 Lakukan manajemen kerentanan
- SEC06-BP02 Komputasi ketentuan dari gambar yang diperkeras
- SEC06-BP03 Mengurangi manajemen manual dan akses interaktif
- SEC06-BP04 Validasi integritas perangkat lunak
- SEC06-BP05 Mengotomatiskan perlindungan komputasi

### SEC06-BP01 Lakukan manajemen kerentanan

Seringlah memindai dan mem-patch kelemahan pada kode, dependensi, dan infrastruktur Anda untuk membantu mencegah ancaman baru.

Hasil yang diinginkan: Membuat dan memelihara sebuah program manajemen kerentanan. Memindai dan menambal resource secara teratur seperti EC2 instans Amazon, container Amazon Elastic Container Service (AmazonECS), dan beban kerja Amazon Elastic Kubernetes Service (Amazon). EKS Konfigurasikan jendela pemeliharaan untuk sumber daya AWS terkelola, seperti database Amazon Relational Database Service (RDSAmazon). Menggunakan pemindaian kode statis untuk memeriksa masalah-masalah umum pada kode sumber aplikasi. Pertimbangkan untuk melakukan uji penetrasi aplikasi web jika organisasi Anda memiliki keterampilan yang diperlukan atau dapat menggunakan bantuan dari luar.

Anti-pola umum:

- Tidak memiliki program manajemen kerentanan.
- Menjalankan patching sistem tanpa mempertimbangkan tingkat keparahan atau penghindaran risiko.
- Menggunakan perangkat lunak yang telah melewati tanggal akhir masa pakai () EOL yang disediakan vendor.
- Melakukan deployment kode ke dalam lingkungan produksi sebelum menganalisis masalah keamanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Program manajemen kerentanan mencakup penilaian keamanan, mengidentifikasi masalah, memprioritaskan, dan menjalankan operasi patching sebagai bagian dari penyelesaian masalah. Otomatisasi adalah kunci agar dapat terus memindai beban kerja untuk menemukan masalah dan paparan jaringan yang tidak diinginkan serta melakukan perbaikan. Melakukan otomatisasi terhadap pembuatan dan pembaruan sumber daya dapat menghemat waktu dan mengurangi risiko kesalahan konfigurasi yang menciptakan masalah lebih lanjut. Program manajemen kerentanan yang dirancang dengan baik juga harus mempertimbangkan untuk melakukan pengujian kerentanan selama tahap pengembangan dan deployment siklus hidup perangkat lunak. Mengimplementasikan manajemen kerentanan selama pengembangan dan deployment dapat membantu mengurangi kemungkinan kerentanan masuk ke lingkungan produksi Anda.

Untuk mengimplementasikan program manajemen kerentanan membutuhkan pemahaman yang baik tentang model Tanggung Jawab Bersama AWS dan bagaimana kaitan model tersebut dengan beban kerja spesifik Anda. Di bawah Model Tanggung Jawab Bersama, AWS bertanggung jawab untuk melindungi infrastruktur AWS Cloud. Infrastruktur ini terdiri dari perangkat keras, perangkat lunak, jaringan, dan fasilitas yang menjalankan AWS Cloud layanan. Anda bertanggung jawab atas keamanan di cloud, misalnya, data aktual, konfigurasi keamanan, dan tugas manajemen EC2 instans Amazon, dan memverifikasi bahwa objek Amazon S3 Anda diklasifikasikan dan dikonfigurasi dengan benar. Pendekatan terhadap manajemen kerentanan juga dapat bervariasi, bergantung pada layanan yang digunakan. Misalnya, AWS mengelola patching untuk layanan database relasional terkelola kamiRDS, Amazon, tetapi Anda akan bertanggung jawab untuk menambal database yang dihosting sendiri.

AWS memiliki berbagai layanan untuk membantu program manajemen kerentanan Anda. Amazon Inspector terus memindai AWS beban kerja untuk masalah perangkat lunak dan akses jaringan yang tidak diinginkan. AWS Systems Manager Patch Manager membantu mengelola patching di seluruh EC2 instans Amazon Anda. Amazon Inspector and Systems Manager dapat dilihat di AWS Security Hub, layanan manajemen postur keamanan cloud yang membantu mengotomatiskan pemeriksaan AWS keamanan dan memusatkan peringatan keamanan.

Amazon CodeGuru dapat membantu mengidentifikasi potensi masalah dalam aplikasi Java dan Python menggunakan analisis kode statis.

Langkah-langkah implementasi

- Konfigurasikan Amazon Inspector: Amazon Inspector secara otomatis mendeteksi instans EC2 Amazon yang baru diluncurkan, fungsi Lambda, dan gambar kontainer yang memenuhi syarat yang didorong ke ECR Amazon dan segera memindai instans Amazon yang baru diluncurkan, kemungkinan cacat, dan paparan jaringan yang tidak diinginkan.
- Pindai kode sumber: Lakukan pemindaian terhadap pustaka dan dependensi untuk mencari masalah dan cacat. Amazon CodeGuru dapat memindai dan memberikan rekomendasi untuk memulihkan masalah keamanan umum untuk aplikasi Java dan Python. OWASPYayasan menerbitkan daftar Alat Analisis Kode Sumber (juga dikenal sebagai SAST alat).
- Terapkan mekanisme untuk melakukan pemindaian dan patching terhadap lingkungan Anda yang ada, serta pemindaian sebagai bagian dari proses pembuatan pipeline CI/CD: Implementasikan mekanisme untuk memindai dan memasang patch untuk masalah-masalah yang ada dalam dependensi dan sistem operasi Anda untuk membantu melindungi dari ancaman baru. Jalankan mekanisme tersebut secara rutin. Anda harus memahami manajemen kerentanan perangkat lunak saat Anda ingin menerapkan patch atau mengatasi masalah perangkat lunak. Prioritaskan perbaikan potensi masalah keamanan dengan menanamkan penilaian kerentanan lebih awal ke dalam pipeline integrasi berkelanjutan/pengiriman berkelanjutan (CI/CD). Pendekatan Anda dapat bervariasi berdasarkan AWS layanan yang Anda konsumsi. Untuk memeriksa potensi masalah dalam perangkat lunak yang berjalan di EC2 instans Amazon, tambahkan Amazon Inspector ke pipeline Anda untuk memberi tahu Anda dan menghentikan proses pembuatan jika masalah atau potensi cacat terdeteksi. Amazon Inspector akan selalu memantau sumber daya. Anda juga dapat menggunakan produk open source seperti OWASPDependency-Check, Snyk, Open, manajer paketVAS, dan alat untuk manajemen kerentanan. AWS Partner
- Penggunaan AWS Systems Manager: Anda bertanggung jawab atas pengelolaan tambalan untuk AWS sumber daya Anda, termasuk instans Amazon Elastic Compute Cloud (AmazonEC2), Amazon Machine Images (AMIs), dan sumber daya komputasi lainnya. AWS Systems Manager

Patch Manager mengotomatiskan proses patching instans terkelola dengan pembaruan terkait keamanan dan jenis pembaruan lainnya. Patch Manager dapat digunakan untuk menerapkan tambalan pada EC2 instans Amazon untuk sistem operasi dan aplikasi, termasuk aplikasi Microsoft, paket layanan Windows, dan peningkatan versi minor untuk instance berbasis Linux. Selain AmazonEC2, Patch Manager juga dapat digunakan untuk menambal server lokal.

Untuk daftar sistem operasi yang didukung, silakan lihat Sistem operasi yang didukung di Panduan Pengguna Systems Manager. Anda dapat memindai instans untuk melihat laporan patch yang hilang saja, atau Anda dapat memindai dan secara otomatis menginstal semua patch yang hilang.

- Penggunaan AWS Security Hub: Security Hub memberikan pandangan komprehensif tentang status keamanan Anda AWS. Ini mengumpulkan data keamanan di berbagai AWS layanan dan memberikan temuan tersebut dalam format standar, memungkinkan Anda memprioritaskan temuan keamanan di seluruh layanan. AWS
- Use AWS CloudFormation: AWS CloudFormation adalah sebuah layanan infrastruktur sebagai kode (IaC) yang dapat membantu manajemen kerentanan dengan mengotomatiskan deployment sumber daya dan menstandardisasi arsitektur sumber daya di beberapa akun dan lingkungan.

Sumber daya

Dokumen terkait:

- AWS Systems Manager
- Ikhtisar Keamanan AWS Lambda
- Amazon CodeGuru
- Manajemen Kerentanan Otomatis dan Ditingkatkan untuk Beban Kerja Cloud dengan Amazon Inspector Baru
- Mengotomatiskan manajemen kerentanan dan remediasi dalam menggunakan Amazon AWS Inspector dan - Bagian 1 AWS Systems Manager

Video terkait:

- Mengamankan Layanan Kontainer dan Nirserver
- Praktik terbaik keamanan untuk layanan EC2 metadata instans Amazon

### SEC06-BP02 Komputasi ketentuan dari gambar yang diperkeras

Kurangi peluang untuk akses yang tidak diinginkan ke lingkungan runtime Anda dengan melakukan deployment-nya dari citra yang diperkeras (hardened images). Dapatkan dependensi runtime, seperti citra kontainer dan pustaka aplikasi, hanya dari registri tepercaya dan verifikasikan tanda tangannya. Buat registri privat Anda sendiri untuk menyimpan citra dan pustaka tepercaya yang akan digunakan dalam proses build dan deployment Anda.

Hasil yang diinginkan: Sumber daya komputasi Anda disediakan dari image dasar yang diperkeras. Anda mengambil dependensi eksternal, seperti citra kontainer dan pustaka aplikasi, hanya dari registri tepercaya dan memverifikasi tanda tangannya. Image dan pustaka ini disimpan dalam registri privat untuk dirujuk oleh proses build dan deployment Anda. Anda memindai dan memperbarui citra dan dependensi secara rutin untuk membantu Anda melindungi terhadap kerentanan yang baru ditemukan.

Anti-pola umum:

- Mendapatkan citra dan pustaka dari registri tepercaya, tetapi tidak memverifikasi tanda tangannya atau melakukan pemindaian kerentanan sebelum menggunakannya.
- Melakukan pengerasan citra, tetapi tidak mengujinya secara rutin untuk menemukan kerentanan baru atau memperbaruinya ke versi terkini.
- Menginstal atau tidak menghapus paket-paket perangkat lunak yang tidak diperlukan selama perkiraan siklus hidup citra.
- Hanya mengandalkan patching untuk menjaga sumber daya komputasi produksi tetap mutakhir. Patching saja masih dapat menyebabkan sumber daya komputasi menyimpang dari standar yang diperkeras dari waktu ke waktu. Patching juga dapat gagal menghapus malware yang mungkin telah diinstal oleh pelaku ancaman selama terjadi peristiwa keamanan.

Manfaat menerapkan praktik terbaik ini: Pengerasan gambar akan membantu mengurangi jumlah jalur yang tersedia di lingkungan runtime Anda yang dapat memungkinkan akses yang tidak diinginkan ke pengguna atau layanan yang tidak sah. Hal ini juga dapat mengurangi cakupan dampak jika terjadi akses yang tidak diinginkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk melakukan pengerasan terhadap sistem Anda, mulailah dari versi sistem operasi, citra kontainer, dan pustaka aplikasi terbaru. Terapkan patch untuk masalah yang diketahui. Minimalkan

sistem dengan menghapus aplikasi-aplikasi, layanan, driver perangkat, pengguna default, dan kredensial lainnya yang tidak diperlukan. Lakukan tindakan-tindakan lain yang diperlukan, seperti menonaktifkan port untuk membuat lingkungan yang hanya memiliki sumber daya dan kemampuan yang dibutuhkan oleh beban kerja Anda. Dari acuan dasar ini, Anda kemudian dapat menginstal perangkat lunak, agen, atau proses lain yang Anda butuhkan untuk berbagai tujuan, seperti pemantauan beban kerja atau manajemen kerentanan.

Anda dapat mengurangi beban sistem pengerasan dengan menggunakan panduan yang disediakan sumber tepercaya, seperti Center for Internet Security (CIS) dan Defense Information Systems Agency () Security Technical Implementation Guides (STIGs). DISA Kami menyarankan Anda memulai dengan Amazon Machine Image (AMI) yang diterbitkan oleh AWS atau APN mitra, dan menggunakan AWS EC2Image Builder untuk mengotomatiskan konfigurasi sesuai dengan kombinasi CIS dan STIG kontrol yang sesuai.

Meskipun ada gambar keras yang tersedia dan resep EC2 Image Builder yang menerapkan CIS atau DISA STIG rekomendasi, Anda mungkin menemukan konfigurasi mereka mencegah perangkat lunak Anda berjalan dengan sukses. Dalam situasi ini, Anda dapat memulai dari gambar dasar yang tidak dikeraskan, menginstal perangkat lunak Anda, dan kemudian secara bertahap menerapkan CIS kontrol untuk menguji dampaknya. Untuk CIS kontrol apa pun yang mencegah perangkat lunak Anda berjalan, uji apakah Anda dapat menerapkan rekomendasi pengerasan berbutir halus sebagai gantinya. DISA Lacak berbagai CIS kontrol dan DISA STIG konfigurasi yang dapat Anda terapkan dengan sukses. Gunakan ini untuk menentukan resep pengerasan gambar Anda di EC2 Image Builder yang sesuai.

Untuk beban kerja kontainer, gambar yang dikeraskan dari Docker tersedia di repositori publik Amazon Elastic Container Registry (). ECR Anda dapat menggunakan EC2 Image Builder untuk mengeraskan gambar kontainer di sampingnyaAMIs.

Mirip dengan sistem operasi dan gambar kontainer, Anda dapat memperoleh paket kode (atau pustaka) dari repositori publik, melalui perkakas seperti pip, npm, Maven, dan. NuGet Kami menyarankan Anda untuk mengelola paket kode dengan mengintegrasikan repositori pribadi, seperti dalam AWS CodeArtifact, dengan repositori publik tepercaya. Integrasi ini dapat menangani pengambilan, penyimpanan, dan penyimpanan paket up-to-date untuk Anda. Proses pembuatan aplikasi Anda kemudian dapat memperoleh dan menguji versi terbaru dari paket-paket ini bersama aplikasi Anda, menggunakan teknik seperti Analisis Komposisi Perangkat Lunak (SCA), Pengujian Keamanan Aplikasi Statis (SAST), dan Pengujian Keamanan Aplikasi Dinamis (DAST).

Untuk beban kerja tanpa server yang digunakan AWS Lambda, sederhanakan pengelolaan dependensi paket menggunakan lapisan Lambda. Gunakan lapisan Lambda untuk mengonfigurasi

sekumpulan dependensi standar yang ada di berbagai fungsi ke dalam arsip mandiri. Anda dapat membuat dan memelihara lapisan melalui proses pembuatannya sendiri, menyediakan cara sentral agar fungsi Anda tetap ada up-to-date.

Langkah-langkah implementasi

- Lakukan pengerasan terhadap sistem operasi. Gunakan gambar dasar dari sumber tepercaya sebagai fondasi untuk membangun pengerasan Anda. AMIs Gunakan EC2Image Builder untuk membantu menyesuaikan perangkat lunak yang diinstal pada gambar Anda.
- Lakukan pengerasan terhadap sumber daya terkontainerisasi. Konfigurasikan sumber daya terkontainerisasi untuk memenuhi praktik-praktik terbaik keamanan. Saat menggunakan kontainer, terapkan Pemindaian ECR Gambar di pipeline build Anda dan secara teratur terhadap repositori gambar Anda untuk dicari CVEs di container Anda.
- Saat menggunakan implementasi tanpa server dengan AWS Lambda, gunakan lapisan Lambda untuk memisahkan kode fungsi aplikasi dan pustaka dependen bersama. Konfigurasikan penandatanganan kode untuk Lambda untuk memastikan bahwa hanya kode tepercaya yang berjalan dalam fungsi Lambda Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP05 Lakukan manajemen tambalan

Video terkait:

- Menyelam jauh ke dalam AWS Lambda keamanan

Contoh terkait:

- Membangun STIG -compliant dengan cepat AMI menggunakan EC2 Image Builder
- Membangun image kontainer yang lebih baik
- Menggunakan lapisan Lambda untuk menyederhanakan proses pengembangan Anda
- Mengembangkan & Menyebarkan AWS Lambda Layers menggunakan Serverless Framework
- Membangun pipa end-to-end AWS DevSecOps CI/CD dengan open sourceSCA, SAST dan alat DAST

SEC06-BP03 Mengurangi manajemen manual dan akses interaktif

Gunakan otomatisasi sebisa mungkin untuk melakukan tugas deployment, konfigurasi, pemeliharaan, dan investigasi. Pertimbangkan akses manual ke sumber daya komputasi saat melakukan prosedur darurat atau berada dalam lingkungan yang aman (sandbox), ketika otomatisasi tidak tersedia.

Hasil yang diinginkan: Skrip terprogram dan dokumen otomatisasi (runbook) merekam tindakan resmi pada sumber daya komputasi Anda. Runbook ini dimulai baik secara otomatis, melalui sistem deteksi perubahan, maupun secara manual, ketika penilaian oleh manusia diperlukan. Akses langsung ke sumber daya komputasi hanya tersedia dalam situasi darurat ketika otomatisasi tidak tersedia. Semua aktivitas manual dicatat lognya dan dimasukkan ke dalam proses peninjauan untuk terus meningkatkan kemampuan-kemampuan otomatisasi Anda.

Anti-pola umum:

- Akses interaktif ke EC2 instans Amazon dengan protokol seperti atau. SSH RDP
- Mempertahankan login pengguna individu seperti `/etc/passwd` atau pengguna lokal Windows.
- Berbagi kata sandi atau kunci privat untuk mengakses instans di antara beberapa pengguna.
- Menginstal perangkat lunak dan membuat atau memperbarui file konfigurasi secara manual.
- Memperbarui atau melakukan patching perangkat lunak secara manual.
- Masuk ke instans untuk memecahkan masalah.

Manfaat menerapkan praktik terbaik ini: Melakukan tindakan dengan otomatisasi dapat membantu Anda mengurangi risiko operasional dari perubahan dan kesalahan konfigurasi yang tidak diinginkan. Menghapus penggunaan Secure Shell (SSH) dan Remote Desktop Protocol (RDP) untuk akses interaktif mengurangi cakupan akses ke sumber daya komputasi Anda. Hal ini akan menghilangkan jalur umum yang memungkinkan tindakan tidak sah. Pencatatan tugas manajemen sumber daya komputasi Anda dalam dokumen otomatisasi dan skrip programatis akan menyediakan sebuah mekanisme untuk menentukan dan mengaudit cakupan penuh aktivitas yang sah secara lebih mendetail.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Masuk ke instans adalah pendekatan klasik untuk administrasi sistem. Setelah menginstal sistem operasi server, pengguna biasanya akan masuk log in secara manual untuk mengonfigurasi

sistem dan menginstal perangkat lunak yang diinginkan. Selama masa pakai server, pengguna mungkin masuk log in untuk melakukan pembaruan perangkat lunak, menerapkan patch, mengubah konfigurasi, dan memecahkan masalah.

Namun demikian, akses secara manual dapat menimbulkan sejumlah risiko. Ini membutuhkan server yang mendengarkan permintaan, seperti SSH atau RDP layanan, yang dapat memberikan jalur potensial untuk akses yang tidak sah. Ini juga meningkatkan risiko kesalahan manusia yang terkait dengan pelaksanaan langkah-langkah secara manual. Hal ini dapat mengakibatkan insiden beban kerja, kerusakan atau pemusnahan data, atau masalah-masalah keamanan lainnya. Akses manusia juga memerlukan perlindungan dari tindakan berbagi kredensial, yang bisa menimbulkan biaya overhead manajemen tambahan.

Untuk mengurangi risiko ini, Anda dapat menerapkan solusi akses jarak jauh berbasis agen, seperti Systems Manager.AWS AWS Systems Manager Agen (SSMAgen) memulai saluran terenkripsi dan dengan demikian tidak bergantung pada mendengarkan permintaan yang dimulai secara eksternal. Pertimbangkan untuk mengonfigurasi SSM Agen untuk membuat saluran ini melalui titik VPC akhir.

Systems Manager memberikan Anda kontrol terperinci tentang cara berinteraksi dengan instans terkelola. Anda dapat menentukan otomatisasi yang akan dijalankan, siapa yang bisa menjalankannya, dan kapan mereka bisa menjalankannya. Systems Manager dapat menerapkan patch, menginstal perangkat lunak, dan membuat perubahan konfigurasi tanpa akses interaktif ke instans. Systems Manager juga dapat menyediakan akses ke shell jarak jauh dan mencatat setiap perintah yang dipanggil, dan outputnya, selama sesi ke log dan Amazon S3. AWS CloudTrailmencatat pemanggilan Systems Manager APIs untuk inspeksi.

Langkah-langkah implementasi

1. Instal Agen AWS Systems Manager (SSMAgen) di EC2 instans Amazon Anda. Periksa untuk melihat apakah SSM Agen disertakan dan dimulai secara otomatis sebagai bagian dari AMI konfigurasi dasar Anda.
2. Verifikasi bahwa IAM Peran yang terkait dengan profil EC2 instans Anda menyertakan IAMkebijakan `AmazonSSMManagedInstanceCore` terkelola.
3. Nonaktifkan SSHRDP,, dan layanan akses jarak jauh lainnya yang berjalan pada instans Anda. Anda dapat melakukan ini dengan menjalankan skrip yang dikonfigurasi di bagian data pengguna dari templat peluncuran Anda atau dengan membangun yang disesuaikan AMIs dengan alat seperti EC2 Image Builder.
4. Verifikasi bahwa aturan masuknya grup keamanan yang berlaku untuk EC2 instans Anda tidak mengizinkan akses pada port 22/tcp (SSH) atau port 3389/tcp (). RDP Implementasikan deteksi

dan peringatan terkait grup keamanan yang salah konfigurasi dengan menggunakan layanan-layanan seperti AWS Config.

5. Tentukan otomatisasi dan runbook yang sesuai, lalu jalankan perintah di Systems Manager. Gunakan IAM kebijakan untuk menentukan siapa yang dapat melakukan tindakan ini dan kondisi di mana mereka diizinkan. Uji otomatisasi ini secara menyeluruh di sebuah lingkungan non-produksi. Lakukan invokasi terhadap otomatisasi ini jika diperlukan, bukan mengakses instans secara interaktif.
6. Gunakan AWS Systems Manager Session Manager untuk menyediakan akses interaktif ke instans bila diperlukan. Aktifkan pencatatan aktivitas sesi untuk mempertahankan jejak audit di Amazon CloudWatch Log atau Amazon S3.

Sumber daya

Praktik-praktik terbaik terkait:

- REL08-BP04 Terapkan menggunakan infrastruktur yang tidak dapat diubah

Contoh terkait:

- Mengganti SSH akses untuk mengurangi overhead manajemen dan keamanan dengan AWS Systems Manager

Alat terkait:

- AWS Systems Manager

Video terkait:

- Mengontrol Akses Sesi Pengguna ke Instans di Manajer AWS Systems Manager Sesi

SEC06-BP04 Validasi integritas perangkat lunak

Gunakan verifikasi kriptografis untuk memvalidasi integritas artefak perangkat lunak (termasuk citra) yang digunakan beban kerja Anda.  Tanda tangani perangkat lunak Anda secara kriptografis sebagai perlindungan terhadap perubahan-perubahan tidak sah yang berjalan di lingkungan komputasi Anda.

Hasil yang diinginkan: Semua artefak diperoleh dari sumber-sumber yang tepercaya. Sertifikat situs web vendor divalidasi. Artefak yang diunduh sudah diverifikasi secara kriptografis berdasarkan tanda tangannya. Perangkat lunak Anda sendiri ditandatangani secara kriptografis dan diverifikasi oleh lingkungan-lingkungan komputasi Anda.

Anti-pola umum:

- Memercayai situs web vendor terkemuka untuk mendapatkan artefak perangkat lunak, tetapi mengabaikan pemberitahuan tentang sertifikat yang kedaluwarsa. Melanjutkan pengunduhan tanpa mengonfirmasi bahwa sertifikatnya valid.
- Memvalidasi sertifikat situs web vendor, tetapi tidak secara kriptografis memverifikasi artefak yang diunduh dari situs web ini.
- Hanya mengandalkan digest atau hash untuk memvalidasi integritas perangkat lunak. Hash menetapkan bahwa artefak belum dimodifikasi dari versi aslinya, tetapi tidak memvalidasi sumbernya.
- Tidak menandatangani perangkat lunak, kode, atau pustaka Anda sendiri, meskipun hanya digunakan dalam deployment Anda sendiri.

Manfaat menerapkan praktik terbaik ini: Memvalidasi integritas artefak yang bergantung pada beban kerja Anda akan membantu Anda dalam mencegah malware memasuki lingkungan komputasi Anda. Menandatangani perangkat lunak Anda akan membantu melindungi dari eksekusi yang tidak sah di lingkungan komputasi Anda. Amankan rantai pasokan perangkat lunak Anda dengan menandatangani dan memverifikasi kodenya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Citra sistem operasi, citra kontainer, dan artefak kode sering kali didistribusikan dengan pemeriksaan integritas yang tersedia, seperti melalui digest atau hash. Hal ini memungkinkan klien untuk memverifikasi integritas dengan melakukan komputasi hash atas payload mereka sendiri dan memastikan itu sama dengan yang dipublikasikan. Meskipun pemeriksaan ini membantu Anda untuk memverifikasi bahwa muatan belum dirusak, namun pemeriksaan tersebut tidak memvalidasi muatan yang berasal dari sumber aslinya (asalnya). Verifikasi asal-usul mengharuskan adanya sertifikat yang dikeluarkan oleh otoritas tepercaya untuk menandatangani artefaknya secara digital.

Jika Anda menggunakan sebuah perangkat lunak atau artefak unduhan dalam beban kerja Anda, periksa apakah penyedianya menyediakan kunci publik untuk verifikasi tanda tangan digital. Berikut

ini adalah beberapa contoh cara AWS menyediakan kunci publik dan instruksi verifikasi untuk perangkat lunak yang kami publikasikan:

- EC2Image Builder: Verifikasi tanda tangan unduhan AWS TOE instalasi
- AWS Systems Manager: Memverifikasi tanda tangan SSM Agen
- Amazon CloudWatch: Memverifikasi tanda tangan paket CloudWatch agen

Masukkan verifikasi tanda tangan digital ke dalam proses yang Anda gunakan untuk memperoleh dan mengeraskan gambar, seperti yang dibahas dalam SEC06-BP02 Komputasi ketentuan dari gambar yang diperkeras.

Anda dapat menggunakannya AWS Signer untuk membantu Anda mengelola verifikasi tanda tangan, serta siklus hidup penandatanganan kode Anda sendiri untuk perangkat lunak dan artefak Anda sendiri.  Keduanya, AWS Lambda dan Amazon Elastic Container Registry menyediakan integrasi dengan Signer untuk memverifikasi tanda tangan kode dan image Anda.  Menggunakan contoh di bagian Sumber Daya, Anda dapat menerapkan Signer ke dalam pipeline integrasi dan pengiriman berkelanjutan (CI/CD) untuk mengotomatiskan verifikasi tanda tangan serta penandatanganan kode dan image Anda sendiri.

Sumber daya

Dokumen terkait:

- Penandatanganan Kriptografi untuk Kontainer
- Praktik Terbaik untuk membantu mengamankan pipeline build image container Anda dengan menggunakan AWS Signer
- Mengumumkan Penandatanganan Gambar Kontainer dengan AWS Signer dan Amazon EKS
- Mengkonfigurasi penandatanganan kode untuk AWS Lambda
- Praktik-praktik terbaik dan pola lanjutan untuk penandatanganan kode Lambda
- Penandatanganan kode menggunakan CA AWS Certificate Manager Pribadi dan kunci AWS Key Management Service asimetris

Contoh terkait:

- Otomatiskan penandatanganan kode Lambda dengan Amazon dan CodeCatalyst AWS Signer
- Menandatangani dan Memvalidasi OCI Artefak dengan AWS Signer

Alat terkait:

- AWS Lambda
- AWS Signer
- AWS Certificate Manager
- AWS Key Management Service
- AWS CodeArtifact

SEC06-BP05 Mengotomatiskan perlindungan komputasi

Lakukan otomatisasi terhadap operasi perlindungan komputasi untuk mengurangi kebutuhan akan intervensi manusia. Gunakan pemindaian otomatis untuk mendeteksi adanya potensi masalah yang mungkin terjadi dalam sumber daya komputasi Anda, dan lakukan remediasi dengan respons programatis otomatis atau operasi manajemen armada. Gabungkan otomatisasi dalam proses CI/CD Anda untuk menyebarkan beban kerja yang dapat dipercaya dengan dependensi. up-to-date

Hasil yang diinginkan: Sistem otomatis melakukan semua pemindaian dan penambalan atas sumber daya komputasi. Anda menggunakan verifikasi otomatis untuk memeriksa apakah gambar dan dependensi perangkat lunak berasal dari sumber tepercaya, dan belum dirusak. Beban kerja secara otomatis diperiksa untuk up-to-date dependensi, dan ditandatangani untuk membangun kepercayaan di lingkungan komputasi. AWS Remediasi otomatis dimulai ketika ada sumber daya yang tidak mematuhi persyaratan terdeteksi.

Anti-pola umum:

- Mengikuti praktik infrastruktur tak dapat diubah, tetapi tidak memiliki solusi untuk melakukan patching atau penggantian darurat sistem produksi.
- Menggunakan otomatisasi untuk memperbaiki sumber daya yang salah konfigurasi, tetapi tidak memiliki mekanisme untuk melakukan penimpaan secara manual. Kesulitan mungkin akan muncul saat Anda perlu menyesuaikan persyaratan, dan Anda mungkin perlu menangguhkan otomatisasi sampai Anda membuat perubahan-perubahan ini.

Manfaat menerapkan praktik terbaik ini: Otomatisasi dapat mengurangi risiko akses dan penggunaan sumber daya komputasi Anda yang tidak sah. Hal ini akan membantu Anda mencegah terjadinya kesalahan konfigurasi yang masuk ke lingkungan produksi, serta mendeteksi dan memperbaiki kesalahan konfigurasi jika terjadi. Otomatisasi juga akan membantu Anda mendeteksi akses dan

penggunaan yang tidak sah atas sumber daya komputasi untuk mempercepat waktu respons Anda. Hal ini pada gilirannya dapat mengurangi cakupan dampak yang ditimbulkan masalah secara keseluruhan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Anda dapat menerapkan otomatisasi yang dijelaskan dalam praktik Pilar Keamanan untuk memberikan proteksi terhadap melindungi sumber daya komputasi Anda. SEC06-BP01 Melakukan manajemen kerentanan menjelaskan cara Anda dapat menggunakan Amazon Inspector di pipeline CI/CD Anda dan untuk terus memindai lingkungan runtime Anda untuk Kerentanan Umum dan Eksposur () yang diketahui. CVEs Anda dapat menggunakan AWS Systems Manager untuk menerapkan pacth atau melakukan deployment ulang atas image gambar baru melalui runbook otomatis agar armada komputasi Anda diperbarui dengan perangkat lunak dan pustaka terbaru. Gunakan teknik-teknik ini untuk mengurangi kebutuhan akan proses-proses manual dan akses interaktif ke sumber daya komputasi Anda. Lihat SEC06-BP03 Mengurangi manajemen manual dan akses interaktif untuk mempelajari lebih lanjut.

Otomasi juga berperan dalam menyebarkan beban kerja yang dapat dipercaya, dijelaskan dalam SEC06-BP02 Komputasi ketentuan dari gambar yang diperkeras dan 06-BP04 Validasi integritas perangkat lunak. SEC Anda dapat menggunakan layanan seperti EC2Image Builder,, AWS SignerAWS CodeArtifact, dan Amazon Elastic Container Registry (ECR) untuk mengunduh, memverifikasi, membuat, dan menyimpan gambar dan dependensi kode yang diperkeras dan disetujui. Di samping Inspector, masing-masing dapat berperan dalam proses CI/CD Anda sehingga beban kerja Anda masuk ke produksi hanya ketika dikonfirmasi bahwa dependensinya berasal dan dari sumber tepercaya. up-to-date Beban kerja Anda juga ditandatangani sehingga lingkungan AWS komputasi, seperti AWS Lambdadan Amazon Elastic EKS Kubernetes Service () dapat memverifikasi bahwa itu belum dirusak sebelum mengizinkannya dijalankan.

Selain kontrol-kontrol preventif ini, Anda juga dapat menggunakan otomatisasi dalam kontrol-kontrol detektif untuk sumber daya komputasi Anda. Sebagai salah satu contoh, AWS Security Hubmenawarkan standar NIST800-53 Rev. 5 yang mencakup pemeriksaan seperti EC2instance [EC2.8] harus menggunakan Instance Metadata Service Version 2 (). IMDSv2 IMDSv2menggunakan teknik otentikasi sesi, memblokir permintaan yang berisi X-Forwarded-For HTTP header, dan jaringan TTL 1 untuk menghentikan lalu lintas yang berasal dari sumber eksternal untuk mengambil informasi tentang instance. EC2 Pemeriksaan di Security Hub ini dapat mendeteksi kapan EC2 instans menggunakan IMDSv1 dan memulai remediasi otomatis. Pelajari lebih lanjut tentang deteksi dan remediasi otomatis di SEC04-BP04 Memulai remediasi untuk sumber daya yang tidak sesuai.

Langkah-langkah implementasi

1. Otomatiskan pembuatan yang aman, sesuai, dan diperkeras dengan Image AMIs BuilderEC2. Anda dapat menghasilkan gambar yang menggabungkan kontrol dari standar Center for Internet Security (CIS) Benchmark atau Security Technical Implementation Guide (STIG) dari gambar dasar AWS dan APN mitra.
2. Melakukan otomatisasi manajemen konfigurasi. Berlakukan dan validasikan konfigurasi-konfigurasi yang aman di sumber daya komputasi Anda secara otomatis dengan menggunakan layanan atau alat manajemen konfigurasi.
   a. Manajemen konfigurasi otomatis menggunakan AWS Config
   b. Keamanan otomatis dan manajemen postur kepatuhan dengan menggunakan AWS Security Hub
3. Otomatiskan penambalan atau penggantian instans Amazon Elastic Compute Cloud (AmazonEC2). AWS Systems Manager Patch Manager mengotomatiskan proses menambal instance terkelola dengan pembaruan terkait keamanan dan jenis pembaruan lainnya. Anda dapat menggunakan Patch Manager untuk menerapkan patch untuk kedua sistem operasi dan aplikasi.
   a. AWS Manajer Patch Systems Manager
4. Otomatiskan pemindaian sumber daya komputasi untuk kerentanan dan eksposur umum (CVEs), dan sematkan solusi pemindaian keamanan dalam pipeline build Anda.
   a. Amazon Inspector
   b. ECRPemindaian Gambar
5. Pertimbangkan Amazon GuardDuty untuk malware otomatis dan deteksi ancaman untuk melindungi sumber daya komputasi. GuardDuty juga dapat mengidentifikasi potensi masalah ketika suatu AWS Lambdafungsi dipanggil di AWS lingkungan Anda.
   a. Amazon GuardDuty
6. Pertimbangkan solusi AWS Mitra. AWS Mitra menawarkan produk terdepan di industri yang setara, identik, atau terintegrasi dengan kontrol yang ada di lingkungan lokal Anda. Produk-produk ini akan melengkapi layanan-layanan AWS yang ada untuk memungkinkan Anda melakukan deployment arsitektur keamanan yang menyeluruh dan pengalaman yang lebih lancar di seluruh lingkungan cloud dan on-premise Anda.
   a. Keamanan infrastruktur

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP06 Mengotomatiskan penerapan kontrol keamanan standar

Dokumen terkait:

- Dapatkan manfaat penuh IMDSv2 dan nonaktifkan IMDSv1 di seluruh AWS infrastruktur Anda

Video terkait:

- Praktik terbaik keamanan untuk layanan EC2 metadata instans Amazon

## Perlindungan data

Pertanyaan

- SEC7. Bagaimana cara mengklasifikasikan data Anda?
- SEC8. Bagaimana cara melindungi data diam Anda?
- SEC9. Bagaimana cara melindungi data bergerak Anda?

### SEC7. Bagaimana cara mengklasifikasikan data Anda?

Klasifikasi memberikan cara untuk mengategorikan data, berdasarkan tingkat kekritisan dan sensitivitas untuk membantu Anda menentukan kontrol retensi dan perlindungan yang sesuai.

Praktik terbaik

- SEC07-BP01 Memahami skema klasifikasi data Anda
- SEC07-BP02 Menerapkan kontrol perlindungan data berdasarkan sensitivitas data
- SEC07-BP03 Mengotomatiskan identifikasi dan klasifikasi
- SEC07-BP04 Tentukan manajemen siklus hidup data yang dapat diskalakan

#### SEC07-BP01 Memahami skema klasifikasi data Anda

Pahami klasifikasi data yang diproses beban kerja Anda, persyaratan penanganannya, proses-proses bisnis terkait, di mana data disimpan, dan siapa pemilik data.  Skema klasifikasi dan penanganan data Anda harus mempertimbangkan persyaratan-persyaratan hukum dan kepatuhan yang berlaku atas beban kerja Anda serta kontrol data apa yang diperlukan. Pemahaman terhadap data adalah langkah pertama dalam perjalanan klasifikasi data.

Hasil yang diinginkan: Jenis data yang ada dalam beban kerja Anda dapat dipahami dan didokumentasikan dengan baik. Kontrol yang tepat diterapkan untuk melindungi data sensitif berdasarkan klasifikasinya. Kontrol ini mengatur berbagai pertimbangan, seperti siapa yang diizinkan mengakses data dan untuk tujuan apa, di mana data disimpan, kebijakan enkripsi yang diterapkan untuk data tersebut dan bagaimana kunci enkripsi dikelola, siklus hidup untuk data dan persyaratan retensinya, proses pemusnahan yang tepat, proses pencadangan dan pemulihan apa yang diterapkan, serta audit akses.

Anti-pola umum:

- Tidak memiliki kebijakan klasifikasi data formal untuk menentukan tingkat sensitivitas data dan persyaratan-persyaratan penanganannya
- Tidak memiliki pemahaman yang baik tentang tingkat sensitivitas data dalam beban kerja Anda, dan tidak mencatat informasi ini dalam dokumentasi arsitektur dan operasi
- Gagal menerapkan kontrol-kontrol yang sesuai terhadap data Anda berdasarkan sensitivitas dan persyaratannya, sebagaimana yang diuraikan dalam kebijakan klasifikasi dan penanganan data Anda
- Gagal memberikan umpan balik tentang persyaratan-persyaratan klasifikasi dan penanganan data kepada pemilik kebijakan.

Manfaat menerapkan praktik terbaik ini: Praktik ini akan menghilangkan ambiguitas yang mungkin muncul di seputar penanganan data yang tepat dalam beban kerja Anda. Penerapan kebijakan formal yang menentukan tingkat sensitivitas data di organisasi Anda dan perlindungan yang diperlukan dapat membantu Anda dalam mematuhi peraturan hukum serta pengesahan dan sertifikasi keamanan siber lainnya. Pemilik beban kerja dapat merasa yakin dengan mengetahui di mana data sensitif disimpan dan kontrol perlindungan apa yang diterapkan. Dengan mencatat hal ini dalam dokumentasi, anggota tim baru akan dapat lebih memahaminya dan dapat memelihara berbagai kontrol di awal masa kerja mereka. Praktik-praktik ini juga dapat membantu Anda mengurangi biaya dengan melakukan penyesuaian ukuran yang tepat terhadap kontrol untuk masing-masing jenis data.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Saat merancang sebuah beban kerja, Anda mungkin mempertimbangkan cara untuk melindungi data sensitif secara intuitif. Misalnya, dalam sebuah aplikasi multi-tenant, data setiap tenant secara intuitif dianggap sebagai data sensitif dan perlindungan diterapkan agar satu tenant tidak dapat mengakses

data tenant yang lain. Demikian juga, Anda dapat secara intuitif merancang kontrol akses sehingga hanya administrator yang dapat melakukan modifikasi data, sedangkan pengguna yang lain hanya memiliki akses tingkat baca atau tidak memiliki akses sama sekali.

Dengan menetapkan dan mencatat tingkat sensitivitas data ini dalam kebijakan, bersama dengan persyaratan-persyaratan perlindungan datanya, Anda dapat secara formal mengidentifikasi data apa yang berada dalam beban kerja Anda. Anda kemudian dapat menentukan apakah kontrol yang tepat sudah diterapkan, apakah kontrol dapat diaudit, dan respons apa yang sesuai jika ternyata ternyata data mendapatkan salah penanganan.

Untuk membantu mengkategorikan di mana data sensitif hadir dalam beban kerja Anda, pertimbangkan untuk menggunakan tag sumber daya, jika tersedia. Misalnya, Anda dapat menerapkan tag yang memiliki kunci tag `Classification` dan nilai tag `PHI` untuk informasi kesehatan yang dilindungi (PHI), dan tag lain yang memiliki kunci tag `Sensitivity` dan nilai tag`High`. Layanan-layanan seperti AWS Config, kemudian dapat digunakan untuk memantau sumber daya ini untuk mendeteksi adanya perubahan dan peringatan jika sumber daya tersebut dimodifikasi dengan cara yang membuatnya melanggar kepatuhan terhadap persyaratan perlindungan Anda (seperti mengubah pengaturan enkripsi). Anda dapat merekam definisi standar kunci tag dan nilai yang dapat diterima menggunakan kebijakan tag, fitur dari AWS Organizations. Sebaiknya kunci atau nilai tag tidak berisi data privat atau sensitif.

Langkah-langkah implementasi

1. Pahami skema klasifikasi dan persyaratan-persyaratan perlindungan data organisasi Anda.
2. Identifikasi jenis data sensitif yang diproses oleh beban kerja Anda.
3. Pastikan bahwa data sensitif disimpan dan dilindungi dalam beban kerja Anda sesuai dengan kebijakan Anda. Gunakan teknik-teknik, seperti pengujian otomatis, untuk mengaudit efektivitas kontrol yang Anda terapkan.
4. Pertimbangkan untuk menggunakan pemberian tag tingkat sumber daya dan data, jika tersedia, untuk memberikan tag pada data dengan tingkat sensitivitasnya dan metadata operasional lainnya yang dapat membantu Anda dalam melakukan pemantauan dan merespons insiden.
   a. AWS Organizations kebijakan tag dapat digunakan untuk menegakkan standar penandaan.

Sumber daya

Praktik-praktik terbaik terkait:

- SUS04-BP01 Menerapkan kebijakan klasifikasi data

Dokumen terkait:

- Laporan Resmi Klasifikasi Data
- Praktik Terbaik untuk Menandai Sumber Daya AWS

Contoh terkait:

- AWS Organizations Tag Policy Sintaks dan Contoh

Alat terkait

- Editor Tag AWS

SEC07-BP02 Menerapkan kontrol perlindungan data berdasarkan sensitivitas data

Terapkan kontrol-kontrol perlindungan data yang memberikan tingkat kontrol yang sesuai untuk setiap kelas data yang ditentukan dalam kebijakan klasifikasi Anda. Praktik ini dapat memungkinkan Anda untuk melindungi data sensitif dari akses dan penggunaan yang tidak sah, sekaligus menjaga ketersediaan dan penggunaan data.

Hasil yang diinginkan: Anda memiliki kebijakan klasifikasi yang mendefinisikan berbagai tingkat sensitivitas untuk data yang ada dalam organisasi Anda. Untuk masing-masing tingkat sensitivitas ini, Anda memiliki pedoman yang jelas yang dipublikasikan untuk layanan dan lokasi penyimpanan dan penanganan yang disetujui, serta konfigurasi yang diperlukan. Anda mengimplementasikan kontrol untuk masing-masing tingkat sesuai dengan tingkat perlindungan yang diperlukan dan biaya yang terkait. Anda memiliki pemantauan dan peringatan untuk mendeteksi apakah data ada di lokasi yang tidak sah, diproses di lingkungan yang tidak sah, diakses oleh pelaku yang tidak sah, atau konfigurasi layanan terkait tidak lagi mematuhi persyaratan yang ditetapkan.

Anti-pola umum:

- Menerapkan tingkat kontrol perlindungan yang sama terhadap semua data. Hal ini dapat menyebabkan penyediaan kontrol keamanan yang berlebihan untuk data yang tidak sensitif, atau perlindungan yang tidak memadai untuk data yang sangat sensitif.
- Tidak melibatkan para pemangku kepentingan yang relevan dari tim keamanan, kepatuhan, dan bisnis saat menentukan kontrol perlindungan data.
- Mengabaikan biaya overhead operasional dan biaya yang terkait dengan implementasi dan pemeliharaan kontrol perlindungan data.

- Tidak melakukan peninjauan kontrol perlindungan data secara berkala untuk menjaga keselarasan dengan kebijakan klasifikasi.

Manfaat menerapkan praktik terbaik ini: Dengan menyelaraskan kontrol Anda dengan tingkat klasifikasi data Anda, organisasi Anda akan dapat berinvestasi dalam tingkat kontrol yang lebih tinggi jika diperlukan. Hal ini dapat mencakup penambahan sumber daya untuk pengamanan, pemantauan, pengukuran, remediasi, dan pelaporan.  Jika kontrol yang diperlukan lebih sedikit, maka Anda dapat meningkatkan aksesibilitas dan kelengkapan data untuk tenaga kerja, pelanggan, atau konstituen Anda.  Pendekatan ini akan memberikan organisasi Anda fleksibilitas penggunaan data yang paling besar, sekaligus tetap mematuhi persyaratan-persyaratan perlindungan data.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Mengimplementasikan kontrol-kontrol perlindungan data berdasarkan tingkat sensitivitas data mencakup beberapa langkah penting. Pertama, mengidentifikasi tingkat sensitivitas data yang berbeda-beda dalam arsitektur beban kerja Anda (seperti publik, internal, rahasia, dan terbatas) dan kemudian mengevaluasi di mana Anda menyimpan dan memproses data ini. Selanjutnya, tentukan batasan-batasan isolasi di sekitar data berdasarkan tingkat sensitivitasnya. Kami menyarankan Anda memisahkan data menjadi berbeda Akun AWS, menggunakan kebijakan kontrol layanan (SCPs) untuk membatasi layanan dan tindakan yang diizinkan untuk setiap tingkat sensitivitas data. Dengan cara ini, Anda dapat membuat batasan-batasan isolasi yang kuat dan memberlakukan prinsip hak akses paling rendah.

Setelah Anda menentukan batasan-batasan isolasi tersebut, implementasikan kontrol-kontrol perlindungan yang sesuai berdasarkan tingkat sensitivitas data. Lihat praktik terbaik untuk Melindungi data diam dan Melindungi data bergerak untuk menerapkan kontrol yang relevan seperti enkripsi, kontrol akses, dan audit. Pertimbangkan teknik-teknik seperti tokenisasi atau anonimisasi untuk mengurangi tingkat sensitivitas data Anda. Sederhanakan penerapan kebijakan data yang konsisten di seluruh bisnis Anda dengan sistem tersentralisasi untuk tokenisasi dan detokenisasi.

Lakukan pemantauan dan pengujian secara terus-menerus terhadap efektivitas kontrol yang diimplementasikan. Lakukan peninjauan dan pembaruan secara rutin terhadap skema klasifikasi data, penilaian risiko, dan kontrol perlindungan karena lanskap data dan ancaman bagi organisasi Anda terus berubah. Selaraskan kontrol-kontrol perlindungan data yang diimplementasikan dengan peraturan industri, standar, dan persyaratan hukum yang relevan. Selanjutnya, berikan pengetahuan

dan pelatihan keamanan untuk membantu para karyawan memahami skema klasifikasi data serta tanggung jawab mereka dalam menangani dan melindungi data sensitif.

Langkah-langkah implementasi

1. Identifikasi tingkat klasifikasi dan sensitivitas data yang ada dalam beban kerja Anda.
2. Tentukan batasan-batasan isolasi untuk setiap tingkat dan tentukan strategi penegakannya.
3. Lakukan evaluasi terhadap kontrol-kontrol yang Anda tetapkan yang mengatur akses, enkripsi, audit, retensi, dan lainnya yang diwajibkan berdasarkan kebijakan klasifikasi data Anda.
4. Lakukan evaluasi terhadap opsi-opsi untuk mengurangi tingkat sensitivitas data jika sesuai, seperti menggunakan tokenisasi atau anonimisasi.
5. Pastikan bahwa kontrol Anda menggunakan pengujian dan pemantauan otomatis terhadap sumber daya yang dikonfigurasi.

Sumber daya

Praktik-praktik terbaik terkait:

- PERF03-BP01 Gunakan penyimpanan data yang dibuat khusus yang paling mendukung persyaratan akses dan penyimpanan data Anda
- COST04-BP05 Menegakkan kebijakan penyimpanan data

Dokumen terkait:

- Laporan Resmi Klasifikasi Data
- Praktik Terbaik untuk Keamanan, Identitas & Kepatuhan
- AWS KMS Praktik Terbaik
- Praktik dan fitur terbaik enkripsi untuk AWS layanan

Contoh terkait:

- Membangun solusi tokenisasi nirserver untuk melakukan masking terhadap data sensitif
- Cara menggunakan tokenisasi untuk meningkatkan keamanan data dan mengurangi cakupan audit

Alat terkait:

- AWS Key Management Service (AWS KMS)
- AWS CloudHSM
- AWS Organizations

SEC07-BP03 Mengotomatiskan identifikasi dan klasifikasi

Otomatisasi identifikasi dan klasifikasi data dapat membantu Anda mengimplementasikan kontrol yang tepat. Penggunaan otomatisasi untuk melengkapi proses penentuan manual akan mengurangi risiko terjadinya kesalahan manusia dan paparan.

Hasil yang diinginkan: Anda dapat melakukan verifikasi apakah kontrol yang tepat sudah dilakukan berdasarkan klasifikasi dan kebijakan penanganan Anda. Alat-alat dan layanan otomatis dapat membantu Anda mengidentifikasi dan mengklasifikasikan tingkat sensitivitas data Anda. Otomatisasi juga akan membantu Anda untuk terus memantau lingkungan Anda guna mendeteksi dan memperingatkan jika data sedang disimpan atau sedang ditangani secara tidak sah sehingga Anda bisa melakukan tindakan korektif dengan cepat.

Anti-pola umum:

- Hanya mengandalkan proses-proses manual untuk melakukan identifikasi dan klasifikasi data, yang bisa jadi rawan kesalahan dan memakan waktu. Hal ini dapat menyebabkan klasifikasi data yang tidak efisien dan tidak konsisten, terutama saat volume data semakin besar.
- Tidak memiliki mekanisme untuk melacak dan mengelola aset data yang ada di seluruh organisasi.
- Mengabaikan perlunya pemantauan dan klasifikasi data yang berkelanjutan seiring pergerakan dan perkembangan data di dalam organisasi.

Manfaat menerapkan praktik terbaik ini: Melakukan otomatisasi atas identifikasi dan klasifikasi data dapat mengantarkan Anda pada penerapan kontrol perlindungan data yang lebih konsisten dan akurat, mengurangi risiko terjadinya kesalahan manusia. Otomatisasi juga dapat memberikan visibilitas terhadap akses dan pergerakan data sensitif, sehingga akan membantu Anda untuk mendeteksi penanganan yang tidak sah dan mengambil tindakan korektif.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Meskipun penilaian manusia sering kali digunakan untuk mengklasifikasikan data selama fase desain awal beban kerja, Anda harus mempertimbangkan untuk memiliki sistem yang mengotomatiskan

identifikasi dan klasifikasi terhadap data uji sebagai sebuah kontrol preventif. Misalnya, pengembang dapat diberi sebuah alat atau layanan untuk memindai data representatif guna menentukan sensitivitasnya. Di dalamnya AWS, Anda dapat mengunggah kumpulan data ke Amazon S3 dan memindainya menggunakan Amazon Macie, Amazon Comprehend, atau Amazon Comprehend Medical. Selain itu, pertimbangkan juga untuk melakukan pemindaian data sebagai bagian dari pengujian unit dan integrasi untuk mendeteksi di mana data sensitif tidak diharapkan. Mengirim peringatan terkait data sensitif pada tahap ini dapat menyoroti adanya kesenjangan dalam perlindungan sebelum dilakukan deployment ke produksi. Fitur lain seperti deteksi data sensitif di AWS Glue, Amazon SNS, dan Amazon juga CloudWatch dapat digunakan untuk mendeteksi PII dan mengambil tindakan mitigasi. Untuk alat atau layanan otomatis apa pun, pahami cara alat atau layanan tersebut menentukan data sensitif, kemudian lengkapi dengan solusi manusia atau solusi otomatis lainnya untuk mengatasi kesenjangan apa pun sesuai kebutuhan.

Sebagai sebuah kontrol pendeteksi, gunakan pemantauan berkelanjutan terhadap lingkungan Anda untuk mendeteksi apakah data sensitif saat ini disimpan dengan cara yang tidak mematuhi persyaratan, atau tidak. Hal ini dapat membantu Anda mendeteksi berbagai kesulitan, seperti data sensitif yang dikirimkan ke file log atau disalin ke lingkungan analitik data tanpa penghapusan atau penyamaran identitas yang tepat. Data yang disimpan di Amazon S3 dapat terus dipantau untuk menemukan data sensitif menggunakan Amazon Macie.

Langkah-langkah implementasi

1. Lakukan pemindaian awal terhadap lingkungan Anda untuk identifikasi dan klasifikasi otomatis.
    a. Pemindaian penuh di awal terhadap data Anda dapat membantu menghasilkan pemahaman komprehensif tentang di mana data sensitif berada di lingkungan Anda. Jika pemindaian penuh pada awalnya tidak diperlukan atau tidak dapat diselesaikan di awal karena biaya, evaluasi apakah teknik-teknik pengambilan sampel data sudah cocok untuk meraih hasi-hasil yang Anda tetapkan. Misalnya, Amazon Macie dapat dikonfigurasi untuk melakukan operasi penemuan data sensitif otomatis secara meluas di seluruh bucket S3 Anda. Kemampuan ini menggunakan teknik-teknik pengambilan sampel untuk melakukan analisis awal terkait di mana data sensitif berada dengan cara yang hemat. Analisis bucket S3 yang lebih mendalam kemudian dapat dilakukan dengan menggunakan pekerjaan penemuan data sensitif. Penyimpanan data lainnya juga dapat diekspor ke S3 untuk dipindai oleh Macie.
2. Konfigurasikan pemindaian yang berkelanjutan terhadap lingkungan Anda.
    a. Kemampuan penemuan data sensitif otomatis yang dimiliki Macie dapat digunakan untuk melakukan pemindaian yang berkelanjutan terhadap lingkungan Anda. Bucket S3 yang

diketahui yang diotorisasi untuk menyimpan data sensitif dapat dikecualikan menggunakan daftar yang diizinkan di Macie.

3. Terapkan identifikasi dan klasifikasi ke dalam proses build dan pengujian Anda.
   a. Identifikasi alat-alat yang dapat digunakan pengembang untuk memindai data guna menentukan sensitivitasnya saat beban kerja sedang dikembangkan. Gunakan alat-alat ini sebagai bagian dari pengujian integrasi untuk memberikan peringatan ketika ada data sensitif yang tidak terduga dan mencegah deployment lebih lanjut.
4. Implementasikan sebuah sistem atau runbook untuk melakukan tindakan ketika data sensitif ditemukan di lokasi yang tidak sah.

Sumber daya

Dokumen terkait:

- AWS Glue: Mendeteksi dan memproses data sensitif
- Menggunakan pengidentifikasi data terkelola di Amazon SNS
- Amazon CloudWatch Logs: Membantu melindungi data log sensitif dengan masking

Contoh terkait:

- Mengaktifkan klasifikasi data untuk RDS database Amazon dengan Macie
- Mendeteksi data sensitif di DynamoDB menggunakan Macie

Alat terkait:

- Amazon Macie
- Amazon Comprehend
- Amazon Comprehend Medical
- AWS Glue

SEC07-BP04 Tentukan manajemen siklus hidup data yang dapat diskalakan

Pahami persyaratan-persyaratan siklus hidup data Anda karena persyaratan tersebut terkait dengan berbagai tingkat klasifikasi dan penanganan data Anda. Hal ini dapat mencakup cara data ditangani ketika pertama kali memasuki lingkungan Anda, cara data ditransformasi, dan aturan untuk

pemusnahannya. Pertimbangkan faktor-faktor seperti periode retensi, akses, audit, dan pelacakan asal.

Hasil yang diinginkan: Anda mengklasifikasikan data sedekat mungkin dengan titik dan waktu konsumsi. Ketika klasifikasi data memerlukan proses masking, tokenisasi, atau proses-proses lain yang mengurangi tingkat sensitivitas, Anda harus melakukan tindakan-tindakan ini sedekat mungkin dengan titik dan waktu penyerapan.

Anda menghapus data sesuai dengan kebijakan Anda ketika data tersebut tidak lagi layak untuk dipertahankan, berdasarkan klasifikasinya.

Anti-pola umum:

- Menerapkan one-size-fits-all pendekatan manajemen siklus hidup data, tanpa mempertimbangkan berbagai tingkat sensitivitas dan persyaratan akses.
- Mempertimbangkan manajemen siklus hidup hanya dari perspektif data yang dapat digunakan, atau data yang dicadangkan, tetapi tidak keduanya.
- Menganggap data yang telah memasuki beban kerja Anda sebagai data yang valid, tanpa mengetahui nilai atau asal-usulnya.
- Mengandalkan durabilitas data sebagai pengganti untuk pencadangan dan perlindungan data.
- Mempertahankan data melampaui masa kegunaannya dan periode retensi yang diperlukan.

Manfaat menerapkan praktik terbaik ini: Strategi manajemen siklus hidup data yang ditentukan dengan baik dan dapat diskalakan akan membantu Anda dalam menjaga kepatuhan terhadap peraturan, meningkatkan keamanan data, mengoptimalkan biaya penyimpanan, dan memungkinkan akses dan berbagi data yang efisien sekaligus mempertahankan kontrol yang tepat.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Data dalam beban kerja sering kali bersifat dinamis.  Bentuk data saat memasuki lingkungan beban kerja Anda dapat berbeda-beda, dari ketika data disimpan atau digunakan dalam logika bisnis, pelaporan, analitik, atau machine learning.  Selain itu, nilai data dapat berubah seiring waktu. Beberapa data bersifat temporal dan kehilangan nilai seiring umurnya bertambah.  Pertimbangkan dampak dari berbagai perubahan data Anda ini terhadap evaluasi berdasarkan skema klasifikasi data Anda dan kontrol-kontrol terkait.  Jika memungkinkan, gunakan mekanisme siklus hidup

otomatis, seperti kebijakan siklus hidup Amazon S3 dan Amazon Data Lifecycle Manager, untuk mengonfigurasi proses retensi data, pengarsipan, dan kedaluwarsa data Anda.

Bedakan antara data yang tersedia untuk digunakan, dan data yang disimpan sebagai cadangan. Pertimbangkan AWS Backupuntuk menggunakan untuk mengotomatiskan cadangan data di seluruh AWS layanan. EBSSnapshot Amazon menyediakan cara untuk menyalin EBS volume dan menyimpannya menggunakan fitur S3, termasuk siklus hidup, perlindungan data, dan akses ke mekanisme perlindungan. Dua dari mekanisme ini adalah Kunci Objek S3 dan AWS Backup Kunci Vault, yang dapat memberi Anda keamanan dan kontrol tambahan atas cadangan Anda. Kelola pemisahan tugas dan akses yang jelas untuk cadangan. Isolasi cadangan di tingkat akun agar tetap terpisah dari lingkungan yang terpengaruh saat ada suatu peristiwa yang terjadi.

Aspek lain dari manajemen siklus hidup adalah merekam riwayat data saat berlangsung melalui beban kerja Anda, yang disebut pelacakan asal data. Pelacakan ini dapat memberikan keyakinan bahwa Anda tahu dari mana data berasal, setiap transformasi yang dilakukan, pemilik atau proses apa yang membuat perubahan tersebut, dan kapan. Riwayat ini dapat membantu Anda dalam melakukan pemecahan masalah dan investigasi selama peristiwa keamanan yang mungkin terjadi. Misalnya, Anda dapat mencatat log metadata tentang transformasi dalam sebuah tabel Amazon DynamoDB. Dalam sebuah danau data, Anda dapat menyimpan salinan data yang ditransformasi di dalam bucket S3 yang berbeda untuk setiap tahap pipeline data. Simpan informasi skema dan stempel waktu dalam file AWS Glue Data Catalog. Terlepas dari solusi yang Anda gunakan, pertimbangkan kebutuhan pengguna akhir Anda untuk menentukan peralatan yang tepat yang Anda butuhkan untuk melaporkan asal-usul data Anda. Hal ini akan membantu Anda menentukan cara terbaik dalam melacak asal-usul data Anda.

Langkah-langkah implementasi

1. Analisis jenis data, tingkat sensitivitas, dan persyaratan akses beban kerja untuk mengklasifikasikan data dan menentukan strategi-strategi manajemen siklus hidup yang sesuai.
2. Rancang dan implementasikan kebijakan retensi data dan proses pemusnahan otomatis yang selaras dengan persyaratan-persyaratan berdasarkan hukum, peraturan, dan organisasi.
3. Tetapkan proses dan otomatisasi untuk melakukan pemantauan, audit, dan penyesuaian berkelanjutan terhadap strategi, kontrol, dan kebijakan manajemen siklus hidup data seiring dengan perubahan persyaratan beban kerja dan perubahan peraturan.

Sumber daya

Praktik-praktik terbaik terkait:

- COST04-BP05 Menegakkan kebijakan penyimpanan data
- SUS04-BP03 Gunakan kebijakan untuk mengelola siklus hidup kumpulan data Anda

Dokumen terkait:

- Laporan Resmi Klasifikasi Data
- Cetak Biru AWS untuk Pertahanan Ransomware
- DevOpsPanduan: Meningkatkan ketertelusuran dengan pelacakan asal data

Contoh terkait:

- Bagaimana melindungi data sensitif untuk seluruh siklus hidupnya di AWS
- Membangun garis keturunan data untuk data lake menggunakan AWS Glue, Amazon Neptunus, dan Spline

Alat terkait:

- AWS Backup
- Amazon Data Lifecycle Manager
- AWS Identity and Access Management Access Analyzer

## SEC8. Bagaimana cara melindungi data diam Anda?

Lindungi data diam dengan mengimplementasikan beberapa kontrol, untuk mengurangi risiko akses tanpa otorisasi atau penanganan yang salah.

Praktik terbaik

- SEC08-BP01 Menerapkan manajemen kunci yang aman
- SEC08-BP02 Menerapkan enkripsi saat istirahat
- SEC08-BP03 Mengotomatiskan data saat perlindungan istirahat
- SEC08-BP04 Menegakkan kontrol akses

### SEC08-BP01 Menerapkan manajemen kunci yang aman

Manajemen kunci yang aman mencakup penyimpanan, rotasi, kontrol akses, dan pemantauan materi kunci yang diperlukan untuk mengamankan data diam untuk beban kerja Anda.

Hasil yang diinginkan: Mekanisme manajemen kunci yang dapat diskalakan, dapat diulang, dan dapat diotomatiskan. Mekanisme ini harus memberikan kemampuan untuk menerapkan hak akses paling rendah ke materi kunci dan memberikan keseimbangan yang tepat antara ketersediaan, kerahasiaan, dan integritas kunci. Akses ke kunci harus dipantau, dan materi kunci dirotasi melalui sebuah proses otomatis. Materi kunci tidak boleh diakses oleh identitas manusia.

Anti-pola umum:

- Akses manusia ke materi kunci yang tidak dienkripsi.
- Membuat algoritma kriptografi kustom.
- Izin yang terlalu luas untuk mengakses materi kunci.

Manfaat menerapkan praktik terbaik ini: Dengan membuat mekanisme manajemen kunci yang aman untuk beban kerja Anda, Anda dapat membantu memberikan perlindungan untuk konten Anda dari akses yang tidak sah. Selain itu, Anda mungkin harus mematuhi persyaratan-persyaratan berdasarkan peraturan untuk mengenkripsi data Anda. Solusi manajemen kunci yang efektif dapat memberikan mekanisme-mekanisme teknis yang selaras dengan peraturan tersebut untuk melindungi materi kunci.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Banyak persyaratan-persyaratan berdasarkan peraturan dan praktik terbaik yang menyertakan enkripsi data diam sebagai kontrol keamanan mendasar. Untuk mematuhi kontrol ini, beban kerja Anda memerlukan sebuah mekanisme untuk menyimpan dan mengelola materi kunci yang digunakan untuk mengenkripsi data diam Anda.

AWS menawarkan AWS Key Management Service (AWS KMS) untuk menyediakan penyimpanan kunci yang tahan lama, aman, dan berlebihan. AWS KMS Banyak AWS layanan terintegrasi dengan AWS KMS untuk mendukung enkripsi data Anda. AWS KMS menggunakan FIPS 140-2 Level 3 modul keamanan perangkat keras yang divalidasi untuk melindungi kunci Anda. Tidak ada mekanisme untuk mengekspor AWS KMS kunci dalam teks biasa.

Saat menerapkan beban kerja menggunakan strategi multi-akun, dianggap praktik terbaik untuk menyimpan AWS KMS kunci di akun yang sama dengan beban kerja yang menggunakannya. Dalam model terdistribusi ini, tanggung jawab untuk mengelola AWS KMS kunci berada pada tim aplikasi. Dalam kasus penggunaan lain, organisasi dapat memilih untuk menyimpan AWS KMS kunci ke dalam akun terpusat. Struktur tersentralisasi ini memerlukan kebijakan tambahan untuk mengaktifkan akses lintas akun yang diperlukan agar akun beban kerja dapat mengakses kunci yang disimpan di akun tersentralisasi tersebut, tetapi mungkin lebih ideal untuk kasus penggunaan di mana satu kunci digunakan bersama-sama di beberapa Akun AWS.

Terlepas dari di mana bahan kunci disimpan, akses ke kunci harus dikontrol dengan ketat melalui penggunaan kebijakan dan IAM kebijakan utama. Kebijakan kunci adalah cara utama untuk mengontrol akses ke AWS KMS kunci. Selain itu, hibah AWS KMS kunci dapat menyediakan akses ke AWS layanan untuk mengenkripsi dan mendekripsi data atas nama Anda. Luangkan waktu untuk meninjau praktik terbaik untuk kontrol akses ke AWS KMS kunci Anda.

Salah satu praktik terbaik yang bisa diterapkan adalah memantau penggunaan kunci enkripsi untuk mendeteksi pola-pola akses yang tidak biasa. Operasi yang dilakukan menggunakan kunci AWS terkelola dan kunci terkelola pelanggan yang disimpan AWS KMS dapat masuk AWS CloudTrail dan harus ditinjau secara berkala. Perhatian khusus harus diberikan pada pemantauan peristiwa-peristiwa penghancuran kunci. Untuk mengurangi penghancuran materi kunci yang tidak disengaja atau berbahaya, peristiwa penghancuran kunci tidak akan langsung menghapus materi kunci tersebut. Upaya untuk menghapus kunci dalam AWS KMS tunduk pada masa tunggu, yang default hingga 30 hari, memberikan waktu administrator untuk meninjau tindakan ini dan mengembalikan permintaan jika perlu.

Sebagian besar AWS layanan digunakan dengan AWS KMS cara yang transparan bagi Anda - satu-satunya persyaratan Anda adalah memutuskan apakah akan menggunakan kunci yang AWS dikelola atau dikelola pelanggan. Jika beban kerja Anda memerlukan penggunaan langsung AWS KMS untuk mengenkripsi atau mendekripsi data, praktik terbaik adalah menggunakan enkripsi amplop untuk melindungi data Anda. AWS Enkripsi SDK dapat menyediakan primitif enkripsi sisi klien aplikasi Anda untuk mengimplementasikan enkripsi amplop dan berintegrasi dengannya. AWS KMS

Langkah-langkah implementasi

1. Tentukan opsi manajemen kunci yang sesuai (AWS dikelola atau dikelola pelanggan) untuk kunci tersebut.

- Untuk kemudahan penggunaan, AWS menawarkan kunci yang AWS dimiliki dan AWS dikelola untuk sebagian besar layanan, yang menyediakan encryption-at-rest kemampuan tanpa perlu mengelola materi utama atau kebijakan utama.
- Saat menggunakan kunci yang dikelola oleh pelanggan, pertimbangkan penyimpanan kunci default untuk memberikan keseimbangan terbaik antara ketangkasan, keamanan, kedaulatan data, dan ketersediaan. Kasus-kasus penggunaan lain mungkin memerlukan penggunaan penyimpanan kunci kustom dengan AWS CloudHSM atau penyimpanan kunci eksternal.

2. Tinjau daftar layanan yang Anda gunakan untuk beban kerja Anda untuk memahami bagaimana AWS KMS terintegrasi dengan layanan. Misalnya, EC2 instance dapat menggunakan volume terenkripsi, memverifikasi bahwa EBS snapshot Amazon yang dibuat dari EBS volume tersebut juga dienkripsi menggunakan kunci yang dikelola pelanggan dan mengurangi pengungkapan data snapshot yang tidak terenkripsi secara tidak disengaja.
   - Bagaimana AWS layanan menggunakan AWS KMS
   - Untuk informasi terperinci tentang opsi enkripsi yang ditawarkan AWS layanan, lihat topik Enkripsi saat Istirahat di panduan pengguna atau panduan pengembang untuk layanan tersebut.
3. Implementasikan AWS KMS: AWS KMS memudahkan Anda untuk membuat dan mengelola kunci dan mengontrol penggunaan enkripsi di berbagai AWS layanan dan aplikasi Anda.
   - Memulai: AWS Key Management Service (AWS KMS)
   - Tinjau praktik terbaik untuk kontrol akses ke AWS KMS kunci Anda.
4. Pertimbangkan AWS Encryption SDK: Gunakan AWS KMS integrasi AWS Encryption SDK with saat aplikasi Anda perlu mengenkripsi sisi klien data.
   - AWS Encryption SDK
5. Aktifkan IAMAccess Analyzer untuk meninjau dan memberi tahu secara otomatis jika ada kebijakan kunci yang terlalu luas AWS KMS .
6. Aktifkan Security Hub agar menerima notifikasi jika ada kebijakan kunci yang salah konfigurasi, kunci yang dijadwalkan untuk dihapus, atau kunci tanpa pengaktifan rotasi otomatis.
7. Tentukan tingkat logging yang sesuai untuk AWS KMS kunci Anda. Karena panggilan ke AWS KMS, termasuk peristiwa hanya-baca, dicatat, CloudTrail log yang terkait dengan AWS KMS dapat menjadi banyak.
   - Beberapa organisasi lebih suka memisahkan aktivitas AWS KMS penebangan menjadi jalur terpisah. Untuk detail selengkapnya, lihat CloudTrail bagian Logging AWS KMS API calls with pada panduan AWS KMS developer.

Sumber daya

Dokumen terkait:

- AWS Key Management Service
- Layanan dan alat kriptografi AWS
- Melindungi Data Amazon S3 Menggunakan Enkripsi
- Enkripsi amplop
- Janji kedaulatan digital
- Menjelaskan operasi kunci AWS KMS , membawa kunci Anda sendiri, penyimpanan kunci kustom, dan portabilitas teks sandi
- AWS Key Management Service rincian kriptografi

Video terkait:

- Bagaimana Enkripsi Bekerja di AWS
- Mengamankan Penyimpanan Blok Anda AWS
- Perlindungan data AWS : Menggunakan gembok, kunci, tanda tangan, dan sertifikat

Contoh terkait:

- Menerapkan mekanisme kontrol akses lanjutan menggunakan AWS KMS

SEC08-BP02 Menerapkan enkripsi saat istirahat

Anda harus menerapkan penggunaan enkripsi untuk data diam. Enkripsi menjaga kerahasiaan data sensitif jika terjadi akses tidak sah atau pengungkapan yang tidak disengaja.

Hasil yang diinginkan: Data pribadi harus dienkripsi secara default saat diam. Enkripsi membantu menjaga kerahasiaan data dan memberikan lapisan perlindungan tambahan terhadap pengungkapan atau eksfiltrasi data yang disengaja atau pun yang tidak disengaja. Data yang dienkripsi tidak dapat dibaca atau diakses jika Anda tidak membuka enkripsi datanya terlebih dahulu. Setiap data tersimpan yang tidak dienkripsi harus diinventarisasi dan dikontrol.

Anti-pola umum:

- Tidak menggunakan encrypt-by-default konfigurasi.

- Memberikan akses yang terlalu permisif ke kunci dekripsi.
- Tidak memantau penggunaan kunci enkripsi dan dekripsi.
- Menyimpan data tidak terenkripsi.
- Menggunakan kunci enkripsi yang sama untuk semua data tanpa memperhatikan penggunaan, jenis, dan klasifikasi data.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Petakan kunci enkripsi ke klasifikasi data di dalam beban kerja Anda. Pendekatan ini membantu melindungi terhadap akses yang terlalu permisif saat menggunakan kunci enkripsi tunggal atau sangat kecil untuk data Anda (lihat SEC07-BP01 Memahami skema klasifikasi data Anda).

AWS Key Management Service (AWS KMS) terintegrasi dengan banyak AWS layanan untuk memudahkan mengenkripsi data Anda saat istirahat. Misalnya, di Amazon Simple Storage Service (Amazon S3), Anda dapat mengatur enkripsi default pada sebuah bucket agar semua objek baru dienkripsi secara otomatis. Saat menggunakan AWS KMS, pertimbangkan seberapa ketat data perlu dibatasi. AWS KMS Kunci default dan dikontrol layanan dikelola dan digunakan atas nama Anda oleh. AWS Untuk data sensitif yang memerlukan akses halus ke kunci enkripsi yang mendasarinya, pertimbangkan kunci terkelola pelanggan (). CMKs Anda memiliki kendali penuh atasCMKs, termasuk rotasi dan manajemen akses melalui penggunaan kebijakan utama.

Selain itu, Amazon Elastic Compute Cloud (AmazonEC2) dan Amazon S3 mendukung penegakan enkripsi dengan menyetel enkripsi default. Anda dapat menggunakan Aturan AWS Configuntuk memeriksa secara otomatis apakah Anda menggunakan enkripsi, misalnya, untuk volume Amazon Elastic Block Store (AmazonEBS), instans Amazon Relational Database Service (RDSAmazon), dan bucket Amazon S3.

AWS juga menyediakan opsi untuk enkripsi sisi klien, memungkinkan Anda mengenkripsi data sebelum mengunggahnya ke cloud. AWS Encryption SDK Ini menyediakan cara untuk mengenkripsi data Anda menggunakan enkripsi amplop. Anda menyediakan kunci pembungkus, dan AWS Encryption SDK menghasilkan kunci data unik untuk setiap objek data yang dienkripsi. Pertimbangkan AWS CloudHSM apakah Anda memerlukan modul keamanan perangkat keras penyewa tunggal terkelola ()HSM. AWS CloudHSM memungkinkan Anda untuk menghasilkan, mengimpor, dan mengelola kunci kriptografi pada FIPS 140-2 level 3 yang divalidasi. HSM Beberapa kasus penggunaan AWS CloudHSM termasuk melindungi kunci pribadi untuk mengeluarkan otoritas sertifikat (CA), dan mengaktifkan enkripsi data transparan (TDE) untuk database Oracle. AWS

CloudHSM Klien SDK menyediakan perangkat lunak yang memungkinkan Anda mengenkripsi sisi klien data menggunakan kunci yang disimpan di dalam AWS CloudHSM sebelum mengunggah data Anda ke dalam. AWS Amazon DynamoDB Encryption Client juga memungkinkan Anda untuk melakukan enkripsi dan penandatanganan terhadap item sebelum diunggah ke tabel DynamoDB.

Langkah-langkah implementasi

- Terapkan enkripsi data diam untuk Amazon S3: Implementasikan enkripsi default bucket Amazon S3.

  Konfigurasikan enkripsi default untuk EBS volume Amazon baru: Tentukan bahwa Anda ingin semua EBS volume Amazon yang baru dibuat dibuat dalam bentuk terenkripsi, dengan opsi untuk menggunakan kunci default yang disediakan oleh AWS atau kunci yang Anda buat.

  Konfigurasikan Gambar Mesin Amazon yang dienkripsi (AMIs): Menyalin yang sudah ada AMI dengan enkripsi yang dikonfigurasi akan secara otomatis mengenkripsi volume root dan snapshot.

  Konfigurasikan RDSenkripsi Amazon: Konfigurasikan enkripsi untuk kluster RDS database Amazon Anda dan snapshot saat istirahat dengan menggunakan opsi enkripsi.

  Buat dan konfigurasikan AWS KMS kunci dengan kebijakan yang membatasi akses ke prinsipal yang sesuai untuk setiap klasifikasi data: Misalnya, buat satu AWS KMS kunci untuk mengenkripsi data produksi dan kunci lain untuk mengenkripsi data pengembangan atau pengujian. Anda juga dapat memberikan akses kunci ke yang lain Akun AWS. Pertimbangkan untuk memiliki akun yang berbeda untuk lingkungan-lingkungan pengembangan dan produksi Anda. Jika lingkungan produksi Anda perlu mendekripsi artefak di akun pengembangan, Anda dapat mengedit CMK kebijakan yang digunakan untuk mengenkripsi artefak pengembangan agar akun produksi dapat mendekripsi artefak tersebut. Dan kemudian lingkungan produksi dapat menyerap data yang didekripsi untuk digunakan dalam lingkungan produksi.

  Konfigurasikan enkripsi di AWS layanan tambahan: Untuk AWS layanan lain yang Anda gunakan, tinjau dokumentasi keamanan untuk layanan tersebut guna menentukan opsi enkripsi layanan.

Sumber daya

Dokumen terkait:

- Alat Kripto AWS
- AWS Encryption SDK

- AWS KMS Whitepaper Detail Kriptografi
- AWS Key Management Service
- Layanan dan alat kriptografi AWS
- EBSEnkripsi Amazon
- Enkripsi default untuk EBS volume Amazon
- Mengenkripsi Sumber Daya Amazon RDS
- Bagaimana cara mengaktifkan enkripsi default untuk bucket Amazon S3?
- Melindungi Data Amazon S3 Menggunakan Enkripsi

Video terkait:

- Bagaimana Enkripsi Bekerja di AWS
- Mengamankan Penyimpanan Blok Anda AWS

SEC08-BP03 Mengotomatiskan data saat perlindungan istirahat

Gunakan otomatisasi untuk memvalidasi dan menerapkan kontrol-kontrol data diam. Gunakan pemindaian otomatis untuk mendeteksi kesalahan konfigurasi pada solusi-solusi penyimpanan data Anda, dan lakukan remediasi melalui respons programatis otomatis jika memungkinkan. Terapkan otomatisasi dalam proses CI/CD Anda untuk mendeteksi kesalahan konfigurasi penyimpanan data sebelum di-deploy ke lingkungan produksi.

Hasil yang diinginkan: Sistem otomatis memindai dan memantau lokasi penyimpanan data untuk mencari adanya kesalahan konfigurasi kontrol, akses tidak sah, dan penggunaan yang tidak terduga. Deteksi lokasi penyimpanan yang salah konfigurasi akan memulai remediasi otomatis. Proses-proses otomatis membuat cadangan data dan menyimpan salinan yang tak bisa diubah di luar lingkungan asli.

Anti-pola umum:

- Tidak mempertimbangkan opsi untuk mengaktifkan enkripsi berdasarkan pengaturan-pengaturan default, jika didukung.
- Tidak mempertimbangkan peristiwa keamanan, selain peristiwa operasional, saat merumuskan strategi pencadangan dan pemulihan otomatis.
- Tidak menerapkan pengaturan akses publik untuk layanan-layanan penyimpanan.

- Tidak memantau dan mengaudit kontrol Anda untuk melindungi data diam.

Manfaat menerapkan praktik terbaik ini: Otomatisasi akan membantu Anda dalam mencegah risiko terjadinya kesalahan konfigurasi lokasi penyimpanan data Anda. Hal ini membantu Anda untuk mencegah kesalahan konfigurasi memasuki lingkungan produksi Anda. Praktik terbaik ini juga membantu Anda untuk mendeteksi dan memperbaiki kesalahan konfigurasi, jika terjadi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Otomatisasi adalah tema yang ada di seluruh praktik untuk melindungi data diam Anda. SEC01-BP06 Mengotomatiskan penerapan kontrol keamanan standar menjelaskan bagaimana Anda dapat menangkap konfigurasi sumber daya Anda menggunakan templat infrastruktur sebagai kode (IAc), seperti dengan. AWS CloudFormation  Template ini berkomitmen untuk sistem kontrol versi, dan digunakan untuk menyebarkan sumber daya AWS melalui pipa CI/CD.  Teknik-teknik ini juga berlaku untuk mengotomatiskan konfigurasi solusi penyimpanan data Anda, seperti pengaturan enkripsi di bucket Amazon S3.

Anda dapat memeriksa pengaturan yang Anda tentukan di templat IaC untuk menemukan kesalahan konfigurasi pada pipeline CI/CD Anda menggunakan aturan di AWS CloudFormation Guard.  Anda dapat memantau pengaturan yang belum tersedia di CloudFormation atau perkakas IAC lainnya untuk kesalahan konfigurasi dengan. AWS Config  Peringatan yang dihasilkan Config untuk kesalahan konfigurasi dapat diperbaiki secara otomatis, seperti yang dijelaskan SEC dalam 04-BP04 Memulai remediasi untuk sumber daya yang tidak sesuai.

Penggunaan otomatisasi sebagai bagian dari strategi manajemen izin Anda juga merupakan komponen integral dari perlindungan data otomatis. SEC03-BP02 Berikan akses hak istimewa paling sedikit dan SEC03-BP04 Kurangi izin terus menerus menjelaskan konfigurasi kebijakan akses hak istimewa terkecil yang terus dipantau oleh untuk menghasilkan temuan ketika izin dapat dikurangi. AWS Identity and Access Management Access Analyzer  Selain otomatisasi untuk izin pemantauan, Anda dapat mengonfigurasi Amazon GuardDuty untuk melihat perilaku akses data anomali untuk EBSvolume Anda (melalui EC2 instance), bucket S3, dan basis data Amazon Relational Database Service yang didukung.

Otomatisasi juga berperan dalam mendeteksi ketika ada data sensitif yang disimpan di lokasi yang tidak sah. SEC07-BP03 Identifikasi dan klasifikasi otomatis menjelaskan cara Amazon Macie dapat memantau bucket S3 Anda untuk data sensitif yang tidak terduga dan menghasilkan peringatan yang dapat memulai respons otomatis.

Ikuti praktik di REL09 Cadangkan data untuk mengembangkan strategi pencadangan dan pemulihan data otomatis. Pencadangan dan pemulihan data sama pentingnya untuk pemulihan dari peristiwa-peristiwa keamanan seperti halnya untuk peristiwa operasional.

Langkah-langkah implementasi

1. Tetapkan konfigurasi penyimpanan data dalam templat IaC. Gunakan pemeriksaan otomatis di pipeline CI/CD Anda untuk mendeteksi terjadinya kesalahan konfigurasi.
   a. Anda dapat menggunakan untuk template IAC Anda, dan CloudFormationGuard untuk memeriksa template untuk kesalahan konfigurasi.
   b. Gunakan AWS Config untuk menjalankan aturan-aturan dalam mode evaluasi proaktif. Gunakan pengaturan ini untuk memeriksa kepatuhan sumber daya sebagai sebuah langkah dalam pipeline CI/CD Anda sebelum membuatnya.
2. Pantau sumber daya untuk menemukan kesalahan konfigurasi penyimpanan data.
   a. Setel AWS Config untuk memantau sumber daya penyimpanan data untuk perubahan dalam konfigurasi kontrol dan menghasilkan peringatan untuk menginvokasi tindakan remediasi saat mendeteksi kesalahan konfigurasi.
   b. Lihat SEC04-BP04 Memulai remediasi untuk sumber daya yang tidak sesuai untuk panduan lebih lanjut tentang remediasi otomatis.
3. Pantau dan kurangi izin akses data secara berkelanjutan melalui otomatisasi.
   a. IAMAccess Analyzer dapat berjalan terus menerus untuk menghasilkan peringatan ketika izin berpotensi dikurangi.
4. Pantau dan beri peringatan tentang terjadinya perilaku akses data yang tidak normal.
   a. GuardDutymengawasi tanda tangan ancaman yang diketahui dan penyimpangan dari perilaku akses dasar untuk sumber daya penyimpanan data seperti EBS volume, bucket S3, dan database. RDS
5. Pantau dan beri peringatan tentang adanya data sensitif yang disimpan di lokasi yang tidak diharapkan.
   a. Gunakan Amazon Macie untuk memindai bucket S3 Anda secara terus menerus untuk mencari data sensitif.
6. Otomatiskan pencadangan yang aman dan terenkripsi terhadap data Anda.
   a. AWS Backupadalah layanan terkelola yang membuat cadangan terenkripsi dan aman dari berbagai sumber data di. AWS Elastic Disaster Recovery memungkinkan Anda menyalin beban kerja server penuh dan mempertahankan perlindungan data berkelanjutan dengan tujuan titik pemulihan (RPO) yang diukur dalam hitungan detik. Anda dapat mengonfigurasi kedua

layanan tersebut untuk bekerja bersama dalam mengotomatiskan pembuatan cadangan data dan menyalinnya ke lokasi-lokasi failover. Hal ini dapat membantu menjaga data Anda tetap tersedia saat terkena dampak peristiwa operasional atau keamanan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC01-BP06 Mengotomatiskan penerapan kontrol keamanan standar
- SEC03-BP02 Berikan akses hak istimewa paling sedikit
- SEC03-BP04 Kurangi izin terus menerus
- SEC04-BP04 Memulai remediasi untuk sumber daya yang tidak sesuai
- SEC07-BP03 Mengotomatiskan identifikasi dan klasifikasi
- REL09-BP02 Mengamankan dan mengenkripsi cadangan
- REL09-BP03 Lakukan pencadangan data secara otomatis

Dokumen terkait:

- AWS Panduan Preskriptif: Secara otomatis mengenkripsi volume Amazon yang ada dan baru EBS
- Manajemen Risiko Ransomware tentang AWS Menggunakan Kerangka Keamanan NIST Cyber () CSF

Contoh terkait:

- Cara menggunakan aturan AWS Config proaktif dan AWS CloudFormation Hooks untuk mencegah pembuatan sumber daya cloud yang tidak sesuai
- Mengotomatiskan dan mengelola perlindungan data secara terpusat untuk Amazon S3 dengan AWS Backup
- AWS re:invent 2023 - Menerapkan perlindungan data proaktif menggunakan snapshot Amazon EBS
- AWS re:Invent 2022 - Membangun dan mengotomatiskan ketahanan dengan perlindungan data modern

Alat terkait:

- AWS CloudFormation Guard
- AWS CloudFormation Guard Aturan Registry
- IAMPenganalisis Akses
- Amazon Macie
- AWS Backup
- Pemulihan Bencana Elastis

SEC08-BP04 Menegakkan kontrol akses

Untuk membantu melindungi data diam, terapkan kontrol akses menggunakan mekanisme, seperti isolasi dan penentuan versi, serta terapkan prinsip hak akses paling rendah. Cegah pemberian akses publik ke data Anda.

Hasil yang diinginkan: Verifikasi bahwa hanya pengguna yang berwenang yang dapat mengakses data need-to-know berdasarkan. Melindungi data Anda dengan pencadangan dan penentuan versi rutin untuk mencegah pengubahan atau penghapusan data yang disengaja atau tidak disengaja. Mengisolasi data penting dari data lain untuk melindungi kerahasiaan dan integritas data tersebut.

Anti-pola umum:

- Menyimpan data dengan kebutuhan atau klasifikasi sensitivitas yang berbeda secara bersamaan.
- Menggunakan izin yang terlalu permisif pada kunci dekripsi.
- Salah mengklasifikasi data.
- Tidak menyimpan pencadangan terperinci untuk data penting.
- Memberikan akses terus-menerus ke data produksi.
- Tidak mengaudit akses data atau meninjau izin secara rutin.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Beberapa kontrol dapat membantu melindungi data diam, termasuk akses (menggunakan hak akses paling rendah), isolasi, dan penentuan versi. Akses ke data Anda harus diaudit menggunakan mekanisme detektif, seperti, dan log tingkat layanan AWS CloudTrail, seperti log akses Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3). Anda harus membuat

inventaris tentang data mana yang dapat diakses publik, dan menyusun rencana untuk mengurangi jumlah data yang tersedia untuk publik dari waktu ke waktu.

Kunci Vault Amazon S3 dan Kunci Objek Amazon S3 memberikan kontrol akses wajib untuk objek yang ada di Amazon S3—begitu kebijakan vault dikunci dengan opsi kepatuhan, kebijakan tersebut tidak akan dapat diubah hingga kedaluwarsa, bahkan oleh pengguna root sekalipun.

Langkah-langkah implementasi

- Terapkan akses kontrol: Terapkan akses kontrol dengan hak akses paling rendah, termasuk akses ke kunci enkripsi.
- Pisahkan data berdasarkan tingkat klasifikasi yang berbeda: Gunakan berbagai Akun AWS untuk tingkat klasifikasi data, dan kelola akun-akun tersebut dengan menggunakan AWS Organizations.
- Kebijakan review AWS Key Management Service (AWS KMS): Meninjau tingkat akses yang diberikan dalam AWS KMS kebijakan.
- Tinjau izin objek dan bucket Amazon S3: Tinjau tingkat akses yang diberikan dalam kebijakan bucket S3 secara rutin. Praktik terbaiknya adalah menghindari penggunaan bucket yang dapat dibaca atau ditulis oleh publik. Pertimbangkan AWS Configuntuk menggunakan untuk mendeteksi bucket yang tersedia untuk umum, dan Amazon CloudFront untuk menayangkan konten dari Amazon S3. Pastikan bucket yang seharusnya tidak mengizinkan akses publik telah dikonfigurasi dengan benar untuk mencegah akses publik. Secara default, semua bucket S3 bersifat privat, dan hanya dapat diakses oleh pengguna yang telah diberi akses secara eksplisit.
- Gunakan AWS IAMAccess Analyzer: IAM Access Analyzer menganalisis bucket Amazon S3 dan menghasilkan temuan saat kebijakan S3 memberikan akses ke entitas eksternal.
- Gunakan penentuan versi Amazon S3 dan kunci objek bila perlu.
- Gunakan Inventaris Amazon S3: Inventaris Amazon S3 dapat digunakan untuk mengaudit dan melaporkan replikasi dan status enkripsi objek S3.
- Tinjau Amazon EBS dan izin AMIberbagi: Izin berbagi dapat memungkinkan gambar dan volume dibagikan dengan Akun AWS yang berada di luar beban kerja Anda.
- Tinjau AWS Resource Access Manager Share secara berkala untuk menentukan apakah sumber daya harus terus dibagikan atau tidak. Resource Access Manager memungkinkan Anda berbagi sumber daya, seperti kebijakan AWS Network Firewall, aturan penyelesai Amazon Route 53, dan subnet, di Amazon Anda. VPCs Lakukan audit sumber daya yang dibagikan secara rutin dan hentikan pembagian sumber daya yang sudah tidak perlu dibagikan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC03-BP01 Tentukan persyaratan akses
- SEC03-BP02 Berikan akses hak istimewa paling sedikit

Dokumen terkait:

- AWS KMS Whitepaper Detail Kriptografi
- Pengantar Manajemen Izin Akses ke Sumber Daya Amazon S3 Anda
- Ikhtisar mengelola akses ke AWS KMS sumber daya Anda
- Aturan AWS Config
- Amazon S3 + Amazon CloudFront: Cocokan yang Dibuat di Cloud
- Menggunakan penentuan versi
- Mengunci Objek Menggunakan Kunci Objek Amazon S3
- Berbagi EBS Snapshot Amazon
- Berbagi AMIs
- Meng-host aplikasi satu halaman di Amazon S3

Video terkait:

- Mengamankan Penyimpanan Blok Anda AWS

## SEC9. Bagaimana cara melindungi data bergerak Anda?

Lindungi data bergerak dengan mengimplementasikan beberapa kontrol untuk mengurangi risiko akses tanpa otorisasi atau hilangnya data.

Praktik terbaik

- SEC09-BP01 Menerapkan manajemen kunci dan sertifikat yang aman
- SEC09-BP02 Menerapkan enkripsi dalam perjalanan
- SEC09-BP03 Mengautentikasi komunikasi jaringan

SEC09-BP01 Menerapkan manajemen kunci dan sertifikat yang aman

Sertifikat Transport Layer Security (TLS) digunakan untuk mengamankan komunikasi jaringan dan menetapkan identitas situs web, sumber daya, dan beban kerja melalui internet, serta jaringan pribadi.

Hasil yang diinginkan: Sistem manajemen sertifikat aman yang dapat menyediakan, menyebarkan, menyimpan, dan memperbarui sertifikat dalam infrastruktur kunci publik ()PKI. Mekanisme manajemen kunci dan sertifikat yang aman akan mencegah pengungkapan materi kunci privat sertifikat dan secara otomatis memperpanjang sertifikat secara berkala. Mekanisme ini juga terintegrasi dengan layanan-layanan lain untuk menyediakan komunikasi jaringan yang aman dan identitas untuk sumber daya mesin di dalam beban kerja Anda. Materi kunci tidak boleh diakses oleh identitas manusia.

Anti-pola umum:

- Melakukan langkah-langkah manual selama proses deployment atau perpanjangan sertifikat.
- Kurang memperhatikan hierarki otoritas sertifikat (CA) saat merancang CA privat.
- Menggunakan sertifikat yang ditandatangani sendiri untuk sumber daya publik.

Manfaat menjalankan praktik terbaik ini:

- Sederhanakan manajemen sertifikat melalui deployment dan perpanjangan otomatis
- Mendorong enkripsi data dalam perjalanan menggunakan TLS sertifikat
- Peningkatan keamanan dan keterauditan tindakan-tindakan sertifikat yang dilakukan oleh otoritas sertifikat
- Manajemen tugas-tugas manajemen di berbagai lapisan hierarki CA

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Beban kerja modern memanfaatkan komunikasi jaringan terenkripsi secara ekstensif menggunakan PKI protokol seperti. TLS PKImanajemen sertifikat bisa rumit, tetapi penyediaan sertifikat otomatis, penyebaran, dan pembaruan dapat mengurangi gesekan yang terkait dengan manajemen sertifikat.

AWS menyediakan dua layanan untuk mengelola PKI sertifikat tujuan umum: AWS Certificate Managerdan AWS Private Certificate Authority ()AWS Private CA. ACMadalah layanan utama yang

digunakan pelanggan untuk menyediakan, mengelola, dan menyebarkan sertifikat untuk digunakan baik dalam beban kerja publik maupun pribadi. AWS ACMmenerbitkan sertifikat yang menggunakan AWS Private CA dan terintegrasi dengan banyak layanan AWS terkelola lainnya untuk memberikan TLS sertifikat aman untuk beban kerja.

AWS Private CA memungkinkan Anda untuk membuat otoritas sertifikat root atau bawahan Anda sendiri dan mengeluarkan TLS sertifikat melalui fileAPI. Anda dapat menggunakan sertifikat semacam ini dalam skenario di mana Anda mengontrol dan mengelola rantai kepercayaan di sisi klien TLS koneksi. Selain kasus TLS penggunaan, AWS Private CA dapat digunakan untuk menerbitkan sertifikat ke pod Kubernetes, pengesahan produk perangkat Matter, penandatanganan kode, dan kasus penggunaan lainnya dengan templat khusus. Anda juga dapat menggunakan IAMPeran Di Mana Saja untuk memberikan IAM kredensil sementara ke beban kerja lokal yang telah diterbitkan sertifikat X.509 yang ditandatangani oleh CA Pribadi Anda.

Selain ACM dan AWS Private CA, AWS IoT Coremenyediakan dukungan khusus untuk penyediaan, pengelolaan, dan penyebaran sertifikat PKI ke perangkat IoT. AWS IoT Core menyediakan mekanisme khusus untuk orientasi perangkat IoT ke infrastruktur kunci publik Anda dalam skala besar.

Pertimbangan untuk membangun hierarki CA privat

Ketika Anda perlu membuat CA privat, penting untuk berhati-hati dalam merancang hierarki CA dengan benar di awal. Ini adalah praktik terbaik untuk menerapkan setiap tingkat hierarki CA Anda menjadi terpisah Akun AWS saat membuat hierarki CA pribadi. Langkah yang disengaja ini mengurangi luas permukaan untuk setiap level dalam hierarki CA, membuatnya lebih mudah untuk menemukan anomali dalam data CloudTrail log dan mengurangi ruang lingkup akses atau dampak jika ada akses tidak sah ke salah satu akun. CA root harus berada di akun terpisahnya sendiri dan hanya boleh digunakan untuk menerbitkan satu atau beberapa sertifikat CA perantara.

Kemudian, buat satu atau beberapa akun CAs perantara yang terpisah dari akun root CA untuk mengeluarkan sertifikat bagi pengguna akhir, perangkat, atau beban kerja lainnya. Terakhir, terbitkan sertifikat dari CA root Anda ke perantaraCAs, yang pada gilirannya akan mengeluarkan sertifikat ke pengguna akhir atau perangkat Anda. Untuk informasi selengkapnya tentang perencanaan penerapan CA dan perancangan hierarki CA, termasuk perencanaan ketahanan, replikasi lintas wilayah, berbagi CAs di seluruh organisasi, dan lainnya, lihat Merencanakan penerapan. AWS Private CA

Langkah-langkah implementasi

1. Tentukan AWS layanan relevan yang diperlukan untuk kasus penggunaan Anda:

- Banyak kasus penggunaan dapat memanfaatkan infrastruktur kunci AWS publik yang ada menggunakan AWS Certificate Manager. ACMdapat digunakan untuk menyebarkan TLS sertifikat untuk server web, penyeimbang beban, atau penggunaan lain untuk sertifikat yang dipercaya publik.
- Pertimbangkan AWS Private CA ketika Anda perlu membuat hierarki otoritas sertifikat privat Anda sendiri atau memerlukan akses ke sertifikat yang dapat diekspor. ACMkemudian dapat digunakan untuk mengeluarkan banyak jenis sertifikat entitas akhir menggunakan. AWS Private CA
- Untuk kasus penggunaan di mana sertifikat harus disediakan dalam skala besar untuk perangkat Internet untuk Segala (IoT) yang disematkan, pertimbangkan AWS IoT Core.

2. Implementasikan perpanjangan sertifikat otomatis jika memungkinkan:
   - Gunakan perpanjangan ACM terkelola untuk sertifikat yang dikeluarkan ACM bersama dengan layanan AWS terkelola terintegrasi.
3. Bangun jejak pencatatan log dan jejak audit:
   - Aktifkan CloudTraillog untuk melacak akses ke akun yang memegang otoritas sertifikat. Pertimbangkan untuk mengonfigurasi validasi integritas file log CloudTrail untuk memverifikasi keaslian data log.
   - Buat dan tinjau secara berkala laporan audit yang mencantumkan sertifikat yang telah dikeluarkan atau dicabut oleh CA privat Anda. Laporan-laporan ini dapat diekspor ke bucket S3.
   - Saat menerapkan CA pribadi, Anda juga perlu membuat bucket S3 untuk menyimpan Daftar Pencabutan Sertifikat (). CRL Untuk panduan cara mengonfigurasi bucket S3 ini berdasarkan persyaratan beban kerja Anda, lihat Merencanakan daftar pencabutan sertifikat (). CRL

Sumber daya

Praktik-praktik terbaik terkait:

- SEC02-BP02 Gunakan kredenal sementara
- SEC08-BP01 Menerapkan manajemen kunci yang aman
- SEC09-BP03 Mengautentikasi komunikasi jaringan

Dokumen terkait:

- Cara meng-host dan mengelola seluruh infrastruktur sertifikat pribadi di AWS
- Cara mengamankan hierarki CA ACM pribadi skala perusahaan untuk otomotif dan manufaktur

- Praktik terbaik CA privat
- Cara menggunakan AWS RAM untuk membagikan akun lintas CA ACM Pribadi Anda

Video terkait:

- Mengaktifkan CA Privat AWS Certificate Manager (lokakarya)

Contoh terkait:

- Lokakarya CA privat
- IOTLokakarya Manajemen Perangkat (termasuk penyediaan perangkat)

Alat terkait:

- Plugin ke Kubernetes cert-manager untuk digunakan AWS Private CA

SEC09-BP02 Menerapkan enkripsi dalam perjalanan

Berlakukan persyaratan-persyaratan enkripsi yang Anda tetapkan berdasarkan kebijakan, kewajiban berdasarkan regulasi, dan standar organisasi Anda untuk membantu memenuhi persyaratan organisasi, hukum, dan kepatuhan. Hanya gunakan protokol dengan enkripsi saat mentransmisikan data sensitif di luar cloud pribadi virtual Anda (). VPC Enkripsi akan membantu menjaga kerahasiaan data, bahkan ketika data berada di jaringan yang tidak tepercaya.

Hasil yang diinginkan: Semua data harus dienkripsi dalam perjalanan menggunakan TLS protokol aman dan cipher suite. Lalu lintas jaringan antara sumber daya Anda dan internet harus dienkripsi untuk mengurangi akses tidak sah ke data. Lalu lintas jaringan hanya dalam AWS lingkungan internal Anda harus dienkripsi menggunakan TLS sedapat mungkin. Jaringan AWS internal dienkripsi secara default dan lalu lintas jaringan di dalam VPC tidak dapat dipalsukan atau diendus kecuali pihak yang tidak berwenang telah memperoleh akses ke sumber daya apa pun yang menghasilkan lalu lintas (seperti instans EC2 Amazon, dan wadah Amazon). ECS Pertimbangkan untuk melindungi network-to-network lalu lintas dengan jaringan pribadi IPsec virtual (VPN).

Anti-pola umum:

- Menggunakan versi usang,, dan komponen cipher suite (misalnya SSLTLS, kunci SSL v3.0, RSA 1024-bit, dan cipher). RC4

- Mengizinkan lalu lintas tidak terenkripsi (HTTP) ke atau dari sumber daya yang menghadap publik.
- Tidak memantau dan tidak mengganti sertifikat X.509 sebelum kedaluwarsa.
- Menggunakan sertifikat X.509 yang ditandatangani sendiri untuk. TLS

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS layanan menyediakan HTTPS titik akhir yang digunakan TLS untuk komunikasi, menyediakan enkripsi dalam perjalanan saat berkomunikasi dengan. AWS APIs Protokol yang tidak aman seperti HTTP dapat diaudit dan diblokir di a VPC melalui penggunaan kelompok keamanan. HTTPPermintaan juga dapat secara otomatis dialihkan ke Amazon CloudFront atau HTTPS di Application Load Balancer. Anda memiliki kendali penuh atas sumber daya komputasi Anda untuk mengimplementasikan enkripsi data bergerak di seluruh layanan Anda. Selain itu, Anda dapat menggunakan VPN konektivitas ke dalam jaringan eksternal atau AWS Direct Connectuntuk memfasilitasi enkripsi lalu lintas. VPC Verifikasi bahwa klien Anda melakukan panggilan untuk AWS APIs menggunakan setidaknya TLS 1.2, seperti AWS menghentikan penggunaan versi sebelumnya pada Juni 2023 TLS. AWS merekomendasikan menggunakan TLS 1.3. Solusi pihak ketiga tersedia di AWS Marketplace jika Anda memiliki persyaratan khusus.

Langkah-langkah implementasi

- Terapkan enkripsi data bergerak: Persyaratan enkripsi yang Anda tetapkan harus didasarkan pada standar dan praktik terbaik paling baru dan hanya mengizinkan protokol yang aman. Misalnya, konfigurasikan grup keamanan untuk hanya mengizinkan HTTPS protokol ke penyeimbang beban aplikasi atau EC2 instans Amazon.
- Konfigurasikan protokol aman di layanan edge: Konfigurasikan dengan HTTPS Amazon CloudFront dan gunakan profil keamanan yang sesuai dengan postur keamanan dan kasus penggunaan Anda.
- Gunakan konektivitas VPN untuk eksternal: Pertimbangkan untuk menggunakan IPsec VPN untuk mengamankan point-to-point atau network-to-network koneksi untuk membantu memberikan privasi dan integritas data.
- Konfigurasikan protokol aman di penyeimbang beban: Pilih sebuah kebijakan keamanan yang menyediakan cipher suite terkuat yang didukung oleh klien yang akan terhubung ke pendengar. Buat HTTPS pendengar untuk Application Load Balancer Anda.
- Konfigurasikan protokol aman di Amazon Redshift: Konfigurasikan klaster Anda agar memerlukan koneksi secure socket layer SSL () atau transport layer security TLS ().

- Konfigurasikan protokol aman: Tinjau dokumentasi AWS layanan untuk menentukan encryption-in-transit kemampuan.
- Konfigurasikan akses yang aman saat mengunggah ke bucket Amazon S3: Gunakan kontrol kebijakan bucket Amazon S3 untuk menerapkan akses yang aman ke data.
- Pertimbangkan untuk menggunakan AWS Certificate Manager: ACM memungkinkan Anda untuk menyediakan, mengelola, dan menyebarkan TLS sertifikat publik untuk digunakan dengan AWS layanan.
- Pertimbangkan AWS Private Certificate Authorityuntuk menggunakan untuk PKI kebutuhan pribadi: AWS Private CA memungkinkan Anda membuat hierarki otoritas sertifikat pribadi (CA) untuk mengeluarkan sertifikat X.509 entitas akhir yang dapat digunakan untuk membuat saluran terenkripsi. TLS

Sumber daya

Dokumen terkait:

- Menggunakan HTTPS dengan CloudFront
- Connect Anda VPC ke jaringan jarak jauh menggunakan AWS Virtual Private Network
- Buat HTTPS pendengar untuk Application Load Balancer
- Tutorial: SSL TLS Konfigurasi/di Amazon Linux 2
- MenggunakanSSL/TLSuntuk mengenkripsi koneksi ke instans DB
- Mengonfigurasi opsi-opsi keamanan untuk koneksi

SEC09-BP03 Mengautentikasi komunikasi jaringan

Verifikasi identitas komunikasi dengan menggunakan protokol yang mendukung otentikasi, seperti Transport Layer Security () TLS atau. IPsec

Rancang beban kerja Anda untuk menggunakan protokol jaringan yang aman dan terautentikasi setiap kali berkomunikasi antara layanan, aplikasi, atau ke pengguna. Menggunakan protokol jaringan yang mendukung autentikasi dan otorisasi memberikan kontrol yang lebih kuat atas alur jaringan dan mengurangi dampak akses yang tidak sah.

Hasil yang diinginkan: Beban kerja dengan bidang data yang terdefinisi dengan baik dan arus lalu lintas bidang kontrol antar layanan. Arus lalu lintas menggunakan protokol jaringan yang diautentikasi dan dienkripsi jika memungkinkan secara teknis.

Anti-pola umum:

- Arus lalu lintas yang tidak dienkripsi atau tidak diautentikasi dalam beban kerja Anda.
- Penggunaan kembali kredensial autentikasi oleh beberapa pengguna atau entitas.
- Hanya mengandalkan kontrol jaringan sebagai mekanisme kontrol akses.
- Membuat mekanisme autentikasi kustom, bukan mengandalkan mekanisme autentikasi standar industri.
- Arus lalu lintas yang terlalu permisif antara komponen layanan atau sumber daya lain di. VPC

Manfaat menjalankan praktik terbaik ini:

- Membatasi cakupan dampak untuk akses tidak sah ke satu bagian dari beban kerja.
- Memberikan tingkat jaminan yang lebih tinggi bahwa tindakan hanya dilakukan oleh entitas-entitas yang diautentikasi.
- Meningkatkan pemisahan layanan dengan menentukan dan menerapkan antarmuka transfer data yang diinginkan secara jelas.
- Meningkatkan pemantauan, pembuatan log, dan respons insiden melalui atribusi permintaan dan antarmuka komunikasi yang ditentukan dengan jelas.
- defense-in-depthMenyediakan beban kerja Anda dengan menggabungkan kontrol jaringan dengan kontrol otentikasi dan otorisasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Pola lalu lintas jaringan beban kerja Anda dapat dikelompokkan ke dalam dua kategori:

- Lalu lintas timur-barat mewakili arus lalu lintas yang terjadi antara layanan-layanan yang membentuk sebuah beban kerja.
- Lalu lintas utara-selatan mewakili arus lalu lintas yang terjadi antara beban kerja Anda dan konsumen.

Mengenkripsi lalu lintas utara-selatan adalah praktik yang umum, sedangkan pengamanan lalu lintas timur-barat menggunakan protokol yang diautentikasi merupakan hal yang kurang umum. Praktik keamanan modern menyebutkan bahwa desain jaringan saja tidak cukup untuk memberikan

hubungan yang dapat dipercaya antara dua entitas. Ketika dua layanan dapat berada dalam batasan jaringan yang sama, sebaiknya enkripsi, autentikasi, dan otorisasi komunikasi di antara layanan-layanan tersebut tetap dilakukan, itulah praktik terbaiknya.

Sebagai contoh, AWS layanan APIs menggunakan protokol AWS tanda tangan Signature Version 4 (SigV4) untuk mengautentikasi pemanggil, tidak peduli dari jaringan mana permintaan berasal. Otentikasi ini memastikan bahwa AWS APIs dapat memverifikasi identitas yang meminta tindakan, dan identitas tersebut kemudian dapat digabungkan dengan kebijakan untuk membuat keputusan otorisasi untuk menentukan apakah tindakan tersebut harus diizinkan atau tidak.

Layanan seperti Amazon VPC Lattice dan Amazon API Gateway memungkinkan Anda menggunakan protokol tanda tangan SiGv4 yang sama untuk menambahkan otentikasi dan otorisasi ke lalu lintas timur-barat di beban kerja Anda sendiri. Jika sumber daya di luar AWS lingkungan Anda perlu berkomunikasi dengan layanan yang memerlukan otentikasi dan otorisasi berbasis SIGV4, Anda dapat menggunakan AWS Identity and Access Management (IAM) Roles Anywhere pada AWS non-sumber daya untuk memperoleh kredensi sementara. AWS Kredensial ini dapat digunakan untuk menandatangani permintaan ke layanan dengan menggunakan SigV4 untuk memberi otorisasi akses.

Mekanisme umum lainnya untuk mengautentikasi lalu lintas timur-barat adalah otentikasi TLS timur-barat (m). TLS Banyak Internet of Things (IoT), business-to-business aplikasi, dan layanan mikro menggunakan m TLS untuk memvalidasi identitas kedua sisi TLS komunikasi melalui penggunaan sertifikat X.509 klien dan sisi server. Sertifikat ini dapat diterbitkan oleh AWS Private Certificate Authority (AWS Private CA). Anda dapat menggunakan layanan seperti Amazon API Gateway dan AWS App Meshmenyediakan TLS otentikasi m untuk komunikasi antar atau intra-beban kerja. Sementara m TLS menyediakan informasi otentikasi untuk kedua sisi TLS komunikasi, itu tidak menyediakan mekanisme untuk otorisasi.

Akhirnya, OAuth 2.0 dan OpenID Connect (OIDC) adalah dua protokol yang biasanya digunakan untuk mengendalikan akses ke layanan oleh pengguna, tetapi sekarang menjadi populer untuk service-to-service lalu lintas juga. APIGateway menyediakan otorisasi Token JSON Web (JWT), yang memungkinkan beban kerja membatasi akses ke API rute menggunakan penyedia identitas yang JWTs dikeluarkan dari OIDC atau OAuth 2.0. OAuth2cakupan dapat digunakan sebagai sumber untuk keputusan otorisasi dasar, tetapi pemeriksaan otorisasi masih perlu diimplementasikan di lapisan aplikasi, dan OAuth2 cakupan saja tidak dapat mendukung kebutuhan otorisasi yang lebih kompleks.

Langkah-langkah implementasi

- Tentukan dan dokumentasikan alur jaringan beban kerja Anda: Langkah pertama dalam menerapkan defense-in-depth strategi adalah menentukan arus lalu lintas beban kerja Anda.
  - Buatlah sebuah diagram alur data yang secara jelas menentukan bagaimana data ditransmisikan antara berbagai layanan yang membentuk beban kerja Anda. Diagram ini merupakan langkah pertama untuk menerapkan alur-alur tersebut melalui saluran jaringan yang diautentikasi.
  - Instrumentasikan beban kerja Anda dalam fase pengembangan dan pengujian untuk memvalidasi bahwa diagram alur data mencerminkan perilaku beban kerja secara akurat pada saat runtime.
  - Diagram alir data juga dapat berguna saat melakukan latihan pemodelan ancaman, seperti yang dijelaskan dalam SEC01-BP07 Identifikasi ancaman dan prioritaskan mitigasi menggunakan model ancaman.
- Tetapkan kontrol jaringan: Pertimbangkan AWS kemampuan untuk membuat kontrol jaringan yang selaras dengan aliran data Anda. Meskipun batas jaringan seharusnya tidak menjadi satu-satunya kontrol keamanan, mereka menyediakan lapisan dalam defense-in-depth strategi untuk melindungi beban kerja Anda.
  - Gunakan grup keamanan untuk menetapkan definisi dan membatasi aliran data antar sumber daya.
  - Pertimbangkan AWS PrivateLinkuntuk menggunakan untuk berkomunikasi dengan kedua AWS dan layanan pihak ketiga yang mendukung AWS PrivateLink. Data yang dikirim melalui titik akhir AWS PrivateLink antarmuka tetap berada dalam tulang punggung AWS jaringan dan tidak melintasi Internet publik.
- Menerapkan otentikasi dan otorisasi di seluruh layanan dalam beban kerja Anda: Pilih rangkaian AWS layanan yang paling tepat untuk menyediakan arus lalu lintas terautentikasi dan terenkripsi dalam beban kerja Anda.
  - Pertimbangkan Amazon VPC Lattice untuk mengamankan service-to-service komunikasi. VPCLattice dapat menggunakan otentikasi SiGv4 yang dikombinasikan dengan kebijakan autentikasi untuk mengontrol akses. service-to-service
  - Untuk service-to-service komunikasi menggunakan mTLS, pertimbangkan APIGateway atau App Mesh. AWS Private CAdapat digunakan untuk menetapkan hierarki CA pribadi yang mampu mengeluarkan sertifikat untuk digunakan dengan m. TLS
  - Saat mengintegrasikan dengan layanan menggunakan OAuth 2.0 atauOIDC, pertimbangkan APIGateway menggunakan JWT otorisasi.

  - Untuk komunikasi antara beban kerja Anda dan perangkat IoT, pertimbangkan untuk menggunakan AWS IoT Core, yang menyediakan beberapa opsi untuk enkripsi dan autentikasi lalu lintas jaringan.
- Memantau akses yang tidak sah: Lakukan pemantauan secara terus-menerus terhadap saluran komunikasi yang tidak diinginkan, pengguna utama yang tidak sah yang mencoba mengakses sumber daya yang dilindungi, dan pola akses yang tidak tepat lainnya.
  - Jika menggunakan VPC Lattice untuk mengelola akses ke layanan Anda, pertimbangkan untuk mengaktifkan dan memantau log akses VPCLattice. Log akses ini mencakup informasi tentang entitas yang meminta, informasi jaringan termasuk sumber dan tujuanVPC, dan metadata permintaan.
  - Pertimbangkan untuk mengaktifkan log VPC aliran untuk menangkap metadata pada alur jaringan dan meninjau anomali secara berkala.
  - Lihat Panduan Respons Insiden AWS Keamanan dan bagian Respons Insiden dari pilar keamanan AWS Well-Architected Framework untuk panduan lebih lanjut tentang perencanaan, simulasi, dan penanggulangan insiden keamanan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC03-BP07 Menganalisis akses publik dan lintas akun
- SEC02-BP02 Gunakan kredensil sementara
- SEC01-BP07 Mengidentifikasi ancaman dan memprioritaskan mitigasi menggunakan model ancaman

Dokumen terkait:

- Mengevaluasi metode kontrol akses untuk mengamankan Amazon Gateway API APIs
- Mengkonfigurasi TLS otentikasi timbal balik untuk a REST API
- Cara mengamankan HTTP titik akhir API Gateway dengan otorisasi JWT
- Mengotorisasi panggilan langsung ke AWS layanan menggunakan penyedia AWS IoT Core kredensi
- AWS Panduan Respons Insiden Keamanan

Video terkait:

- AWS re:invent 2022: Memperkenalkan Kisi VPC
- AWS re:invent 2020: Otentikasi tanpa server API untuk aktif HTTP APIs AWS

Contoh terkait:

- Lokakarya Amazon VPC Lattice
- Zero-Trust Episode 1 – Lokakarya Perimeter Layanan Phantom

## Respons insiden

Pertanyaan

- SEC10. Bagaimana cara mengantisipasi, merespons, dan pulih dari insiden?

### SEC10. Bagaimana cara mengantisipasi, merespons, dan pulih dari insiden?

Dengan kontrol detektif dan preventif yang matang sekalipun, organisasi Anda harus mengimplementasikan mekanisme untuk memberikan respons dan melakukan mitigasi atas potensi dampak insiden keamanan. Persiapan Anda sangat berpengaruh pada kemampuan tim Anda untuk beroperasi secara efektif selama insiden, untuk mengisolasi, membatasi, dan melakukan forensik terhadap masalah, serta untuk memulihkan operasi ke kondisi yang baik dan dikenal. Menetapkan alat dan akses sebelum terjadi insiden keamanan, lalu secara rutin melatih respons insiden melalui game day, membantu memastikan bahwa Anda dapat melakukan pemulihan sembari tetap meminimalkan gangguan bisnis.

Praktik terbaik

- SEC10-BP01 Identifikasi personel kunci dan sumber daya eksternal
- SEC10-BP02 Mengembangkan rencana manajemen insiden
- SEC10-BP03 Siapkan kemampuan forensik
- SEC10-BP04 Mengembangkan dan menguji pedoman respons insiden keamanan
- SEC10-BP05 Akses pra-penyediaan
- SEC10-BP06 Alat pra-penerapan
- SEC10-BP07 Jalankan simulasi

- SEC10-BP08 Menetapkan kerangka kerja untuk belajar dari insiden

SEC10-BP01 Identifikasi personel kunci dan sumber daya eksternal

Identifikasikan personel, sumber daya, dan kewajiban hukum internal serta eksternal untuk membantu organisasi Anda merespons insiden.

Hasil yang diinginkan: Anda memiliki daftar personel kunci, informasi kontak mereka, dan peran yang mereka mainkan saat menanggapi peristiwa keamanan yang terjadi. Anda meninjau informasi ini secara rutin dan memperbaruinya untuk menyesuaikan dengan perubahan personel dari perspektif alat internal dan eksternal. Anda mempertimbangkan semua penyedia layanan dan vendor pihak ketiga saat mendokumentasikan informasi ini, termasuk mitra keamanan, penyedia cloud, dan aplikasi ( software-as-a-serviceSaaS). Saat peristiwa keamanan berlangsung, tersedia personel dengan tingkat tanggung jawab, konteks, dan akses yang sesuai untuk melakukan respons dan pemulihan.

Anti-pola umum:

- Tidak memelihara daftar terbaru personel penting dengan informasi kontak, peran mereka, dan tanggung jawab mereka saat merespons peristiwa keamanan.
- Mengasumsikan bahwa setiap orang mengetahui staf yang bertanggung jawab, dependensi, infrastruktur, dan solusi ketika melakukan respons dan pemulihan dari suatu peristiwa.
- Tidak memiliki repositori dokumen atau pengetahuan yang merepresentasikan desain infrastruktur atau aplikasi utama.
- Tidak memiliki proses onboarding yang tepat bagi karyawan baru untuk berkontribusi secara efektif terhadap respons peristiwa keamanan, seperti melakukan simulasi peristiwa.
- Tidak memiliki jalur eskalasi ketika personel penting tidak tersedia untuk sementara waktu atau tidak merespons saat terjadi peristiwa keamanan.

Manfaat menerapkan praktik terbaik ini: Praktik ini akan mengurangi triase dan waktu respons yang dihabiskan untuk mengidentifikasi personel yang tepat dan peran mereka selama suatu peristiwa. Minimalkan waktu yang terbuang saat terjadi sebuah peristiwa dengan memelihara daftar terbaru personel penting dan peran mereka sehingga Anda dapat mendatangkan orang-orang yang tepat untuk melakukan triase dan pemulihan dari sebuah peristiwa.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Identifikasikan personel kunci dalam organisasi Anda: Kelola daftar kontak personel di dalam organisasi Anda yang perlu Anda libatkan. Tinjau dan perbarui informasi ini secara rutin jika terjadi perpindahan personel, seperti perubahan organisasi, promosi, dan perubahan tim. Hal ini penting terutama untuk peran penting seperti manajer insiden, responden insiden, dan kepala komunikasi.

- Manajer insiden: Manajer insiden memiliki otoritas keseluruhan selama merespons peristiwa.
- Responden insiden: Responden insiden bertanggung jawab atas kegiatan investigasi dan remediasi. Orang-orang ini dapat berbeda berdasarkan jenis peristiwanya, tetapi biasanya adalah pengembang dan tim operasi yang bertanggung jawab atas aplikasi yang terkena dampak.
- Pimpinan komunikasi: Pimpinan komunikasi bertanggung jawab atas komunikasi internal dan eksternal, terutama komunikasi dengan lembaga publik, regulator, dan pelanggan.
- Pakar materi pelajaran (SMEs): Dalam kasus tim terdistribusi dan otonom, kami sarankan Anda mengidentifikasi beban kerja kritis SME untuk misi. Mereka memberikan wawasan tentang operasi dan klasifikasi data pada beban kerja krusial yang terpengaruh dalam peristiwa.

Pertimbangkan untuk menggunakan fitur AWS Systems Manager Incident Manager untuk merekam kontak utama, menentukan rencana respons, mengotomatiskan jadwal panggilan, dan membuat rencana eskalasi. Lakukan otomatisasi dan rotasi semua staf melalui jadwal jaga, sehingga tanggung jawab atas beban kerja dibagi di antara pemiliknya. Hal ini mendukung praktik-praktik yang baik, seperti menghasilkan metrik dan log yang relevan serta menentukan ambang batas alarm yang penting untuk beban kerja.

Identifikasi mitra eksternal: Perusahaan menggunakan alat yang dibangun oleh vendor perangkat lunak independen (ISVs), mitra, dan subkontraktor untuk membangun solusi yang berbeda bagi pelanggan mereka. Libatkan personel penting dari pihak-pihak ini yang dapat membantu merespons dan melakukan pemulihan dari suatu insiden. Kami menyarankan Anda mendaftar untuk tingkat yang AWS Support sesuai untuk mendapatkan akses cepat ke ahli materi AWS pelajaran melalui kasus dukungan. Pertimbangkan untuk melakukan hal yang serupa dengan semua penyedia solusi yang krusial untuk beban kerja. Beberapa peristiwa keamanan tertentu mengharuskan bisnis yang terdaftar di bursa saham memberi tahu lembaga publik dan badan pengatur yang relevan tentang peristiwa yang terjadi dan dampaknya. Pelihara dan perbarui informasi kontak untuk departemen yang relevan dan orang-orang yang bertanggung jawab.

Langkah-langkah implementasi

1. Siapkan sebuah solusi manajemen insiden.

   a. Pertimbangkan untuk melakukan deployment Incident Manager di akun Security Tooling Anda.
2. Tentukan kontak dalam solusi manajemen insiden Anda.
   a. Tentukan setidaknya dua jenis saluran kontak untuk setiap kontak (sepertiSMS, telepon, atau email), untuk memastikan jangkauan selama insiden.
3. Tentukan rencana respons.
   a. Identifikasi kontak yang paling tepat untuk digunakan selama insiden. Tentukan rencana eskalasi yang selaras dengan peran yang dimiliki oleh personel yang akan dilibatkan, bukan kontak individu. Pertimbangkan untuk memasukkan kontak yang mungkin bertanggung jawab untuk memberi tahu entitas eksternal, meskipun mereka tidak terlibat langsung untuk menyelesaikan insiden.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS02-BP03 Kegiatan operasi telah mengidentifikasi pemilik yang bertanggung jawab atas kinerjanya

Dokumen terkait:

- Panduan Respons Insiden Keamanan AWS

Contoh terkait:

- Kerangka kerja playbook pelanggan AWS
- Bersiap dan merespons insiden keamanan di lingkungan AWS Anda

Alat terkait:

- AWS Manajer Insiden Systems Manager

Video terkait:

- Pendekatan Amazon terhadap keamanan selama pengembangan

SEC10-BP02 Mengembangkan rencana manajemen insiden

Dokumen pertama yang dikembangkan untuk respons insiden adalah rencana respons insiden. Rencana respons insiden dirancang untuk menjadi dasar bagi program dan strategi respons insiden Anda.

Manfaat menerapkan praktik terbaik ini: Mengembangkan proses respons insiden yang menyeluruh dan jelas adalah kunci untuk program respons insiden yang sukses dan terukur. Ketika sebuah peristiwa keamanan terjadi, langkah dan alur kerja yang jelas dapat membantu Anda merespons secara tepat waktu. Anda mungkin sudah memiliki proses respons insiden sendiri. Terlepas dari keadaan saat ini, penting untuk memperbarui, mengulangi, dan menguji proses respons insiden Anda secara teratur.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Rencana manajemen insiden sangat penting untuk merespons, memitigasi, dan pulih dari potensi dampak yang ditimbulkan insiden keamanan. Rencana manajemen insiden adalah sebuah proses terstruktur untuk mengidentifikasi, memperbaiki, dan merespons insiden keamanan secara tepat waktu.

Cloud memiliki banyak peran dan persyaratan operasional yang sama yang juga ditemukan di lingkungan on-premise. Saat membuat sebuah rencana manajemen insiden, Anda harus mempertimbangkan strategi respons dan pemulihan yang paling selaras dengan hasil bisnis dan persyaratan kepatuhan Anda. Misalnya, jika Anda mengoperasikan beban kerja AWS yang RAMP sesuai dengan Fed di Amerika Serikat, ada baiknya untuk mematuhi Panduan Penanganan Keamanan Komputer NIST SP 800-61. Demikian pula, ketika mengoperasikan beban kerja dengan data informasi (PII) identitas pribadi Eropa, pertimbangkan skenario seperti bagaimana Anda dapat melindungi dan menanggapi masalah yang terkait dengan residensi data sebagaimana diamanatkan oleh Peraturan Perlindungan Data Umum Uni Eropa () Peraturan. GDPR

Saat membuat rencana manajemen insiden untuk beban kerja Anda AWS, mulailah dengan Model Tanggung Jawab AWS Bersama untuk membangun defense-in-depth pendekatan terhadap respons insiden. Dalam model ini, AWS mengelola keamanan cloud, dan Anda bertanggung jawab atas keamanan di cloud. Ini artinya Anda mempertahankan kontrol dan bertanggung jawab atas kontrol keamanan yang ingin Anda implementasikan. Panduan Respons Insiden Keamanan AWS menguraikan konsep utama dan panduan mendasar untuk membangun rencana manajemen insiden yang berorientasi cloud.

Rencana manajemen insiden yang efektif harus diulang-ulang (iterasi) secara berkelanjutan, dan harus tetap mutakhir sesuai tujuan-tujuan operasi cloud Anda. Pertimbangkan untuk menggunakan rencana implementasi yang diuraikan di bawah ini saat Anda membuat dan mengembangkan rencana manajemen insiden Anda.

Langkah-langkah implementasi

Menentukan peran dan tanggung jawab

Menangani peristiwa keamanan membutuhkan disiplin lintas organisasi dan komitmen untuk bertindak. Dalam struktur organisasi Anda, harus ada banyak orang yang bertanggung jawab, akuntabel, dimintai pendapat, atau diinformasikan saat terjadi insiden, seperti perwakilan dari sumber daya manusia (SDM), tim eksekutif, dan hukum. Pertimbangkan peran dan tanggung jawab ini, dan apakah ada pihak ketiga yang harus dilibatkan. Perhatikan bahwa banyak kawasan yang memiliki undang-undang setempat yang mengatur apa yang seharusnya dilakukan dan tidak boleh dilakukan. Meskipun mungkin tampak birokratis untuk membangun bagan yang bertanggung jawab, akuntabel, dikonsultasikan, dan diinformasikan (RACI) untuk rencana respons keamanan Anda, hal itu memfasilitasi komunikasi yang cepat dan langsung dan dengan jelas menguraikan kepemimpinan di berbagai tahap acara.

Selama insiden, termasuk pemilik dan pengembang aplikasi dan sumber daya yang terkena dampak adalah kunci karena mereka adalah ahli materi pelajaran (SMEs) yang dapat memberikan informasi dan konteks untuk membantu dalam mengukur dampak. Pastikan untuk mempraktikkan dan membangun hubungan dengan developer serta pemilik aplikasi sebelum Anda mengandalkan keahlian mereka untuk respons insiden. Pemilik aplikasi atauSMEs, seperti administrator atau insinyur cloud Anda, mungkin perlu bertindak dalam situasi di mana lingkungan tidak dikenal atau memiliki kompleksitas, atau di mana responden tidak memiliki akses.

Terakhir, partner tepercaya mungkin terlibat dalam penyelidikan atau respons karena mereka dapat memberikan keahlian tambahan dan pengawasan yang berharga. Ketika tidak ada orang yang memiliki keterampilan ini dalam tim Anda sendiri, ada baiknya Anda menyewa pihak eksternal untuk bantuan.

Memahami tim AWS respons dan dukungan

- AWS Support
  - AWS Supportmenawarkan berbagai rencana yang menyediakan akses ke alat dan keahlian yang mendukung keberhasilan dan kesehatan operasional AWS solusi Anda. Jika Anda memerlukan dukungan teknis dan lebih banyak sumber daya untuk membantu merencanakan,

menerapkan, dan mengoptimalkan AWS lingkungan Anda, Anda dapat memilih paket dukungan yang paling sesuai dengan kasus AWS penggunaan Anda.

  - Pertimbangkan Support Center in AWS Management Console (diperlukan login) sebagai titik kontak utama untuk mendapatkan dukungan untuk masalah yang memengaruhi AWS sumber daya Anda. Akses ke AWS Support dikendalikan oleh AWS Identity and Access Management. Untuk informasi selengkapnya tentang mendapatkan akses ke AWS Support fitur, lihat Memulai AWS Support.

- AWS Tim Respons Insiden Pelanggan (CIRT)
  - AWS Customer Incident Response Team (CIRT) adalah AWS tim global khusus 24/7 yang memberikan dukungan kepada pelanggan selama acara keamanan aktif di sisi pelanggan Model Tanggung Jawab AWS Bersama.
  - Ketika AWS CIRT mendukung Anda, mereka memberikan bantuan dengan triase dan pemulihan untuk acara keamanan aktif. AWS Mereka dapat membantu dalam analisis akar penyebab melalui penggunaan log AWS layanan dan memberi Anda rekomendasi untuk pemulihan. Mereka juga dapat memberikan rekomendasi dan praktik terbaik keamanan untuk membantu Anda menghindari peristiwa keamanan di masa depan.
  - AWS pelanggan dapat terlibat AWS CIRT melalui AWS Support kasus.
- DDoSdukungan respon
  - AWS penawaran AWS Shield, yang menyediakan layanan perlindungan penolakan layanan (DDoS) terdistribusi terkelola yang melindungi aplikasi web yang berjalan. AWS Shield menyediakan deteksi selalu aktif dan mitigasi inline otomatis yang dapat meminimalkan waktu henti dan latensi aplikasi, sehingga tidak perlu terlibat untuk mendapatkan manfaat dari perlindungan. AWS Support DDoS Ada dua tingkatan Shield: AWS Shield Standard dan AWS Shield Advanced. Untuk mengetahui perbedaan antara kedua tingkatan ini, silakan lihat Dokumentasi fitur Shield.
- AWS Managed Services (AMS)
  - AWS Managed Services (AMS) menyediakan manajemen berkelanjutan AWS infrastruktur Anda sehingga Anda dapat fokus pada aplikasi Anda. Dengan menerapkan praktik terbaik untuk memelihara infrastruktur Anda, AMS membantu mengurangi overhead dan risiko operasional Anda. AMSmengotomatiskan aktivitas umum seperti permintaan perubahan, pemantauan, manajemen patch, keamanan, dan layanan pencadangan, dan menyediakan layanan siklus hidup penuh untuk menyediakan, menjalankan, dan mendukung infrastruktur Anda.
  - AMSbertanggung jawab untuk menyebarkan serangkaian kontrol detektif keamanan dan memberikan respons 24/7 pertama terhadap peringatan. Saat peringatan dimulai, AMS ikuti satu

set standar buku pedoman otomatis dan manual untuk memverifikasi respons yang konsisten. Buku pedoman ini dibagikan dengan AMS pelanggan selama orientasi sehingga mereka dapat mengembangkan dan mengoordinasikan respons dengannya. AMS

Kembangkan rencana respons insiden

Rencana respons insiden dirancang untuk menjadi dasar bagi program dan strategi respons insiden Anda. Rencana respons insiden harus dalam bentuk dokumen resmi. Rencana respons insiden biasanya menyertakan bagian-bagian ini:

- ikhtisar tim respons insiden: Menguraikan tujuan dan fungsi tim respons insiden.
- Peran dan tanggung jawab: Membuat daftar pemangku kepentingan respons insiden dan menjabarkan peran mereka ketika insiden terjadi.
- Rencana komunikasi: Detail informasi kontak dan bagaimana mekanisme komunikasi selama insiden.
- Metode komunikasi Backup: Ini adalah praktik terbaik untuk memiliki out-of-band komunikasi sebagai cadangan untuk komunikasi insiden. Contoh aplikasi yang menyediakan saluran out-of-band komunikasi yang aman adalah AWS Wickr.
- Fase respons insiden dan tindakan yang perlu diambil: Mengenumerasi fase respons insiden, (misalnya, mendeteksi, menganalisis, memberantas, menahan, dan memulihkan), termasuk tindakan tingkat tinggi yang harus diambil dalam fase-fase tersebut.
- Definisi keparahan insiden dan prioritas: Memerinci cara mengklasifikasikan tingkat keparahan suatu insiden, bagaimana memprioritaskan insiden, lalu bagaimana definisi keparahan mempengaruhi prosedur eskalasi.

Meskipun bagian-bagian ini umumnya ada di perusahaan dalam berbagai ukuran dan industri yang berbeda, rencana respons insiden akan berbeda-beda di setiap organisasi. Anda perlu membangun rencana respons insiden yang paling cocok untuk organisasi Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC04 (Bagaimana Anda mendeteksi dan menyelidiki peristiwa keamanan?)

Dokumen terkait:

- AWS Panduan Respons Insiden Keamanan
- NIST: Panduan Penanganan Insiden Keamanan Komputer

SEC10-BP03 Siapkan kemampuan forensik

Menjelang insiden keamanan, pertimbangkan untuk mengembangkan kemampuan forensik guna mendukung investigasi peristiwa keamanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Konsep dari forensik lokal tradisional berlaku untuk. AWS Untuk informasi kunci untuk mulai membangun kemampuan forensik di AWS Cloud, lihat Strategi lingkungan investigasi forensik di. AWS Cloud

Setelah Anda menyiapkan lingkungan dan Akun AWS struktur untuk forensik, tentukan teknologi yang diperlukan untuk secara efektif melakukan metodologi forensik yang sehat di empat fase:

- Koleksi: Kumpulkan AWS log yang relevan, seperti AWS CloudTrail, AWS Config, Log VPC Aliran, dan log tingkat host. Kumpulkan snapshot, backup, dan dump memori dari sumber daya yang terkena dampak AWS jika tersedia.
- Pemeriksaan: Memeriksa data yang dikumpulkan dengan mengekstraksi dan menilai informasi yang relevan.
- Analisis: Menganalisis data yang dikumpulkan untuk memahami insiden dan menarik kesimpulan dari insiden tersebut.
- Pelaporan: Menyajikan informasi yang dihasilkan dari fase analisis.

Langkah-langkah implementasi

Persiapkan lingkungan forensik Anda

AWS Organizationsmembantu Anda mengelola dan mengatur AWS lingkungan secara terpusat saat Anda tumbuh dan meningkatkan AWS sumber daya. Sebuah AWS organisasi mengkonsolidasikan Anda Akun AWS sehingga Anda dapat mengelolanya sebagai satu unit. Anda dapat menggunakan unit organisasi (OUs) untuk mengelompokkan akun bersama untuk mengelola sebagai satu unit.

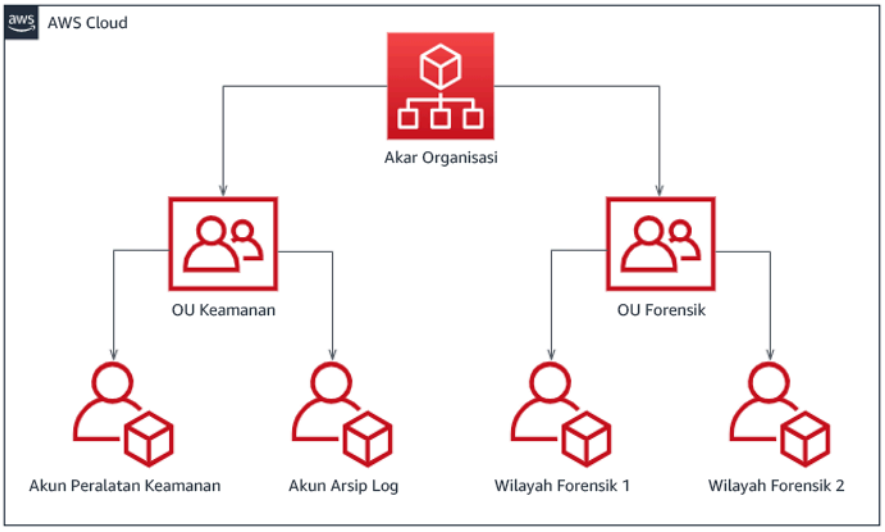
Untuk respons insiden, akan sangat membantu untuk memiliki Akun AWS struktur yang mendukung fungsi respons insiden, yang mencakup OU keamanan dan OU forensik. Dalam unit organisasi keamanan, Anda harus memiliki akun untuk:

- Arsip log: Agregat log dalam arsip log Akun AWS dengan izin terbatas.
- Alat keamanan: Memusatkan layanan keamanan dalam alat Akun AWS keamanan. Akun ini beroperasi sebagai administrator yang didelegasikan untuk layanan keamanan.

Dalam forensik unit organisasi, Anda memiliki opsi untuk menerapkan satu akun forensik atau akun-akun untuk setiap Wilayah tempat Anda beroperasi, bergantung pada mana yang paling sesuai untuk model bisnis dan operasional Anda. Jika Anda membuat akun forensik per Wilayah, Anda dapat memblokir pembuatan AWS sumber daya di luar Wilayah tersebut dan mengurangi risiko sumber daya disalin ke wilayah yang tidak diinginkan. Misalnya, jika Anda hanya beroperasi di Wilayah AS Timur (Virginia Utara) (`us-east-1`) dan AS Barat (Oregon) (`us-west-2`), maka Anda akan memiliki dua akun yang ada di forensik OU: satu untuk `us-east-1` dan satu untuk `us-west-2`.

Anda dapat membuat forensik Akun AWS untuk beberapa Wilayah. Anda harus berhati-hati dalam menyalin AWS sumber daya ke akun tersebut untuk memverifikasi bahwa Anda selaras dengan persyaratan kedaulatan data Anda. Karena penyediaan akun baru membutuhkan waktu, akun forensik harus dibuat dan digunakan jauh sebelum insiden, sehingga bisa siap digunakan oleh responden secara efektif ketika merespons insiden.

Diagram berikut menampilkan struktur akun sampel, termasuk unit organisasi forensik dengan akun forensik per Wilayah:




Struktur akun per wilayah untuk respons insiden

Menangkap cadangan dan snapshot

Menyiapkan cadangan sistem kunci dan basis data sangat penting untuk pemulihan dari insiden keamanan dan untuk tujuan forensik. Dengan memiliki cadangan, Anda dapat memulihkan sistem Anda ke keadaan aman sebelumnya. Pada AWS, Anda dapat mengambil snapshot dari berbagai sumber daya. Snapshot memberi Anda point-in-time cadangan sumber daya tersebut. Ada banyak AWS layanan yang dapat mendukung Anda dalam pencadangan dan pemulihan. Untuk membaca detail tentang layanan dan pendekatan pencadangan dan pemulihan ini, lihat Panduan Preskriptif Pencadangan dan Pemulihan dan Gunakan cadangan untuk memulihkan dari insiden keamanan.

Terutama ketika berhubungan dengan situasi seperti ransomware, sangat penting agar cadangan Anda dilindungi dengan baik. Untuk membaca panduan tentang cara mengamankan cadangan Anda, silakan lihat 10 teratas praktik terbaik untuk mengamankan cadangan di AWS . Selain mengamankan cadangan, Anda juga sebaiknya menguji proses pencadangan dan pemulihan Anda secara teratur untuk memverifikasi bahwa teknologi dan proses yang Anda miliki berfungsi sesuai harapan.

Mengotomatiskan forensik

Selama peristiwa keamanan, tim respons insiden Anda harus dapat mengumpulkan dan menganalisis bukti dengan cepat sambil mempertahankan akurasi untuk periode waktu sekitar peristiwa (seperti menangkap log yang terkait dengan peristiwa atau sumber daya tertentu atau mengumpulkan dump memori dari EC2 instans Amazon). Ini adalah hal yang menantang dan memakan waktu bagi tim respons insiden untuk mengumpulkan bukti yang relevan secara manual, terutama di sejumlah besar instans dan akun. Selain itu, kesalahan manusia rentan terjadi dalam pengumpulan secara manual. Untuk alasan-alasan ini, Anda harus mengembangkan dan mengimplementasikan otomatisasi untuk forensik sebisa mungkin.

AWS menawarkan sejumlah sumber daya otomatisasi untuk forensik, yang tercantum di bagian Sumber Daya berikut. Sumber daya ini adalah contoh pola forensik yang telah kami kembangkan dan telah diterapkan pelanggan. Meskipun sumber daya ini mungkin merupakan arsitektur referensi yang berguna untuk memulai, pertimbangkan untuk memodifikasinya atau membuat pola otomatisasi forensik baru berdasarkan lingkungan, persyaratan, alat, dan proses forensik Anda.

Sumber daya

Dokumen terkait:

- AWS Panduan Respons Insiden Keamanan - Kembangkan Kemampuan Forensik
- AWS Panduan Respons Insiden Keamanan - Sumber Daya Forensik
- Strategi lingkungan investigasi forensik di AWS Cloud

- Cara mengotomatiskan koleksi disk forensik di AWS
- AWS Panduan Preskriptif - Mengotomatiskan respons insiden dan forensik

Video terkait:

- Mengotomatiskan Respons Insiden dan Forensik

Contoh terkait:

- Respons Insiden Otomatis dan Kerangka Forensik
- Orkestrator Forensik Otomatis untuk Amazon EC2

SEC10-BP04 Mengembangkan dan menguji pedoman respons insiden keamanan

Bagian penting dari mempersiapkan proses respons insiden Anda adalah mengembangkan playbook. Playbook respons insiden memberikan serangkaian panduan preskriptif dan langkah-langkah yang harus diikuti ketika terjadi peristiwa keamanan. Struktur dan langkah yang jelas akan menyederhanakan respons dan mengurangi kemungkinan kesalahan manusia.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Playbook sebaiknya dibuat untuk skenario insiden seperti:

- Insiden yang diantisipasi: Playbook harus dibuat untuk insiden yang Anda antisipasi. Hal ini termasuk ancaman seperti denial of service (DoS), ransomware, dan pembobolan kredensial.
- Temuan atau peringatan keamanan yang diketahui: Buku pedoman harus dibuat untuk temuan dan peringatan keamanan Anda yang diketahui, seperti temuan. GuardDuty Anda mungkin menerima GuardDuty temuan dan berpikir, “Sekarang apa?” Untuk mencegah kesalahan penanganan atau mengabaikan GuardDuty temuan, buat buku pedoman untuk setiap temuan potensial. GuardDuty Beberapa rincian remediasi dan panduan dapat ditemukan dalam GuardDutydokumentasi. Perlu dicatat bahwa tidak GuardDuty diaktifkan secara default dan menimbulkan biaya. Untuk detail selengkapnya GuardDuty, lihat Lampiran A: Definisi kemampuan cloud - Visibilitas dan peringatan.

Playbook harus berisi langkah-langkah teknis yang akan dijalankan oleh analis keamanan untuk menyelidiki dan merespons insiden keamanan potensial secara memadai.

Langkah-langkah implementasi

Item yang akan disertakan dalam playbook meliputi:

- Gambaran umum playbook: Skenario risiko atau insiden apa yang ditangani oleh playbook ini? Apa tujuan dari playbook ini?
- Persyaratan: Log, mekanisme deteksi, dan alat otomatis apa yang diperlukan untuk skenario insiden ini? Apa notifikasi yang diharapkan?
- Informasi komunikasi dan eskalasi: Siapa saja yang terlibat dan apa informasi kontak mereka? Apa saja tanggung jawab setiap pemangku kepentingan?
- Langkah respons: Di seluruh fase respons insiden, langkah taktis apa yang perlu diambil? Kueri apa yang perlu dijalankan analis? Kode apa yang perlu dijalankan untuk mencapai hasil yang diinginkan?
  - Deteksi: Bagaimana insiden tersebut akan terdeteksi?
  - Analisis: Bagaimana cakupan dampak akan ditentukan?
  - Tahan: Bagaimana insiden akan diisolasi untuk membatasi cakupan?
  - Berantas: Bagaimana ancaman akan dihilangkan dari lingkungan?
  - Pulihkan: Bagaimana sistem atau sumber daya yang terpengaruh akan dibawa kembali ke produksi?
- Hasil yang diharapkan: Setelah kueri dan kode dijalankan, apa hasil yang diharapkan dari playbook tersebut?

Sumber daya

Praktik terbaik Well-Architected terkait:

- SEC10-BP02 - Mengembangkan rencana manajemen insiden

Dokumen terkait:

- Kerangka Kerja untuk Playbook Respons Insiden
- Mengembangkan Playbook Respons Insiden Anda sendiri
- Contoh Playbook Respons Insiden
- Membangun runbook respons AWS insiden menggunakan pedoman Jupyter dan Danau CloudTrail

SEC10-BP05 Akses pra-penyediaan

Verifikasi bahwa responden insiden memiliki akses yang benar yang telah disediakan sebelumnya AWS untuk mengurangi waktu yang diperlukan untuk penyelidikan hingga pemulihan.

Anti-pola umum:

- Menggunakan akun root untuk merespons insiden.
- Mengubah akun-akun yang ada.
- Memanipulasi IAM izin secara langsung saat memberikan elevasi just-in-time hak istimewa.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS merekomendasikan mengurangi atau menghilangkan ketergantungan pada kredensi berumur panjang sedapat mungkin, demi kredensi sementara dan mekanisme eskalasi hak istimewa. just-in-time Kredensial berumur panjang rentang terkena risiko keamanan dan meningkatkan biaya overhead operasional. Untuk sebagian besar tugas manajemen, serta tugas respons insiden, kami sarankan Anda untuk menerapkan federasi identitas bersama eskalasi sementara untuk akses administratif. Di model ini, seorang pengguna meminta peningkatan ke tingkat hak akses yang lebih tinggi (seperti peran respons insiden) dan, apabila pengguna tersebut memenuhi syarat peningkatan hak, permintaan tersebut dikirimkan ke seorang pemberi persetujuan. Jika permintaan disetujui, pengguna akan menerima satu set kredensial AWS sementara yang dapat digunakan untuk menyelesaikan tugas mereka. Setelah kredensial ini kedaluwarsa, pengguna harus mengirimkan permintaan peningkatan baru.

Kami menyarankan penggunaan peningkatan hak akses sementara di sebagian besar skenario respons insiden. Cara yang benar untuk melakukan hal itu adalah dengan menggunakan AWS Security Token Service dan kebijakan sesi untuk mencakup akses.

Terdapat skenario di mana identitas terfederasi tidak tersedia, seperti:

- Pemadaman yang berkaitan dengan penyedia identitas (IdP) yang terganggu.
- Kesalahan konfigurasi atau kesalahan manusiawi yang menyebabkan rusaknya sistem manajemen akses terfederasi.
- Aktivitas berbahaya seperti peristiwa penolakan layanan (DDoS) terdistribusi atau rendering tidak tersedianya sistem.

Dalam kasus sebelumnya, harus ada akses kaca pecah darurat yang dikonfigurasi untuk memungkinkan penyelidikan dan perbaikan insiden dilakukan secara tepat waktu. Kami menyarankan Anda menggunakan pengguna, grup, atau peran dengan izin yang sesuai untuk melakukan tugas dan mengakses AWS sumber daya. Gunakan pengguna root hanya untuk tugas yang memerlukan kredensial pengguna root. Untuk memverifikasi bahwa responden insiden memiliki tingkat akses yang benar AWS dan sistem lain yang relevan, kami merekomendasikan pra-penyediaan akun khusus. Akun-akun tersebut memerlukan akses istimewa, dan harus dikontrol dan dipantau secara ketat. Akun-akun tersebut harus dibuat dengan hak akses paling rendah yang diperlukan untuk menjalankan tugas yang diperlukan, dan tingkat akses harus didasarkan pada playbook yang dibuat sebagai bagian dari rencana manajemen insiden.

Gunakan pengguna dan peran yang dibuat khusus sebagai praktik terbaik. Meningkatnya akses pengguna atau peran untuk sementara melalui penambahan IAM kebijakan membuat tidak jelas akses apa yang dimiliki pengguna selama insiden, dan risiko hak istimewa yang meningkat tidak dicabut.

Penting untuk menghapus dependensi sebanyak mungkin untuk memastikan akses dapat diperoleh dalam sebanyak mungkin skenario kegagalan. Untuk mendukung hal ini, buat buku pedoman untuk memverifikasi bahwa pengguna respons insiden dibuat sebagai pengguna di akun keamanan khusus, dan tidak dikelola melalui solusi Federation atau single sign-on () SSO yang ada. Tiap-tiap perespons harus memiliki akun dengan nama mereka sendiri. Konfigurasi akun harus menerapkan kebijakan kata sandi yang kuat dan otentikasi multi-faktor (). MFA Jika buku pedoman respons insiden hanya memerlukan akses ke AWS Management Console, pengguna tidak boleh memiliki kunci akses yang dikonfigurasi dan harus secara eksplisit dilarang membuat kunci akses. Ini dapat dikonfigurasi dengan IAM kebijakan atau kebijakan kontrol layanan (SCPs) seperti yang disebutkan dalam Praktik Terbaik AWS Keamanan untuk AWS Organizations SCPs. Pengguna tidak boleh memiliki hak ases selain kemampuan untuk mengambil peran respons insiden di akun-akun lainnya.

Selama insiden, mungkin diperlukan pemberian akses ke individu internal atau eksternal untuk mendukung aktivitas penyelidikan, perbaikan, atau pemulihan. Pada kasus ini, gunakan mekanisme playbook yang disebutkan sebelumnya, dan harus ada proses untuk memverifikasi bahwa akses tambahan apa pun segera dicabut setelah insiden selesai.

Untuk memverifikasi bahwa penggunaan peran respons insiden dapat dipantau dan diaudit dengan benar, penting bahwa IAM akun yang dibuat untuk tujuan ini tidak dibagi antar individu, dan bahwa tidak digunakan kecuali diperlukan untuk tugas tertentu. Pengguna root akun AWS Jika pengguna root diperlukan (misalnya, IAM akses ke akun tertentu tidak tersedia), gunakan proses terpisah

dengan buku pedoman yang tersedia untuk memverifikasi ketersediaan kredenal masuk pengguna root dan token. MFA

Untuk mengonfigurasi IAM kebijakan untuk peran respons insiden, pertimbangkan untuk menggunakan IAMAccess Analyzer untuk membuat kebijakan berdasarkan AWS CloudTrail log. Untuk melakukannya, berikan akses administrator ke peran respons insiden di akun non-produksi dan jalankan playbook Anda. Setelah selesai, kebijakan dapat dibuat yang hanya mengizinkan tindakan yang diambil. Kebijakan ini kemudian dapat diterapkan ke semua peran respons insiden di semua akun. Anda mungkin ingin membuat IAM kebijakan terpisah untuk setiap buku pedoman agar manajemen dan audit lebih mudah. Contoh playbook dapat mencakup rencana respons untuk ransomware, pembobolan data, hilangnya akses produksi, dan skenario lain.

Gunakan akun respons insiden untuk mengambil IAMperan respons insiden khusus di akun lain Akun AWS. Peran ini harus dikonfigurasi agar hanya dapat diasumsikan oleh pengguna di akun keamanan, dan hubungan kepercayaan harus mengharuskan prinsipal panggilan telah diautentikasi menggunakan. MFA Peran harus menggunakan kebijakan dengan cakupan ketat untuk mengontrol akses. IAM Pastikan bahwa semua `AssumeRole` permintaan untuk peran ini masuk CloudTrail dan diperingatkan, dan tindakan apa pun yang diambil menggunakan peran ini dicatat.

Sangat disarankan agar IAM akun dan IAM peran diberi nama dengan jelas agar mudah ditemukan di CloudTrail log. Contohnya adalah memberi nama IAM akun *<USER_ID>*`-BREAK-GLASS` dan IAM perannya`BREAK-GLASS-ROLE`.

CloudTraildigunakan untuk mencatat API aktivitas di AWS akun Anda dan harus digunakan untuk mengonfigurasi peringatan tentang penggunaan peran respons insiden. Lihat postingan blog tentang konfigurasi perintanan saat kunci root digunakan. Instruksi dapat dimodifikasi untuk mengonfigurasi CloudWatch metrik Amazon filter-to-filter pada `AssumeRole` peristiwa yang terkait dengan IAM peran respons insiden:

```
{ $.eventName = "AssumeRole" && $.requestParameters.roleArn =
 "<INCIDENT_RESPONSE_ROLE_ARN>" && $.userIdentity.invokedBy NOT EXISTS && $.eventType !
= "AwsServiceEvent" }
```

Karena peran respons insiden kemungkinan memiliki tingkat akses yang tinggi, peringatan-peringatan ini harus menjangkau grup yang luas dan ditindaklanjuti segera.

Selama insiden, ada kemungkinan bahwa responden mungkin memerlukan akses ke sistem yang tidak diamankan secara langsung. IAM Ini dapat mencakup instans Amazon Elastic Compute Cloud, database Amazon Relational Database Service, atau platform ( software-as-a-serviceSaaS). Sangat

disarankan bahwa daripada menggunakan protokol asli seperti SSH atauRDP, AWS Systems Manager Session Managerdigunakan untuk semua akses administratif ke instans AmazonEC2. Akses ini dapat dikontrol menggunakanIAM, yang aman dan diaudit. Dimungkinkan juga untuk mengotomatiskan bagian dari playbook Anda dengan menggunakan dokumen AWS Systems Manager Run Command, yang dapat mengurangi kesalahan pengguna dan meningkatkan waktu untuk pemulihan. Untuk akses ke database dan alat pihak ketiga, kami sarankan untuk menyimpan kredensi akses AWS Secrets Manager dan memberikan akses ke peran responden insiden.

Terakhir, pengelolaan IAM akun respons insiden harus ditambahkan ke proses Joiner, Movers, dan Leavers Anda dan ditinjau dan diuji secara berkala untuk memverifikasi bahwa hanya akses yang dimaksudkan yang diizinkan.

Sumber daya

Dokumen terkait:

- Mengelola akses sementara yang ditinggikan ke AWS lingkungan Anda
- AWS Panduan Respons Insiden Keamanan
- AWS Elastic Disaster Recovery
- AWS Systems Manager Incident Manager
- Menyetel kebijakan kata sandi akun untuk IAM pengguna
- Menggunakan otentikasi multi-faktor () MFA di AWS
- Mengkonfigurasi Akses Lintas Akun dengan MFA
- Menggunakan IAM Access Analyzer untuk menghasilkan kebijakan IAM
- Praktik Terbaik untuk Kebijakan Kontrol AWS Organizations Layanan di Lingkungan Multi-Akun
- Cara Menerima Pemberitahuan Saat Kunci Akses Root AWS Akun Anda Digunakan
- Buat izin sesi berbutir halus menggunakan kebijakan terkelola IAM

Video terkait:

- Mengotomatiskan Respon Insiden dan Forensik di AWS
- DIYpanduan untuk runbook, laporan insiden, dan respons insiden
- Mempersiapkan dan menanggapi insiden keamanan di lingkungan Anda AWS

Contoh terkait:

- Lab: Pengaturan AWS Akun dan Pengguna Root
- Lab: Respons Insiden dengan AWS Konsol dan CLI

SEC10-BP06 Alat pra-penerapan

Pastikan personel keamanan sejak awal telah melakukan deployment alat yang tepat untuk mengurangi waktu investigasi melalui pemulihan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Untuk mengotomatiskan respons keamanan dan fungsi operasi, Anda dapat menggunakan seperangkat APIs dan alat yang komprehensif dari AWS. Anda dapat sepenuhnya mengotomatiskan manajemen identitas, keamanan jaringan, perlindungan data, dan kemampuan pemantauan, serta menyediakannya menggunakan metode pengembangan perangkat lunak populer yang sudah Anda gunakan. Saat Anda membangun otomatisasi keamanan, sistem Anda dapat memantau, meninjau, dan menginisiasi respons, tanpa memerlukan orang untuk memantau posisi keamanan Anda dan memberikan reaksi terhadap peristiwa secara manual.

Jika tim respons insiden Anda terus merespons peringatan dengan cara yang sama, mereka berisiko mengalami kelelahan alarm (alarm fatigue). Seiring berjalannya waktu, tim dapat menjadi tidak peka terhadap peringatan sehingga dapat membuat kesalahan saat menangani situasi biasa atau melewatkan peringatan yang tidak biasa. Otomatisasi membantu mencegah kelelahan alarm dengan menggunakan fungsi yang memproses peringatan biasa dan repetitif, sehingga manusia cukup menangani insiden yang sensitif dan unik. Mengintegrasikan sistem deteksi anomali, seperti Amazon, AWS CloudTrail Insights GuardDuty, dan Amazon CloudWatch Anomaly Detection, dapat mengurangi beban peringatan berbasis ambang batas umum.

Anda dapat memperbaiki proses manual dengan mengotomatiskan langkah-langkah dalam proses secara terprogram. Setelah Anda menentukan perbaikan pola pada peristiwa, Anda dapat menguraikan pola tersebut menjadi logika yang dapat ditindaklanjuti, dan menulis kode untuk menjalankan logika tersebut. Pemberi respons selanjutnya dapat menjalankan kode tersebut untuk memperbaiki masalah. Seiring berjalannya waktu, Anda dapat mengotomatiskan lebih banyak langkah, dan pada akhirnya secara otomatis menangani semua jenis insiden yang biasa muncul.

Selama penyelidikan keamanan, Anda harus dapat meninjau log yang relevan untuk mencatat dan memahami cakupan serta garis waktu lengkap insiden tersebut. Log juga diperlukan untuk

pembuatan peringatan, yang menunjukkan terjadinya tindakan tertentu yang menarik. Sangat penting untuk memilih, mengaktifkan, menyimpan, serta mengatur mekanisme kueri dan pengambilan, serta mengatur peringatan. Selain itu, cara yang efektif untuk menyediakan alat untuk mencari data log adalah Amazon Detective.

AWS menawarkan lebih dari 200 layanan cloud dan ribuan fitur. Kami menyarankan Anda meninjau layanan yang dapat mendukung dan menyederhanakan strategi respons insiden Anda.

Selain pencatatan log, Anda harus mengembangkan dan menerapkan strategi pemberian tag. Penandaan dapat membantu memberikan konteks seputar tujuan AWS sumber daya. Pemberian tag juga dapat digunakan untuk otomatisasi.

Langkah-langkah implementasi

Memilih dan mengatur log untuk analisis dan peringatan

Lihat dokumentasi berikut tentang cara mengonfigurasi pencatatan log untuk respons insiden:

- Strategi pencatatan log untuk respons insiden keamanan
- SEC04-BP01 Mengkonfigurasi layanan dan pencatatan aplikasi

Aktifkan layanan keamanan untuk mendukung deteksi dan respons

AWS menyediakan kemampuan detektif, pencegahan, dan responsif asli, dan layanan lainnya dapat digunakan untuk merancang solusi keamanan khusus. Untuk daftar layanan yang paling relevan untuk respons insiden keamanan, lihat Definisi kemampuan cloud.

Mengembangkan dan menerapkan strategi pemberian tag

Memperoleh informasi kontekstual tentang kasus penggunaan bisnis dan pemangku kepentingan internal yang relevan di sekitar AWS sumber daya bisa jadi sulit. Salah satu cara untuk melakukannya adalah dalam bentuk tag, yang menetapkan metadata ke AWS sumber daya Anda dan terdiri dari kunci dan nilai yang ditentukan pengguna. Anda dapat menggunakan tag untuk mengelompokkan sumber daya berdasarkan tujuan, pemilik, lingkungan, jenis data yang diproses, dan kriteria lainnya yang Anda pilih.

Memiliki strategi penandaan yang konsisten dapat mempercepat waktu respons dan meminimalkan waktu yang dihabiskan untuk konteks organisasi dengan memungkinkan Anda mengidentifikasi dan membedakan informasi kontekstual tentang sumber daya dengan cepat. AWS Tag juga dapat

berfungsi sebagai mekanisme untuk memulai otomatisasi respons. Untuk detail selengkapnya tentang apa yang harus diberi tag, lihat Menandai AWS sumber daya Anda. Anda harus terlebih dahulu menentukan tag yang ingin Anda terapkan di organisasi Anda. Setelah itu, Anda akan menerapkan dan menegakkan strategi pemberian tag ini. Untuk detail selengkapnya tentang implementasi dan penegakan, lihat Menerapkan strategi penandaan AWS sumber daya menggunakan Kebijakan AWS Tag dan Kebijakan Kontrol Layanan (SCPs).

Sumber daya

Praktik terbaik Well-Architected terkait:

- SEC04-BP01 Mengkonfigurasi layanan dan pencatatan aplikasi
- SEC04-BP02 Tangkap log, temuan, dan metrik di lokasi standar

Dokumen terkait:

- Strategi pencatatan log untuk respons insiden keamanan
- Definisi kemampuan cloud respons insiden

Contoh terkait:

- Deteksi dan Respons Ancaman dengan Detektif Amazon GuardDuty dan Amazon
- Lokakarya Security Hub
- Manajemen Kerentanan dengan Amazon Inspector

SEC10-BP07 Jalankan simulasi

Organisasi tumbuh dan berkembang dari waktu ke waktu, begitu juga dengan lanskap ancaman. Oleh karena itu, penting untuk terus-menerus mengkaji kemampuan Anda dalam merespons insiden. Menjalankan simulasi (juga dikenal dengan nama game day) adalah salah satu metode yang dapat digunakan untuk melakukan penilaian ini. Simulasi menggunakan skenario peristiwa keamanan dunia nyata yang dirancang untuk meniru taktik, teknik, dan prosedur aktor ancaman (TTPs) dan memungkinkan organisasi untuk melatih dan mengevaluasi kemampuan respons insiden mereka dengan menanggapi peristiwa cyber tiruan ini karena mungkin terjadi dalam kenyataan.

Manfaat menerapkan praktik terbaik ini: Simulasi memiliki berbagai manfaat:

- Memvalidasi kesiapan siber dan mengembangkan kepercayaan diri responden insiden Anda.

- Menguji akurasi dan efisiensi alat serta alur kerja.
- Menyempurnakan metode komunikasi dan eskalasi yang selaras dengan rencana respons insiden Anda.
- Memberikan kesempatan untuk merespons vektor yang kurang umum.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Ada tiga jenis simulasi utama:

- Latihan meja: Pendekatan latihan meja dalam simulasi adalah sesi berbasis diskusi yang melibatkan berbagai pemangku kepentingan respons insiden untuk mempraktikkan peran dan tanggung jawab serta menggunakan alat komunikasi dan playbook yang telah ditetapkan. Penyelenggaraan latihan biasanya dapat dilakukan dalam sehari penuh di tempat virtual, bangunan fisik, atau kombinasi keduanya. Karena berbasis diskusi, latihan meja berfokus pada proses, orang, dan kolaborasi. Teknologi merupakan bagian tak terpisahkan dari diskusi, tetapi penggunaan nyata alat atau skrip respons insiden umumnya bukan bagian dari latihan meja.
- Latihan tim ungu: Latihan Tim Ungu meningkatkan level kolaborasi antara tim responden insiden (Tim Biru) dan tim aktor ancaman simulasi (Tim Merah). Tim biru terdiri dari anggota pusat operasi keamanan (SOC), tetapi juga dapat mencakup pemangku kepentingan lain yang akan terlibat selama acara cyber yang sebenarnya. Tim merah terdiri dari tim uji penetrasi atau pemangku kepentingan utama yang terlatih dalam hal keamanan ofensif. Tim Merah bekerja secara kolaboratif dengan fasilitator latihan dalam merancang skenario yang akurat dan memungkinkan. Selama latihan tim ungu, fokus utama adalah pada mekanisme deteksi, alat, dan prosedur operasi standar (SOPs) yang mendukung upaya respons insiden.
- Latihan Tim Merah: Dalam latihan Tim Merah, penyerang (Tim Merah) melakukan simulasi untuk mencapai tujuan tertentu atau serangkaian tujuan dari cakupan yang telah ditentukan sebelumnya. Tim pertahanan (Tim Biru) tidak harus memiliki pengetahuan tentang cakupan dan durasi latihan, sehingga memberikan penilaian yang lebih realistis tentang bagaimana mereka akan merespons insiden aktual. Karena latihan tim merah bisa bersifat invasif, berhati-hatilah dan terapkan kontrol untuk memverifikasi bahwa latihan tidak menyebabkan kerusakan nyata pada lingkungan Anda.

Pertimbangkan untuk memfasilitasi simulasi siber secara reguler. Setiap jenis latihan dapat memberikan manfaat tersendiri bagi peserta dan organisasi secara keseluruhan, sehingga Anda dapat memilih untuk memulai dengan jenis simulasi yang kurang kompleks (seperti latihan meja)

lalu beralih ke jenis simulasi yang lebih kompleks (latihan Tim Merah). Anda sebaiknya memilih jenis simulasi berdasarkan kematangan keamanan, sumber daya, dan hasil yang Anda inginkan. Beberapa pelanggan mungkin tidak memilih untuk melakukan latihan Tim Merah karena kompleksitas dan biayanya.

Langkah-langkah implementasi

Terlepas dari jenis simulasi yang Anda pilih, simulasi umumnya mengikuti langkah-langkah implementasi berikut ini:

1. Menentukan elemen latihan inti: Tentukan skenario simulasi dan tujuan simulasi. Dua hal ini harus disetujui oleh kepemimpinan.
2. Mengidentifikasi pemangku kepentingan utama: Latihan setidaknya membutuhkan fasilitator dan peserta latihan. Tergantung skenarionya, pemangku kepentingan tambahan seperti pimpinan dari departemen hukum, komunikasi, atau eksekutif dapat dilibatkan.
3. Membangun dan menguji skenario: Skenario mungkin perlu disesuaikan jika elemen tertentu tidak memungkinkan dalam pengembangannya. Tahap ini diharapkan menghasilkan skenario final.
4. Memfasilitasi simulasi: Jenis simulasi menentukan fasilitas yang digunakan (skenario tertulis atau skenario simulasi yang sangat teknis). Fasilitator harus menyelaraskan taktik fasilitasi mereka dengan objek latihan dan harus sebisa mungkin melibatkan semua peserta latihan agar hasilnya bisa optimal.
5. Kembangkan laporan setelah tindakan (AAR): Identifikasi area yang berjalan dengan baik, area yang dapat menggunakan perbaikan, dan potensi kesenjangan. AARHarus mengukur efektivitas simulasi serta respons tim terhadap peristiwa simulasi sehingga kemajuan dapat dilacak dari waktu ke waktu dengan simulasi masa depan.

Sumber daya

Dokumen terkait:

- AWS Panduan Respons Insiden

Video terkait:

- AWS GameDay - Edisi Keamanan

SEC10-BP08 Menetapkan kerangka kerja untuk belajar dari insiden

Menerapkan kerangka kerja pelajaran yang didapatkan dan kemampuan analisis akar masalah tidak hanya dapat membantu Anda meningkatkan kemampuan respons insiden, tetapi juga membantu mencegah berulang kejadian insiden. Dengan belajar dari setiap kejadian, Anda dapat membantu menghindari mengulangi kesalahan, paparan, atau kesalahan konfigurasi yang sama, sehingga tidak hanya meningkatkan postur keamanan Anda, tetapi juga meminimalkan waktu yang hilang untuk situasi yang dapat dicegah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Penting untuk menerapkan kerangka kerja pembelajaran dan meraih poin-poin berikut di tingkatan tinggi:

- Kapan pembelajaran diadakan?
- Apa saja yang terlibat dalam proses pembelajaran tersebut?
- Bagaimana pembelajaran dilakukan?
- Siapa yang terlibat dalam proses tersebut dan bagaimana caranya?
- Bagaimana cara mengenali area yang perlu ditingkatkan?
- Bagaimana Anda memastikan peningkatan dilacak dan diimplementasikan secara efektif?

Kerangka kerja ini tidak boleh fokus pada individu atau menyalahkan individu, tetapi harus fokus pada perbaikan alat dan proses.

Langkah-langkah implementasi

Selain hasil tingkat tinggi yang dicantumkan sebelumnya, penting untuk memastikan bahwa Anda mengajukan pertanyaan yang tepat untuk mendapatkan nilai paling besar (informasi yang mengarah pada perbaikan yang dapat ditindaklanjuti) dari proses tersebut. Pertimbangkan pertanyaan-pertanyaan ini untuk membantu Anda memulai dalam mendorong diskusi pembelajaran Anda:

- Apa insiden yang terjadi?
- Kapan insiden tersebut pertama kali diidentifikasi?
- Bagaimana insiden tersebut diidentifikasi?
- Sistem apa yang memunculkan peringatan tentang aktivitas tersebut?

- Sistem, layanan, dan data apa yang terlibat?
- Secara khusus, apa yang terjadi?
- Apa yang berjalan dengan baik?
- Apa yang tidak berjalan dengan baik?
- Proses atau prosedur mana yang gagal atau tidak dapat diskalakan untuk merespons insiden tersebut?
- Apa yang dapat ditingkatkan dalam bidang berikut:
  - Orang
    - Apakah orang-orang yang perlu dihubungi benar-benar tersedia dan apakah daftar kontak sudah aktual?
    - Apakah orang-orang tidak mendapatkan pelatihan atau tidak memiliki kemampuan yang diperlukan untuk merespons dan menyelidiki insiden tersebut secara efektif?
    - Apakah sumber daya yang sesuai siap dan tersedia?
  - Proses
    - Apakah proses dan prosedur diikuti?
    - Apakah proses dan prosedur didokumentasikan dan tersedia untuk (jenis) insiden ini?
    - Apakah proses dan prosedur yang diperlukan tidak ada?
    - Apakah responden dapat memperoleh akses tepat waktu ke informasi yang diperlukan untuk merespons masalah ini?
  - Teknologi
    - Apakah sistem peringatan yang ada mampu mengidentifikasi dan memperingatkan tentang aktivitas tersebut secara efektif?
    - Bagaimana kita bisa mengurangi time-to-detection 50%?
    - Apakah peringatan yang ada perlu ditingkatkan atau apakah peringatan baru perlu dibangun untuk (jenis) insiden ini?
    - Apakah alat yang ada memungkinkan penyelidikan (pencarian/analisis) insiden yang efektif?
    - Apa yang dapat dilakukan untuk membantu mengidentifikasi (jenis) insiden ini lebih cepat?
    - Apa yang dapat dilakukan untuk membantu mencegah (jenis) insiden ini terjadi lagi?
    - Siapa yang bertanggung jawab atas rencana peningkatan dan bagaimana cara untuk menguji apakah rencana tersebut telah diimplementasikan?
    - Bagaimana garis waktu untuk mengimplementasikan dan menguji pemantauan tambahan atau kontrol dan proses pencegahan?

Daftar ini bukanlah daftar lengkap, melainkan dimaksudkan sebagai titik awal untuk mengidentifikasi kebutuhan organisasi dan bisnis dan bagaimana Anda dapat menganalisisnya agar dapat belajar secara efektif dari insiden dan terus meningkatkan postur keamanan Anda. Yang paling penting adalah memulai dengan memasukkan pembelajaran yang diambil sebagai bagian standar dari proses respons insiden, dokumentasi, dan ekspektasi di seluruh pemangku kepentingan.

Sumber daya

Dokumen terkait:

- Panduan Respons Insiden Keamanan AWS - Menetapkan kerangka kerja untuk belajar dari insiden
- NCSCCAFbimbingan - Pelajaran yang dipetik

## Keamanan aplikasi

Pertanyaan

- SEC11. Bagaimana cara menyertakan dan memvalidasi karakteristik keamanan aplikasi sepanjang siklus hidup desain, pengembangan, dan deployment?

### SEC11. Bagaimana cara menyertakan dan memvalidasi karakteristik keamanan aplikasi sepanjang siklus hidup desain, pengembangan, dan deployment?

Melatih karyawan, menguji menggunakan otomatisasi, memahami dependensi, dan memvalidasi karakteristik keamanan alat dan aplikasi akan membantu mengurangi kemungkinan masalah keamanan dalam beban kerja produksi.

Praktik terbaik

- SEC11-BP01 Train untuk keamanan aplikasi
- SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis
- SEC11-BP03 Lakukan pengujian penetrasi reguler
- SEC11-BP04 Ulasan kode manual
- SEC11-BP05 Memusatkan layanan untuk paket dan dependensi
- SEC11-BP06 Menyebarkan perangkat lunak secara terprogram
- SEC11-BP07 Secara teratur menilai sifat keamanan jaringan pipa

- SEC11-BP08 Membangun program yang menanamkan kepemilikan keamanan dalam tim beban kerja

SEC11-BP01 Train untuk keamanan aplikasi

Berikan pelatihan kepada builder dalam organisasi Anda mengenai praktik umum untuk pengembangan dan pengoperasian aplikasi yang aman. Adopsi praktik pengembangan yang berfokus pada keamanan akan membantu mengurangi kemungkinan munculnya masalah yang hanya terdeteksi pada tahap peninjauan keamanan.

Hasil yang diinginkan: Perangkat lunak harus dirancang dan dibangun dengan mempertimbangkan keamanan. Saat builder di sebuah organisasi berlatih praktik pengembangan aman yang dimulai dengan model ancaman, langkah ini meningkatkan keseluruhan kualitas dan keamanan perangkat lunak yang dibuat. Pendekatan ini dapat mempersingkat waktu untuk mengirimkan perangkat lunak atau fitur karena tidak perlu banyak pengerjaan ulang setelah tahap peninjauan keamanan.

Untuk tujuan praktik terbaik ini, pengembangan aman mengacu pada perangkat lunak yang sedang ditulis dan alat atau sistem yang mendukung siklus hidup pengembangan perangkat lunak ()SDLC.

Anti-pola umum:

- Menunggu sampai peninjauan keamanan, lalu mempertimbangkan karakteristik keamanan sistem.
- Menyerahkan semua keputusan keamanan kepada tim keamanan.
- Gagal mengkomunikasikan bagaimana keputusan yang diambil SDLC terkait dengan harapan keamanan atau kebijakan organisasi secara keseluruhan.
- Terlambat melibatkan diri dalam proses peninjauan keamanan.

Manfaat menjalankan praktik terbaik ini:

- Memiliki pengetahuan yang lebih baik seputar persyaratan organisasi untuk keamanan pada fase awal siklus pengembangan.
- Dapat mengidentifikasi dan mengatasi potensi masalah keamanan lebih cepat, sehingga dapat mengirim fitur lebih cepat.
- Peningkatan kualitas perangkat lunak dan sistem.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sediakan pelatihan kepada builder di organisasi Anda. Memulai dengan kursus pemodelan ancaman adalah sebuah fondasi yang baik untuk membantu melatih keamanan. Idealnya, builder harus dapat mengakses secara mandiri informasi yang relevan dengan beban kerja mereka. Akses ini membantu mereka mengambil keputusan yang tepat tentang karakteristik keamanan sistem yang mereka bangun tanpa perlu bertanya kepada tim lain. Proses melibatkan tim keamanan untuk peninjauan harus diatur dengan jelas dan mudah diikuti. Langkah-langkah di proses peninjauan harus disertakan dalam pelatihan keamanan. Jika tersedia templat atau pola implementasi yang diketahui, keduanya harus mudah dicari dan berhubungan dengan persyaratan keamanan keseluruhan. Pertimbangkan untuk menggunakan AWS CloudFormation, AWS Cloud Development Kit (AWS CDK) Constructs, Service Catalog, atau alat pembuat templat lainnya untuk mengurangi kebutuhan konfigurasi kustom.

Langkah-langkah implementasi

- Mulailah pelatihan builder dengan kursus tentang pemodelan ancaman untuk membangun landasan yang baik, dan bantu latih mereka tentang cara berpikir tentang keamanan.
- Menyediakan akses ke AWS Training dan Sertifikasi, industri, atau pelatihan AWS Mitra.
- Berikan pelatihan terkait proses peninjauan keamanan organisasi Anda, yang menguraikan pembagian tanggung jawab antara tim keamanan, tim beban kerja, dan pemegang kepentingan lainnya.
- Publikasikan panduan layanan mandiri terkait cara memenuhi persyaratan keamanan Anda, termasuk templat dan contoh kode, jika tersedia.
- Dapatkan umpan balik secara rutin dari tim builder terkait pengalaman mereka seputar pelatihan dan proses peninjauan keamanan, dan gunakan umpan balik tersebut untuk meningkatkan kualitasnya.
- Gunakan kampanye game day atau bug bash untuk membantu menurunkan jumlah masalah, dan mengasah kemampuan builder Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC11-BP08 Membangun program yang menanamkan kepemilikan keamanan dalam tim beban kerja

Dokumen terkait:

- AWS Training dan Sertifikasi
- Bagaimana cara berpikir tentang tata kelola keamanan cloud
- Cara melakukan pendekatan terhadap pemodelan ancaman
- Pelatihan akselerasi - The AWS Skills Guild

Video terkait:

- Keamanan proaktif: Pertimbangan dan Pendekatan

Contoh terkait:

- Lokakarya pemodelan ancaman
- Kesadaran industri untuk para pengembang

Layanan terkait:

- AWS CloudFormation
- AWS Cloud Development Kit (AWS CDK) (AWS CDK) Konstruksi
- Katalog Layanan
- AWS BugBust

SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis

Otomatiskan pengujian untuk karakteristik keamanan sepanjang siklus hidup pengembangan dan rilis. Otomatisasi mempermudah identifikasi yang konsisten dan berulang atas potensi masalah dalam perangkat lunak sebelum rilis, yang mengurangi risiko masalah keamanan dalam perangkat lunak yang disediakan.

Hasil yang diinginkan: Tujuan dari pengujian otomatis adalah untuk menyediakan cara terprogram untuk mendeteksi potensi masalah lebih awal dan seringkali sepanjang siklus hidup pengembangan. Saat Anda mengotomatiskan pengujian regresi, Anda dapat menjalankan kembali pengujian fungsional dan nonfungsional untuk memastikan bahwa perangkat lunak yang diuji sebelumnya masih berfungsi seperti yang diharapkan setelah perubahan. Saat Anda mendefinisikan pengujian unit keamanan untuk memeriksa apakah ada kesalahan konfigurasi umum, seperti autentikasi yang rusak atau hilang, Anda dapat mengidentifikasi dan memperbaiki masalah ini lebih dini dalam proses pengembangan.

Otomatisasi pengujian menggunakan kasus pengujian yang dibuat berdasarkan tujuan untuk validasi aplikasi, berdasarkan persyaratan aplikasi dan fungsionalitas yang diinginkan. Hasil pengujian otomatis berdasarkan perbandingan output pengujian yang dibuat dengan output yang diharapkan, sehingga mempercepat keseluruhan siklus hidup pengujian. Metodologi pengujian seperti pengujian regresi dan rangkaian pengujian unit adalah pilihan yang terbaik untuk otomatisasi. Otomatisasi pengujian karakteristik keamanan memungkinkan builder menerima umpan balik otomatis tanpa harus menunggu peninjauan keamanan. Pengujian otomatis dalam bentuk analisis kode statis atau dinamis dapat meningkatkan kualitas kode dan membantu mendeteksi potensi masalah perangkat lunak lebih dini dalam siklus hidup pengembangan.

Anti-pola umum:

- Tidak menyampaikan kasus pengujian dan hasil pengujian dari pengujian otomatis.
- Hanya menjalankan pengujian otomatis segera sebelum rilis.
- Mengotomatiskan kasus pengujian dengan berulang kali mengubah persyaratan.
- Gagal memberikan panduan mengenai cara menangani hasil pengujian keamanan.

Manfaat menjalankan praktik terbaik ini:

- Menurunkan dependensi pada orang yang mengevaluasi karakteristik keamanan sistem.
- Memiliki temuan yang konsisten di beberapa aliran kerja meningkatkan konsisten.
- Menurunkan kemungkinan munculnya masalah keamanan dalam produksi perangkat lunak.
- Periode waktu lebih pendek antara deteksi dan penyelesaian karena mengidentifikasi masalah perangkat lunak lebih dini.
- Meningkatkan visibilitas perilaku sistemik atau berulang di beberapa aliran kerja, yang dapat digunakan untuk mendorong peningkatan berskala organisasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Saat Anda membuat perangkat lunak, adopsi beragam mekanisme untuk pengujian perangkat lunak guna memastikan bahwa Anda menguji aplikasi untuk kedua persyaratan aplikasi, berdasarkan logika bisnis aplikasi, dan persyaratan nonfungsional, yang fokus pada keandalan, performa, dan keamanan aplikasi.

Pengujian keamanan aplikasi statis (SAST) menganalisis kode sumber Anda untuk pola keamanan anomali, dan memberikan indikasi untuk kode rawan cacat. SASTbergantung pada input statis, seperti dokumentasi (spesifikasi persyaratan, dokumentasi desain, dan spesifikasi desain) dan kode sumber aplikasi untuk menguji berbagai masalah keamanan yang diketahui. Penganalisis kode statis dapat membantu mempercepat analisis kode dalam volume besar. Grup NIST Kualitas menyediakan perbandingan Source Code Security Analyzers, yang mencakup alat open source untuk Byte Code Scanner dan Binary Code Scanner.

Lengkapi pengujian statis Anda dengan metodologi pengujian keamanan analisis dinamis (DAST), yang melakukan pengujian terhadap aplikasi yang sedang berjalan untuk mengidentifikasi perilaku yang berpotensi tidak terduga. Pengujian dinamis dapat digunakan untuk mendeteksi potensi masalah yang tidak terdeteksi melalui analisis statis. Pengujian di tahap repositori kode, build, dan pipeline memungkinkan Anda memeriksa berbagai jenis potensi masalah agar tidak masuk ke dalam kode Anda. Amazon CodeWhisperer memberikan rekomendasi kode, termasuk pemindaian keamanan, di pembuatIDE. Amazon CodeGuru Reviewer dapat mengidentifikasi masalah kritis, masalah keamanan, dan hard-to-find bug selama pengembangan aplikasi, dan memberikan rekomendasi untuk meningkatkan kualitas kode.

Lokakarya Security for AWS Developers menggunakan alat pengembang, seperti AWS CodeBuild, AWS CodeCommit, dan AWS CodePipeline, untuk otomatisasi saluran rilis yang mencakup SAST dan metodologi DAST pengujian.

Saat Anda majuSDLC, buat proses berulang yang mencakup tinjauan aplikasi berkala dengan tim keamanan Anda. Umpan balik yang didapatkan dari peninjauan keamanan ini harus diatasi dan divalidasi sebagai bagian dari peninjauan kesiapan rilis Anda. Tinjauan ini membuat postur keamanan aplikasi yang kokoh, dan memberikan umpan balik yang dapat ditindaklanjuti kepada builder untuk menangani potensi masalah.

Langkah-langkah implementasi

- Menerapkan konsistenIDE, tinjauan kode, dan alat CI/CD yang mencakup pengujian keamanan.
- Pertimbangkan di mana yang SDLC tepat untuk memblokir jaringan pipa alih-alih hanya memberi tahu pembangun bahwa masalah perlu diperbaiki.
- Lokakarya Keamanan untuk Pengembang memberikan contoh bagaimana mengintegrasikan pengujian statis dan dinamis ke dalam sebuah pipeline perilisan.
- Melakukan pengujian atau analisis kode menggunakan alat otomatis, seperti Amazon yang CodeWhisperer terintegrasi dengan pengembangIDEs, dan Amazon CodeGuru Reviewer untuk

memindai kode saat komit, membantu pembangun mendapatkan umpan balik pada waktu yang tepat.

- Saat membangun menggunakan AWS Lambda, Anda dapat menggunakan Amazon Inspector untuk memindai kode aplikasi dalam fungsi Anda.
- Saat pengujian otomatis disertakan dalam pipeline CI/CD, Anda harus menggunakan sistem tiket untuk melacak notifikasi dan penyelesaian masalah perangkat lunak.
- Untuk pengujian keamanan yang mungkin menghasilkan temuan, menautkan ke panduan untuk penyelesaian membantu builder meningkatkan kualitas kode.
- Analisis temuan secara berkala dari alat otomatis untuk memprioritaskan otomatisasi berikutnya, pelatihan builder, atau kampanye kesadaran.

Sumber daya

Dokumen terkait:

- Pengiriman Berkelanjutan dan Deployment Berkelanjutan

- AWS DevOps Mitra Kompetensi
- Mitra Kompetensi Keamanan AWS untuk Keamanan Aplikasi
- Memilih pendekatan CI/CD Well-Architected
- Memantau CodeCommit peristiwa di Amazon EventBridge dan Amazon CloudWatch Events
- Deteksi rahasia di Amazon CodeGuru Review
- Mempercepat penerapan AWS dengan tata kelola yang efektif
- Bagaimana pendekatan AWS melakukan otomatisasi deployment secara aman dan otonom

Video terkait:

- Hands-off: Mengotomatiskan pipeline pengiriman berkelanjutan di Amazon
- Mengotomatiskan pipeline CI/CD lintas akun

Contoh terkait:

- Kesadaran industri untuk para pengembang
- AWS CodePipeline Tata Kelola () GitHub

- Lokakarya Keamanan untuk Para Pengembang

SEC11-BP03 Lakukan pengujian penetrasi reguler

Lakukan uji penetrasi perangkat lunak secara teratur. Mekanisme ini membantu mengidentifikasi potensi masalah perangkat lunak yang tidak dapat dideteksi oleh pengujian otomatis atau peninjauan kode manual. Mekanisme ini juga membantu Anda memahami efikasi kontrol-kontrol detektif Anda. Uji penetrasi harus mencoba untuk menentukan apakah perangkat lunak dapat dibuat untuk berkinerja dengan cara-cara yang tak terduga, seperti mengungkapkan data yang seharusnya dilindungi, atau memberikan izin yang lebih luas daripada yang diharapkan.

Hasil yang diinginkan: Pengujian penetrasi digunakan untuk mendeteksi, memulihkan, dan melakukan validasi terhadap properti keamanan aplikasi Anda. Pengujian penetrasi reguler dan terjadwal harus dilakukan sebagai bagian dari siklus hidup pengembangan perangkat lunak ()SDLC. Temuan dari uji penetrasi harus diatasi sebelum perangkat lunak dirilis. Anda harus menganalisis temuan dari uji penetrasi untuk mengidentifikasi apakah ada masalah yang dapat ditemukan menggunakan otomatisasi. Memiliki uji penetrasi yang teratur dan dapat diulangi serta menyertakan mekanisme umpan balik yang aktif membantu menginformasikan panduan kepada builder dan meningkatkan kualitas perangkat lunak.

Anti-pola umum:

- Hanya melakukan uji penetrasi untuk masalah keamanan yang diketahui atau umum.
- Melakukan uji penetrasi aplikasi tanpa pustaka dan alat pihak ketiga yang dependen.
- Hanya melakukan uji penetrasi untuk masalah keamanan paket, dan tidak mengevaluasi logika bisnis yang diimplementasikan.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan kredibilitas karakteristik keamanan dari perangkat lunak sebelum rilis.
- Peluang untuk mengidentifikasi pola aplikasi yang dipilih, yang menghasilkan kualitas perangkat lunak yang lebih baik.
- Loop umpan balik yang mengidentifikasi lebih dini di siklus pengembangan di mana otomatisasi atau pelatihan tambahan dapat meningkatkan karakteristik keamanan perangkat lunak.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Uji penetrasi adalah langkah pengujian keamanan terstruktur untuk menjalankan skenario pelanggaran keamanan terencana guna mendeteksi, menyelesaikan, dan memvalidasi kontrol keamanan. Uji penetrasi dimulai dengan pengintaian, yang mana data dikumpulkan berdasarkan desain aplikasi dan dependensinya saat ini. Daftar kurasi skenario pengujian khusus keamanan dibuat dan dijalankan. Tujuan utama pengujian ini adalah mengungkap masalah keamanan di aplikasi Anda, yang dapat dieksploitasi untuk mendapatkan akses yang tidak direncanakan ke lingkungan Anda, atau akses yang tidak diotorisasi ke data Anda. Anda harus melakukan uji penetrasi saat meluncurkan fitur baru atau setiap kali aplikasi Anda menjalani perubahan besar pada implementasi teknis atau fungsi.

Anda harus mengidentifikasi tahap yang paling sesuai dalam siklus hidup pengembangan untuk melakukan uji penetrasi. Pengujian ini harus dilakukan pada waktu yang hampir mendekati status rilis fungsionalitas sistem yang direncanakan, tetapi ada waktu yang cukup untuk menyelesaikan masalah yang ada.

Langkah-langkah implementasi

- Memiliki proses terstruktur untuk bagaimana pengujian penetrasi dicakup, mendasarkan proses ini pada model ancaman adalah sebuah cara yang baik untuk mempertahankan konteks.
- Identifikasi tempat yang sesuai dalam siklus pengembangan untuk melakukan uji penetrasi. Hal ini harus dilakukan saat ada perubahan minim yang diharapkan pada aplikasi, tetapi ada waktu yang cukup untuk melakukan penyelesaian masalah.
- Latih builder Anda untuk mengetahui apa saja yang diharapkan dari temuan uji penetrasi dan cara mendapatkan informasi dalam penyelesaian.
- Gunakan alat untuk mempercepat alat uji penetrasi dengan mengotomatiskan pengujian yang umum atau dapat diulang.
- Analisis temuan uji penetrasi untuk mengidentifikasi masalah keamanan sistemik, dan gunakan data ini untuk menginformasikan pengujian tambahan yang diotomatisasi dan pendidikan builder yang sedang berlangsung.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC11-BP01 Train untuk keamanan aplikasi

- SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis

Dokumen terkait:

- AWS Pengujian Penetrasi memberikan panduan terperinci untuk pengujian penetrasi AWS
- Mempercepat penerapan AWS dengan tata kelola yang efektif
- Mitra Kompetensi Keamanan AWS
- Modernisasi arsitektur pengujian penetrasi Anda AWS Fargate
- AWS Simulator injeksi kesalahan

Contoh terkait:

- Mengotomatiskan API pengujian dengan AWS CodePipeline () GitHub
- Pembantu keamanan otomatis () GitHub

SEC11-BP04 Ulasan kode manual

Lakukan peninjauan kode manual atas perangkat lunak yang Anda hasilkan. Proses ini membantu memverifikasi bahwa orang yang menulis kode bukan satu-satunya orang yang memeriksa kualitas kode.

Hasil yang diinginkan: Menyertakan langkah peninjauan kode manual selama pengembangan akan meningkatkan kualitas perangkat lunak yang sedang ditulis, dan membantu Anda meningkatkan keterampilan anggota tim yang kurang berpengalaman, dan memberikan kesempatan untuk mengidentifikasi tempat-tempat di mana otomatisasi dapat digunakan. Peninjauan kode manual dapat didukung oleh pengujian dan alat otomatis.

Anti-pola umum:

- Tidak melakukan peninjauan kode sebelum deployment.
- Penulis dan peninjau kode adalah orang yang sama.
- Tidak menggunakan otomatisasi untuk membantu atau mengatur peninjauan kode.
- Tidak melatih builder agar memahami keamanan aplikasi sebelum mereka meninjau kode.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan kualitas kode.
- Peningkatan konsistensi pengembangan kode sepanjang penggunaan ulang pendekatan umum.
- Penurunan jumlah masalah yang ditemukan selama uji penetrasi dan tahap-tahap terakhir.
- Peningkatan transfer ilmu di dalam tim.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Langkah peninjauan harus diimplementasikan sebagai bagian dari keseluruhan alur manajemen kode. Spesifikasinya bergantung pada pendekatan yang digunakan untuk pencabangan, permintaan penarikan, dan penggabungan. Anda mungkin menggunakan AWS CodeCommit atau solusi pihak ketiga seperti GitHub, GitLab, atau Bitbucket. Apa pun metode yang Anda gunakan, penting untuk memastikan bahwa proses Anda memerlukan peninjauan kode sebelum di-deploy di lingkungan produksi. Menggunakan alat seperti Amazon CodeGuru Reviewer dapat mempermudah mengatur proses peninjauan kode.

Langkah-langkah implementasi

- Implementasikan langkah peninjauan manual sebagai bagian dari alur manajemen kode Anda dan lakukan peninjauan ini sebelum melanjutkan.
- Pertimbangkan Amazon CodeGuru Reviewer untuk mengelola dan membantu dalam ulasan kode.
- Implementasikan alur persetujuan yang mengharuskan peninjauan kode selesai sebelum kode dapat lanjut ke tahap berikutnya.
- Pastikan ada proses untuk mengidentifikasi masalah yang ditemukan selama peninjauan kode manual yang dapat dideteksi secara otomatis.
- Integrasikan langkah peninjauan kode manual menggunakan cara yang selaras dengan praktik pengembangan kode Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis

Dokumen terkait:

- Bekerja dengan permintaan tarik di AWS CodeCommit repositori
- Bekerja dengan templat aturan persetujuan di AWS CodeCommit
- Tentang permintaan tarik di GitHub
- Otomatiskan ulasan kode dengan Amazon CodeGuru Reviewer
- Mengotomatiskan deteksi kerentanan keamanan dan bug di pipeline CI/CD menggunakan Amazon Reviewer CodeGuru CLI

Video terkait:

- Peningkatan kualitas kode yang berkelanjutan dengan Amazon CodeGuru

Contoh terkait:

- Lokakarya Keamanan untuk Para Pengembang

SEC11-BP05 Memusatkan layanan untuk paket dan dependensi

Berikan layanan terpusat agar tim builder dapat memperoleh paket perangkat lunak dan dependensi lainnya. Hal ini akan memungkinkan validasi paket sebelum paket disertakan dalam perangkat lunak yang Anda tulis, dan memberikan sumber data untuk analisis perangkat lunak yang digunakan dalam organisasi Anda.

Hasil yang diinginkan: Perangkat lunak terdiri dari satu set paket perangkat lunak lain selain kode yang sedang ditulis. Ini membuatnya mudah untuk mengkonsumsi implementasi fungsionalitas yang berulang kali digunakan, seperti JSON parser atau pustaka enkripsi. Memusatkan secara logis sumber untuk paket dan dependensi ini memberikan mekanisme bagi tim keamanan untuk memvalidasi karakteristik paket sebelum paket digunakan. Pendekatan ini juga menurunkan risiko masalah tidak terduga yang disebabkan oleh perubahan dalam paket yang ada, atau oleh tim builder, termasuk paket arbitrer langsung dari internet. Gunakan pendekatan ini sehubungan dengan alur pengujian manual dan otomatis untuk meningkatkan kredibilitas kualitas perangkat lunak yang sedang Anda kembangkan.

Anti-pola umum:

- Menarik paket dari repositori arbitrer di internet.
- Tidak menguji paket baru sebelum menyediakannya kepada builder.

Manfaat menjalankan praktik terbaik ini:

- Pemahaman lebih baik mengenai paket apa yang digunakan di perangkat lunak yang sedang dibangun.
- Dapat memberi tahu tim beban kerja saat paket perlu diperbarui berdasarkan pemahaman siapa yang menggunakan apa.
- Menurunkan risiko terjadinya penyertaan paket bermasalah di perangkat lunak Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sediakan layanan tersentralisasi untuk paket dan dependensi dengan cara yang mudah digunakan bagi builder. Layanan tersentralisasi dapat dipusatkan secara logis daripada diimplementasikan sebagai sistem monolitik. Pendekatan ini memungkinkan Anda menyediakan layanan dengan cara yang memenuhi kebutuhan builder Anda. Anda harus menerapkan cara yang efisien untuk menambahkan paket ke repositori ketika pembaruan terjadi atau persyaratan baru muncul. AWS layanan seperti AWS CodeArtifactatau solusi AWS mitra serupa menyediakan cara untuk memberikan kemampuan ini.

Langkah-langkah implementasi:

- Implementasikan layanan repositori tersentralisasi secara logis yang tersedia di semua lingkungan tempat perangkat lunak dikembangkan.
- Sertakan akses ke repositori sebagai bagian dari proses vending Akun AWS .
- Buat otomatisasi untuk menguji paket sebelum paket dipublikasikan ke sebuah repositori.
- Pertahankan metrik paket yang paling sering digunakan, bahasa, dan tim dengan jumlah perubahan tertinggi.
- Sediakan mekanisme otomatis untuk tim builder guna meminta paket baru dan memberikan umpan balik.
- Pindai paket secara rutin di repositori Anda untuk mengidentifikasi adanya potensi dampak masalah yang baru ditemukan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis

Dokumen terkait:

- Mempercepat penerapan AWS dengan tata kelola yang efektif
- Kencangkan keamanan paket Anda dengan toolkit CodeArtifact Package Origin Control
- Mendeteksi masalah keamanan saat masuk dengan Amazon Reviewer CodeGuru
- Tingkat rantai pasokan untuk Artefak Perangkat Lunak () SLSA

Video terkait:

- Keamanan proaktif: Pertimbangan dan Pendekatan
- Filosofi AWS Keamanan (re:Invent 2017)
- Ketika keamanan, keselamatan, dan urgensi semuanya penting: Menangani Log4Shell

Contoh terkait:

- Pipa Penerbitan Package Multi Region (GitHub)
- Menerbitkan Modul Node.js pada AWS CodeArtifact penggunaan AWS CodePipeline (GitHub)
- AWS CDK Contoh CodeArtifact Pipa Java (GitHub)
- Mendistribusikan pribadi. NET NuGet paket dengan AWS CodeArtifact (GitHub)

SEC11-BP06 Menyebarkan perangkat lunak secara terprogram

Lakukan deployment perangkat lunak secara terprogram jika memungkinkan. Pendekatan ini mengurangi kemungkinan terjadinya kegagalan deployment atau masalah tak terduga karena kesalahan manusia.

Hasil yang diinginkan: Menjauhkan orang-orang dari data adalah prinsip utama untuk melakukan pembangunan dengan aman di AWS Cloud. Prinsip ini termasuk cara Anda melakukan deployment pada perangkat lunak Anda.

Dengan tidak bergantung pada orang untuk men-deploy perangkat lunak, Anda akan mendapatkan manfaat peningkatan kredibilitas bahwa apa yang Anda uji adalah apa yang di-deploy, dan deployment dilakukan secara konsisten setiap kali dijalankan. Perangkat lunak tidak perlu diubah agar berfungsi di lingkungan yang berbeda. Menggunakan prinsip pengembangan aplikasi dua

belas faktor, terutama eksternalisasi konfigurasi, memungkinkan Anda men-deploy kode yang sama ke beberapa lingkungan tanpa memerlukan perubahan. Menandatangani paket perangkat lunak secara kriptografis adalah cara yang baik untuk memastikan bahwa tidak ada yang berubah di antara lingkungan. Keseluruhan hasil dari pendekatan ini adalah penurunan risiko proses perubahan dan peningkatan konsistensi rilis perangkat lunak.

Anti-pola umum:

- Men-deploy perangkat lunak secara manual ke tahap produksi.
- Melakukan perubahan secara manual ke perangkat lunak agar dapat menyesuaikan dengan lingkungan yang berbeda.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan kredibilitas dalam proses rilis perangkat lunak.
- Penurunan risiko kegagalan perubahan yang berdampak pada fungsionalitas bisnis.
- Peningkatan jadwal rilis karena risiko terhadap perubahan lebih rendah.
- Kapabilitas pengembalian (rollback) otomatis untuk peristiwa tidak terduga selama deployment.
- Kemampuan untuk membuktikan secara kriptografis bahwa perangkat lunak yang diuji adalah perangkat lunak yang di-deploy.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Bangun Akun AWS struktur Anda untuk menghapus akses manusia yang persisten dari lingkungan dan gunakan alat CI/CD untuk melakukan penerapan. Rancang arsitektur aplikasi Anda sehingga data konfigurasi khusus lingkungan diperoleh dari sumber eksternal, seperti AWS Systems Manager Parameter Store. Tanda tangani paket setelah paket diuji, dan validasikan tanda tangan ini selama deployment. Konfigurasikan pipeline CI/CD Anda untuk mendorong kode aplikasi dan menggunakan canary untuk mengonfirmasi deployment yang berhasil. Gunakan alat-alat seperti AWS CloudFormation atau AWS CDK untuk menentukan infrastruktur Anda, dan kemudian gunakan AWS CodeBuild dan AWS CodePipeline untuk melakukan operasi CI/CD.

Langkah-langkah implementasi

- Bangun pipeline CI/CD yang ditetapkan dengan baik untuk menyederhanakan proses deployment.

- Menggunakan AWS CodeBuild dan AWS Code Pipeline untuk menyediakan kemampuan CI/CD akan membuat Anda lebih mudah dalam mengintegrasikan pengujian keamanan ke dalam pipeline Anda.
- Ikuti panduan tentang pemisahan lingkungan di whitepaper Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun.
- Pastikan tidak ada akses manusia yang persisten ke lingkungan tempat beban kerja produksi berjalan.
- Rancang arsitektur aplikasi Anda untuk mendukung eksternalisasi data konfigurasi.
- Pertimbangkan untuk men-deploy dengan menggunakan model deployment blue/green.
- Implementasikan canary untuk memvalidasi deployment perangkat lunak yang berhasil.
- Gunakan alat-alat kriptografi seperti AWS Signer atau AWS Key Management Service (AWS KMS) untuk menandatangani dan memverifikasi paket perangkat lunak yang Anda deploy.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis

Dokumen terkait:

- Lokakarya CI/CD AWS
- Mempercepat penerapan AWS dengan tata kelola yang efektif
- Melakukan otomatisasi deployment secara aman dan otonom
- Penandatanganan kode menggunakan AWS Certificate Manager Private CA dan kunci AWS Key Management Service asimetris
- Penandatanganan Kode, Kontrol Kepercayaan dan Integritas untuk AWS Lambda

Video terkait:

- Hands-off: Mengotomatiskan pipeline pengiriman berkelanjutan di Amazon

Contoh terkait:

- Penerapan Biru/Hijau dengan AWS Fargate

SEC11-BP07 Secara teratur menilai sifat keamanan jaringan pipa

Terapkan prinsip-prinsip Pilar Keamanan Well-Architected pada pipeline Anda, dengan perhatian khusus pada pemisahan izin. Nilai karakteristik keamanan infrastruktur pipeline Anda secara teratur. Mengelola keamanan dari pipeline secara efektif akan memungkinkan Anda untuk memberikan keamanan perangkat lunak yang lolos melalui pipeline.

Hasil yang diinginkan: Saluran pipeline yang digunakan untuk melakukan pembangunan dan deployment perangkat lunak Anda harus mengikuti praktik yang direkomendasikan yang sama seperti beban kerja lainnya di lingkungan Anda. Pengujian yang diimplementasikan di pipeline seharusnya tidak dapat diedit oleh builder yang menggunakannya. Pipeline seharusnya hanya memiliki izin yang diperlukan untuk deployment yang dilakukannya dan harus mengimplementasikan perlindungan untuk menghindari deployment ke lingkungan yang salah. Pipeline tidak boleh bergantung pada kredensial jangka panjang, dan harus dikonfigurasikan untuk memberikan status sehingga integritas lingkungan build dapat divalidasi.

Anti-pola umum:

- Pengujian keamanan yang dapat dilewati oleh builder.
- Izin yang terlalu luas untuk pipeline deployment.
- Pipeline tidak dikonfigurasikan untuk memvalidasi input.
- Tidak rutin meninjau izin yang terkait dengan infrastruktur CI/CD Anda.
- Penggunaan kredensial jangka panjang atau yang diberi hardcode.

Manfaat menjalankan praktik terbaik ini:

- Kredibilitas lebih tinggi pada integritas perangkat lunak yang dibangun dan di-deploy melalui pipeline.
- Kemampuan untuk menghentikan deployment saat ada aktivitas yang mencurigakan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Dimulai dengan layanan CI/CD terkelola yang mendukung IAM peran mengurangi risiko kebocoran kredenal. Menerapkan prinsip-prinsip Pilar Keamanan ke infrastruktur pipeline CI/CD Anda dapat membantu Anda menentukan titik mana yang dapat ditingkatkan keamanannya. Mengikuti Arsitektur Referensi Pipeline Deployment AWS adalah titik awal yang baik untuk membangun lingkungan CI/CD

Anda. Meninjau secara rutin implementasi pipeline dan menganalisis log untuk menemukan perilaku tidak terduga dapat membantu Anda memahami pola penggunaan pipeline yang digunakan untuk men-deploy perangkat lunak.

Langkah-langkah implementasi

- Mulailah dengan Arsitektur Referensi Pipeline Deployment AWS.
- Pertimbangkan untuk menggunakan AWS IAMAccess Analyzer untuk secara terprogram menghasilkan IAM kebijakan hak istimewa paling sedikit untuk saluran pipa.
- Integrasikan saluran pipa Anda dengan pemantauan dan peringatan sehingga Anda diberi tahu tentang aktivitas yang tidak terduga atau tidak normal, untuk AWS layanan terkelola Amazon EventBridge memungkinkan Anda merutekan data ke target seperti atau AWS LambdaAmazon Simple Notification Service (Amazon). SNS

Sumber daya

Dokumen terkait:

- Arsitektur Referensi Pipeline Deployment AWS
- Memantau AWS CodePipeline
- Praktik terbaik keamanan untuk AWS CodePipeline

Contoh terkait:

- DevOpsdasbor pemantauan (GitHub)

SEC11-BP08 Membangun program yang menanamkan kepemilikan keamanan dalam tim beban kerja

Buat program atau mekanisme yang memberdayakan tim builder untuk membuat keputusan keamanan tanpa perangkat lunak yang mereka buat. Tim keamanan Anda masih harus memvalidasi keputusan ini selama peninjauan, tetapi menanamkan kepemilikan keamanan dalam tim builder memungkinkan beban kerja dibangun dengan lebih cepat dan lebih aman. Mekanisme ini juga mendukung budaya kepemilikan yang secara positif memengaruhi operasi sistem yang Anda buat.

Hasil yang diinginkan: Untuk menyematkan kepemilikan keamanan dan pengambilan keputusan dalam tim builder, Anda dapat melatih builder tentang cara berpikir tentang keamanan atau Anda

dapat meningkatkan pelatihan mereka dengan orang-orang keamanan yang disematkan atau terkait dengan tim builder. Pendekatan mana pun bisa dilakukan dan memungkinkan tim mengambil keputusan keamanan yang lebih berkualitas pada awal siklus pengembangan. Model kepemilikan ini didasarkan pada pelatihan untuk keamanan aplikasi. Memulai dengan model ancaman untuk beban kerja tertentu akan membantu fokus pada pemikiran desain dalam konteks yang sesuai. Manfaat lain dalam memiliki komunitas builder yang fokus pada keamanan, atau kelompok rekayasawan keamanan yang bekerja sama dengan tim builder, adalah pemahaman yang lebih mendalam mengenai cara perangkat lunak ditulis. Pemahaman ini membantu Anda menentukan area peningkatan selanjutnya dalam kemampuan otomatisasi Anda.

Anti-pola umum:

- Menyerahkan semua keputusan desain keamanan kepada tim keamanan.
- Tidak menangani persyaratan keamanan cukup dini dalam proses pengembangan.
- Tidak memperoleh umpan balik dari builder dan orang keamanan dalam pengoperasian program.

Manfaat menjalankan praktik terbaik ini:

- Waktu penyelesaian peninjauan keamanan lebih cepat.
- Penurunan masalah keamanan yang hanya terdeteksi pada tahap peninjauan keamanan.
- Peningkatan keseluruhan kualitas perangkat lunak yang sedang ditulis.
- Peluang untuk mengidentifikasi dan memahami masalah-masalah sistemik atau area peningkatan bernilai tinggi.
- Penurunan jumlah pengerjaan ulang yang diperlukan akibat adanya temuan dari peninjauan keamanan.
- Peningkatan persepsi fungsi keamanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Mulailah dengan panduan di SEC11-BP01 Train untuk keamanan aplikasi. Lalu, identifikasi model operasional untuk program yang menurut Anda paling sesuai dengan organisasi Anda. Dua pola utamanya adalah melatih builder atau menyertakan orang keamanan ke dalam tim builder. Setelah Anda memutuskan pendekatan awal yang akan digunakan, Anda harus melakukan uji coba dengan satu grup atau grup kecil tim beban kerja untuk membuktikan model mana yang sesuai dengan

organisasi Anda. Dukungan kepemimpinan dari bagian builder dan keamanan organisasi membantu pengiriman dan kesuksesan program. Saat Anda membangun program ini, penting untuk memilih metrik yang dapat digunakan untuk menunjukkan nilai program. Belajar dari bagaimana AWS mendekati masalah ini adalah pengalaman belajar yang baik. Praktik terbaik ini sangat berfokus pada budaya dan perubahan organisasi. Alat yang Anda gunakan harus mendukung kolaborasi antara komunitas keamanan dan builder.

Langkah-langkah implementasi

- Mulai dengan melatih builder Anda untuk memahami keamanan aplikasi.
- Buat komunitas dan program orientasi untuk mengedukasi builder.
- Pilih nama untuk program. Pelindung, Jawara, atau Pendukung adalah nama yang sering digunakan.
- Identifikasi model yang akan digunakan: latih builder, sertakan rekayasawan keamanan, atau miliki peran keamanan afinitas.
- Identifikasi sponsor proyek dari grup keamanan, builder, dan grup lain yang berpotensi.
- Lacak metrik untuk jumlah orang yang terlibat dalam program, waktu yang dihabiskan untuk peninjauan, dan umpan balik dari orang keamanan dan builder. Gunakan metrik-metrik ini untuk membuat peningkatan.

Sumber daya

Praktik-praktik terbaik terkait:

- SEC11-BP01 Train untuk keamanan aplikasi
- SEC11-BP02 Mengotomatiskan pengujian selama siklus hidup pengembangan dan rilis

Dokumen terkait:

- Cara melakukan pendekatan terhadap pemodelan ancaman
- Bagaimana cara berpikir tentang tata kelola keamanan cloud

Video terkait:

- Keamanan proaktif: Pertimbangan dan Pendekatan

# Keandalan

Pilar Keandalan berkenaan dengan kemampuan beban kerja untuk menjalankan fungsinya dengan benar dan konsisten sesuai dengan yang diharapkan. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keandalan.

Area praktik terbaik

- Fondasi
- Arsitektur beban kerja
- Manajemen perubahan
- Manajemen kegagalan

## Fondasi

Pertanyaan

- REL1. Bagaimana cara mengelola Kuota Layanan (Service Quotas) dan batasan?
- REL2. Bagaimana cara merencanakan topologi jaringan Anda?

### REL1. Bagaimana cara mengelola Kuota Layanan (Service Quotas) dan batasan?

Untuk arsitektur beban kerja berbasis cloud, ada Service Quotas (kuota layanan) (yang juga disebut sebagai batas layanan). Kuota ini ada untuk mencegah penyediaan sumber daya lebih dari yang Anda butuhkan secara tidak sengaja dan untuk membatasi tarif permintaan pada API operasi sehingga dapat melindungi layanan dari penyalahgunaan. Ada juga kendala-kendala sumber daya, misalnya, laju Anda dapat mendorong bit di kabel serat optik, atau jumlah penyimpanan di diska secara fisik.

Praktik terbaik

- REL01-BP01 Sadar akan kuota dan kendala layanan
- REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah
- REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur
- REL01-BP04 Memantau dan mengelola kuota
- REL01-BP05 Mengotomatiskan manajemen kuota

- REL01-BP06 Pastikan ada kesenjangan yang cukup antara kuota saat ini dan penggunaan maksimum untuk mengakomodasi failover

REL01-BP01 Sadar akan kuota dan kendala layanan

Perhatikan kuota default Anda dan kelola permintaan penambahan kuota untuk arsitektur beban kerja Anda. Ketahui kendala sumber daya cloud mana, seperti disk atau jaringan, yang berpotensi memberi dampak.

Hasil yang diinginkan: Pelanggan dapat mencegah degradasi atau gangguan layanan Akun AWS dengan menerapkan pedoman yang tepat untuk memantau metrik utama, tinjauan infrastruktur, dan langkah-langkah remediasi otomatisasi untuk memverifikasi bahwa kuota dan kendala layanan tidak tercapai yang dapat menyebabkan degradasi atau gangguan layanan.

Anti-pola umum:

- Melakukan deployment beban kerja tanpa memahami kuota keras dan lunak serta batasan-batasannya untuk layanan yang digunakan.
- Melakukan deployment beban kerja pengganti tanpa menganalisis dan mengonfigurasi ulang kuota yang diperlukan atau menghubungi tim Dukungan sebelumnya.
- Berasumsi bahwa layanan cloud tidak memiliki batasan dan layanan dapat digunakan tanpa mempertimbangkan angka permintaan, batas, hitungan, dan kuantitas.
- Berasumsi bahwa kuota akan ditingkatkan secara otomatis.
- Tidak mengetahui proses dan lini waktu permintaan kuota.
- Berasumsi bahwa kuota layanan (service quotas) cloud default selalu sama untuk setiap layanan di semua wilayah.
- Berasumsi bahwa kendala layanan dapat ditembus dan sistem akan diskalakan secara otomatis dan meningkatkan batas di luar kendala sumber daya
- Tidak menguji aplikasi saat lalu lintas memuncak untuk membebani pemanfaatan sumber dayanya.
- Melakukan penyediaan sumber daya tanpa melakukan analisis terhadap ukuran sumber daya yang diperlukan.
- Melakukan penyediaan berlebihan dalam hal kapasitas dengan memilih jenis sumber daya yang jauh melampaui kebutuhan riil atau perkiraan lalu lintas puncak.
- Tidak menilai persyaratan kapasitas untuk tingkat lalu lintas yang baru sebelum adanya peristiwa pelanggan baru atau deployment teknologi baru.

Manfaat menerapkan praktik terbaik ini: Pemantauan dan manajemen otomatis kuota layanan (service quotas) dan kendala sumber daya dapat mengurangi kegagalan secara proaktif. Perubahan pada pola lalu lintas untuk layanan pelanggan dapat menyebabkan terjadinya gangguan atau penurunan kualitas jika praktik terbaik tidak diikuti. Dengan memantau dan mengelola nilai-nilai ini di semua wilayah dan semua akun, aplikasi dapat memiliki ketahanan yang lebih baik saat terjadi peristiwa yang tidak direncanakan atau tidak diinginkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Service Quotas adalah AWS layanan yang membantu Anda mengelola kuota untuk lebih dari 250 AWS layanan dari satu lokasi. Seiring dengan mencari nilai kuota, Anda juga dapat meminta dan melacak kenaikan kuota dari konsol Service Quotas atau menggunakan. AWS SDK AWS Trusted Advisor menawarkan pemeriksaan kuota layanan yang menampilkan penggunaan dan kuota Anda untuk beberapa aspek dari beberapa layanan. Kuota layanan default per layanan juga ada dalam AWS dokumentasi per layanan masing-masing (misalnya, lihat VPCKuota Amazon).

Beberapa batasan layanan, seperti batas tarif pada throttled APIs ditetapkan dalam Amazon API Gateway itu sendiri dengan mengonfigurasi paket penggunaan. Beberapa batasan yang ditetapkan sebagai konfigurasi pada layanan masing-masing termasuk Provisioned, RDS penyimpanan IOPS Amazon dialokasikan, dan alokasi volume AmazonEBS. Amazon Elastic Compute Cloud memiliki dasbor batas layanannya sendiri yang akan dapat membantu Anda mengelola instans, Amazon Elastic Block Store, dan batas-batas alamat IP Elastic. Jika Anda memiliki kasus penggunaan di mana kuota layanan memengaruhi kinerja aplikasi Anda dan tidak dapat disesuaikan dengan kebutuhan Anda, hubungi AWS Support untuk melihat apakah ada mitigasi.

Kuota layanan (service quotas) bisa menurut Wilayah atau bersifat global. Menggunakan AWS layanan yang mencapai kuota tidak akan bertindak seperti yang diharapkan dalam penggunaan normal dan dapat menyebabkan gangguan atau penurunan layanan. Misalnya, kuota layanan membatasi jumlah EC2 instans Amazon DL yang digunakan di Wilayah. Batas tersebut dapat dicapai selama peristiwa penskalaan lalu lintas menggunakan grup ASG Auto Scaling ().

Kuota layanan (service quotas) untuk setiap akun harus dinilai secara rutin dalam hal penggunaan untuk menentukan batas layanan yang tepat untuk akun tersebut. Kuota layanan (service quotas) ini dibuat sebagai pagar pembatas operasional, agar Anda tidak melakukan pengadaan sumber daya lebih dari yang dibutuhkan tanpa disadari. Mereka juga berfungsi untuk membatasi tingkat permintaan pada API operasi untuk melindungi layanan dari penyalahgunaan.

Kendala layanan berbeda dari kuota layanan (service quotas). Kendala-kendala layanan mewakili batasan sumber daya tertentu yang ditentukan oleh jenis sumber daya tersebut. Ini mungkin kapasitas penyimpanan (misalnya, gp2 memiliki batas ukuran 1 GB - 16 TB) atau throughput disk. Sangat penting untuk merancang dan terus menilai kendala jenis sumber daya untuk penggunaan yang mungkin mencapai batasnya. Jika suatu kendala tercapai di luar perkiraan, maka aplikasi dan layanan akun dapat mengalami gangguan atau mengalami penurunan kualitas.

Jika ada kasus penggunaan di mana kuota layanan memengaruhi kinerja aplikasi dan tidak dapat disesuaikan dengan kebutuhan yang diperlukan, hubungi AWS Support untuk melihat apakah ada mitigasi. Untuk detail lebih lanjut tentang cara menyesuaikan kuota tetap, lihat REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur.

Ada sejumlah AWS layanan dan alat untuk membantu memantau dan mengelola Service Quotas. Layanan dan alat harus dimanfaatkan untuk menyediakan pemeriksaan level kuota otomatis atau manual.

- AWS Trusted Advisor menawarkan pemeriksaan kuota layanan yang menampilkan penggunaan dan kuota Anda untuk beberapa aspek dari beberapa layanan. Alat ini dapat membantu Anda mengidentifikasi layanan yang sudah mendekati kuota.
- AWS Management Console menyediakan metode untuk menampilkan nilai kuota layanan, mengelola, meminta kuota baru, memantau status permintaan kuota, dan menampilkan riwayat kuota.
- AWS CLI dan CDKs menawarkan metode terprogram untuk secara otomatis mengelola dan memantau tingkat kuota layanan dan penggunaan.

Langkah-langkah implementasi

Untuk Service Quotas:

- Tinjau AWS Service Quotas.
- Untuk mengetahui kuota layanan yang ada, tentukan layanan (seperti IAM Access Analyzer) yang digunakan. Ada sekitar 250 AWS layanan yang dikendalikan oleh kuota layanan. Lalu, tentukan nama kuota layanan (service quotas) tertentu yang dapat digunakan di dalam setiap akun dan Wilayah. Terdapat sekitar 3000 nama kuota layanan (service quotas) per Wilayah.
- Tambahkan analisis kuota ini dengan AWS Config untuk menemukan semua AWS sumber daya yang digunakan dalam Anda. Akun AWS

- Gunakan AWS CloudFormation data untuk menentukan AWS sumber daya yang Anda gunakan. Lihatlah sumber daya yang dibuat baik di AWS Management Console atau dengan `list-stack-resources` AWS CLI perintah. Anda juga dapat melihat sumber daya yang dikonfigurasi untuk di-deploy di templat itu sendiri.
- Tentukan semua layanan yang diperlukan oleh beban kerja Anda dengan melihat kode deployment.
- Tentukan kuota layanan (service quotas) yang berlaku. Gunakan informasi yang dapat diakses secara terprogram dari Trusted Advisor dan Service Quotas.
- Tetapkan metode pemantauan otomatis (lihat REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah dan REL01-BP04 Memantau dan mengelola kuota) untuk memperingatkan dan menginformasikan apakah kuota layanan (service quotas) sudah dekat atau telah mencapai batasnya.
- Tetapkan metode otomatis dan terprogram untuk memeriksa apakah kuota layanan (service quotas) telah diubah di satu wilayah tetapi tidak di wilayah lain dalam akun yang sama (lihat REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah dan REL01-BP04 Memantau dan mengelola kuota).
- Lakukan otomatisasi terhadap pemindaian log dan metrik aplikasi untuk menentukan apakah terdapat kesalahan kuota atau kendala layanan yang terjadi. Jika terdapat kesalahan, kirimkan peringatan ke sistem pemantauan.
- Tetapkan prosedur teknik untuk menghitung perubahan kuota yang diperlukan (lihat REL01-BP05 Mengotomatiskan manajemen kuota) setelah diidentifikasi bahwa kuota yang lebih besar diperlukan untuk layanan tertentu.
- Buat alur kerja pengadaan dan persetujuan untuk meminta perubahan dalam kuota layanan (service quotas). Sertakan di dalamnya alur kerja pengecualian untuk mengantisipasi jika permintaan ditolak atau disetujui sebagian.
- Buat metode teknik untuk meninjau kuota layanan sebelum penyediaan dan menggunakan AWS layanan baru sebelum diluncurkan ke lingkungan produksi atau dimuat. (misalnya, memuat akun pengujian).

Untuk kendala layanan:

- Bangun metode pemantauan dan metrik untuk memberi peringatan jika terdapat pembacaan sumber daya yang mendekati kendala sumber dayanya. Manfaatkan yang CloudWatch sesuai untuk metrik atau pemantauan log.

- Tetapkan ambang batas peringatan untuk masing-masing sumber daya yang memiliki kendala yang dapat berpengaruh terhadap aplikasi atau sistem.
- Ciptakan prosedur manajemen alur kerja dan infrastruktur untuk mengubah jenis sumber daya jika pemanfaatan mendekati kendala. Alur kerja ini harus mencakup pengujian beban sebagai sebuah praktik terbaik untuk memverifikasi bahwa jenis sumber daya baru tersebut sudah tepat dengan kendala-kendala baru.
- Migrasikan sumber daya yang diidentifikasi ke jenis sumber daya baru yang disarankan, dengan menggunakan prosedur dan proses yang ada.

Sumber daya

Praktik-praktik terbaik terkait:

- REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah
- REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur
- REL01-BP04 Memantau dan mengelola kuota
- REL01-BP05 Mengotomatiskan manajemen kuota
- REL01-BP06 Pastikan ada kesenjangan yang cukup antara kuota saat ini dan penggunaan maksimum untuk mengakomodasi failover
- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan

Dokumen terkait:

- AWS Pilar Keandalan Well-Architected Framework: Ketersediaan
- AWS Service Quotas (sebelumnya disebut sebagai batas layanan)
- AWS Trusted Advisor Pemeriksaan Praktik Terbaik (lihat bagian Batas Layanan)
- AWS batasi monitor pada AWS jawaban
- Batas EC2 Layanan Amazon
- Apa itu Service Quotas?
- Cara Meminta Peningkatan Kuota

- Titik akhir dan kuota layanan (service quotas)
- Panduan Pengguna Service Quotas
- Monitor Kuota untuk AWS
- AWS Batas Isolasi Kesalahan
- Ketersediaan dengan redundansi
- AWS untuk Data
- Apa itu Integrasi Berkelanjutan?
- Apa itu Pengiriman Berkelanjutan?
- APNMitra: mitra yang dapat membantu manajemen konfigurasi
- Mengelola siklus hidup akun di lingkungan SaaS account-per-tenant pada AWS
- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Lihat AWS Trusted Advisor rekomendasi dalam skala besar dengan AWS Organizations
- Mengotomatisasi Peningkatan Batas Layanan dan Dukungan Perusahaan dengan AWS Control Tower

Video terkait:

- AWS Live re: Inforce 2019 - Service Quotas
- Melihat dan Mengelola Kuota untuk AWS Layanan Menggunakan Service Quotas
- AWS IAMKuota Demo

Alat terkait:

- CodeGuru Peninjau Amazon
- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- DevOpsGuru Amazon
- AWS Config
- AWS Trusted Advisor
- AWS CDK

- AWS Systems Manager
- AWS Marketplace

REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah

Jika Anda menggunakan beberapa akun atau Wilayah, minta kuota yang sesuai di semua lingkungan tempat beban kerja produksi Anda dijalankan.

Hasil yang diinginkan: Layanan dan aplikasi tidak boleh terpengaruh oleh kehabisan kuota layanan (service quotas) untuk konfigurasi yang menjangkau akun-akun atau Wilayah atau yang memiliki desain ketahanan dengan menggunakan zona, Wilayah, atau failover akun.

Anti-pola umum:

- Membiarkan penggunaan sumber daya di satu Wilayah terisolasi untuk terus bertambah, tanpa mekanisme untuk mempertahankan kapasitas di wilayah-wilayah lainnya.
- Mengatur semua kuota di Wilayah isolasi secara manual dan independen.
- Tidak mempertimbangkan efek arsitektur ketahanan (seperti aktif atau pasif) dalam kebutuhan kuota masa depan saat terjadi penurunan kualitas di dalam Wilayah non-primer.
- Tidak melakukan evaluasi kuota secara rutin dan membuat perubahan yang diperlukan di setiap Wilayah dan akun tempat beban kerja dijalankan.
- Tidak memanfaatkan templat permintaan kuota untuk meminta peningkatan di beberapa Wilayah dan akun.
- Tidak memperbarui kuota layanan (service quotas) disebabkan pemikiran yang tidak tepat bahwa peningkatan kuota memiliki dampak biaya seperti permintaan pemesanan komputasi.

Manfaat menerapkan praktik terbaik ini: Memverifikasi bahwa Anda dapat menangani beban Anda saat ini di wilayah sekunder atau akun jika layanan regional tidak tersedia. Hal ini dapat membantu Anda mengurangi jumlah kesalahan atau tingkat penurunan kualitas yang terjadi selama kerugian wilayah (region loss).

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kuota layanan (service quotas) dilacak per akun. Kecuali dinyatakan lain, setiap kuota adalah Wilayah AWS-spesifik. Selain lingkungan produksi, kelola juga kuota yang ada di semua

lingkungan non-produksi yang berlaku agar pengujian dan pengembangan tidak terhambat. Untuk mempertahankan tingkat ketahanan yang tinggi diperlukan penilaian kuota layanan (service quotas) secara kontinu (baik otomatis maupun manual).

Dengan lebih banyak beban kerja yang mencakup Wilayah karena penerapan desain dengan menggunakan pendekatan Active/Active, Active/Passive – Hot, Active/Passive – Cold, dan Active/Passive – Pilot Light, penting untuk memahami semua tingkat kuota Wilayah dan akun. Pola lalu lintas terdahulu tidak selalu menjadi indikator yang tepat bahwa kuota layanan (service quotas) diatur dengan benar.

Sama pentingnya, batas nama kuota layanan (service quotas) tidak selalu sama untuk setiap Wilayah. Di satu Wilayah, nilainya bisa jadi lima, sedangkan di wilayah lain nilainya bisa jadi sepuluh. Pengelolaan atas kuota-kuota ini harus meliputi semua layanan, akun, dan Wilayah yang sama untuk menyediakan ketahanan yang konsisten saat beban meningkat.

Sesuaikan semua perbedaan kuota layanan (service quotas) di semua Wilayah yang berbeda (Wilayah Aktif atau Wilayah Pasif) dan buat proses untuk menyesuaikan semua perbedaan tersebut secara kontinu. Rencana pengujian failover Wilayah pasif sangat jarang diskalakan ke kapasitas aktif puncak, sehingga kegiatan game day atau table top bisa gagal menemukan perbedaan kuota layanan (service quotas) antar-Wilayah dan juga mempertahankan batas-batas yang tepat.

Penyimpangan kuota layanan (service quotas), kondisi di mana batas kuota layanan (service quotas) untuk kuota bernama tertentu diubah dalam satu Wilayah dan tidak semua Wilayah, sangat penting untuk dilacak dan dilakukan evaluasi. Anda harus mempertimbangkan untuk mengubah kuota di Wilayah yang memiliki lalu lintas atau yang berpotensi mendatangkan lalu lintas.

- Pilih Wilayah dan akun yang relevan berdasarkan persyaratan layanan, latensi, peraturan, dan pemulihan bencana (DR) Anda.
- Identifikasikan kuota layanan (service quotas) di semua akun, Wilayah, dan Zona Ketersediaan yang relevan. Batasannya mencakup akun dan Wilayah. Nilai-nilai ini harus dibandingkan untuk mengetahui perbedaannya.

Langkah-langkah implementasi

- Lakukan peninjauan atas nilai Service Quotas yang mungkin telah melampaui level risiko penggunaan a. AWS Trusted Advisor menyediakan peringatan untuk pelanggaran 80% dan 90% ambang batas.

- Tinjau nilai untuk kuota layanan (service quotas) di Wilayah Pasif mana pun (dalam desain Aktif/Pasif). Verifikasi bahwa muatan akan berhasil dijalankan di dalam Wilayah sekunder apabila terjadi kegagalan di dalam Wilayah primer.
- Otomatiskan penilaian jika penyelewengan kuota layanan (service quotas) apa pun telah terjadi antar-Wilayah di dalam akun yang sama dan lakukan tindakan yang semestinya untuk mengubah batas.
- Jika Unit Organisasi (OU) pelanggan disusun dengan cara yang didukung, templat kuota layanan (service quotas) harus diperbarui agar mencerminkan perubahan pada kuota apa pun yang harus diterapkan ke beberapa Wilayah dan akun.
  - Buat templat dan kaitkan Wilayah dengan perubahan kuota.
  - Tinjau semua templat kuota layanan (service quotas) yang ada untuk mengetahui jika ada perubahan yang diperlukan (Wilayah, batas, dan akun).

Sumber daya

Praktik-praktik terbaik terkait:

- REL01-BP01 Sadar akan kuota dan kendala layanan
- REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur
- REL01-BP04 Memantau dan mengelola kuota
- REL01-BP05 Mengotomatiskan manajemen kuota
- REL01-BP06 Pastikan ada kesenjangan yang cukup antara kuota saat ini dan penggunaan maksimum untuk mengakomodasi failover
- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan

Dokumen terkait:

- AWS Pilar Keandalan Well-Architected Framework: Ketersediaan
- AWS Service Quotas (sebelumnya disebut sebagai batas layanan)

- AWS Trusted Advisor Pemeriksaan Praktik Terbaik (lihat bagian Batas Layanan)
- AWS batasi monitor pada AWS jawaban
- Batas EC2 Layanan Amazon
- Apa itu Service Quotas?
- Cara Meminta Peningkatan Kuota
- Titik akhir dan kuota layanan (service quotas)
- Panduan Pengguna Service Quotas
- Monitor Kuota untuk AWS
- AWS Batas Isolasi Kesalahan
- Ketersediaan dengan redundansi
- AWS untuk Data
- Apa itu Integrasi Berkelanjutan?
- Apa itu Pengiriman Berkelanjutan?
- APNMitra: mitra yang dapat membantu manajemen konfigurasi
- Mengelola siklus hidup akun di lingkungan SaaS account-per-tenant AWS
- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Lihat AWS Trusted Advisor rekomendasi dalam skala besar dengan AWS Organizations
- Mengotomatisasi Peningkatan Batas Layanan dan Dukungan Perusahaan dengan AWS Control Tower

Video terkait:

- AWS Live re:Inforce 2019 - Service Quotas
- Melihat dan Mengelola Kuota untuk AWS Layanan Menggunakan Service Quotas
- AWS IAMKuota Demo

Layanan terkait:

- CodeGuru Peninjau Amazon
- AWS CodeDeploy
- AWS CloudTrail

- Amazon CloudWatch
- Amazon EventBridge
- DevOpsGuru Amazon
- AWS Config
- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager
- AWS Marketplace

REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur

Perhatikan kuota layanan (service quotas), batasan layanan, dan batas sumber daya fisik yang tidak dapat diubah. Rancang arsitektur untuk aplikasi dan layanan agar batas ini tidak memengaruhi keandalan.

Contohnya termasuk bandwidth jaringan, ukuran payload pemanggilan fungsi tanpa server, kecepatan burst throttle untuk API gateway, dan koneksi pengguna bersamaan ke database.

Hasil yang diinginkan: Aplikasi atau layanan berfungsi sebagaimana yang diharapkan, baik saat kondisi lalu lintas normal atau pun sedang tinggi. Aplikasi dan layanan telah dirancang untuk berfungsi dengan batasan untuk kuota layanan (service quotas) atau kendala tetap sumber daya tersebut.

Anti-pola umum:

- Memilih sebuah desain yang menggunakan sebuah sumber daya dari sebuah layanan, tidak menyadari bahwa terdapat kendala desain yang akan menyebabkan desain ini gagal begitu Anda menyesuaikan skala.
- Melakukan tolok ukur yang tidak realistis dan akan mencapai kuota tetap layanan selama pengujian. Contohnya, menjalankan pengujian pada batas lonjakan tetapi dalam jangka waktu yang lama.
- Memilih desain yang tidak dapat diskalakan atau dimodifikasi jika kuota layanan (service quotas) tetap akan terlampaui. Misalnya, ukuran SQS muatan 256KB.
- Observabilitas belum dirancang dan diimplementasikan untuk memantau dan memberikan peringatan tentang ambang batas kuota layanan (service quotas) yang mungkin berisiko selama lalu lintas sedang tinggi

Manfaat menerapkan praktik terbaik ini: Memverifikasi bahwa aplikasi akan berjalan berdasarkan semua tingkat beban layanan yang diproyeksikan tanpa terjadi gangguan atau degradasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Tidak seperti kuota layanan lunak atau sumber daya yang diganti dengan unit berkapasitas lebih tinggi, kuota AWS tetap layanan tidak dapat diubah. Ini berarti bahwa semua jenis AWS layanan ini harus dievaluasi untuk potensi batas kapasitas keras ketika digunakan dalam desain aplikasi.

Batas keras ditunjukkan di konsol Service Quotas. Jika kolom menampilkan `ADJUSTABLE = No`, layanan memiliki batas keras. Batas keras juga ditunjukkan di beberapa halaman konfigurasi sumber daya. Contohnya, Lambda memiliki batas keras spesifik yang tidak dapat disesuaikan.

Sebagai contoh, ketika merancang desain dari sebuah aplikasi python untuk beroperasi dalam fungsi Lambda, aplikasi tersebut harus dievaluasi untuk menentukan apakah ada peluang bagi Lambda untuk beroperasi lebih lama dari 15 menit. Jika kode mungkin akan dijalankan lebih dari batas kuota layanan (service quotas) ini, desain atau teknologi lain harus dipertimbangkan. Jika batas ini tercapai setelah deployment produksi, maka aplikasi akan mengalami penurunan kualitas dan gangguan sampai aplikasi itu dapat diperbaiki. Tidak seperti kuota lunak, tidak ada metode untuk mengubah batas-batas ini meskipun dalam kondisi darurat Keparahan 1.

Setelah aplikasi telah di-deploy ke lingkungan pengujian, Anda harus berstrategi untuk menemukan apakah batas keras dapat tercapai. Pengujian stres, pengujian beban, dan pengujian kekacauan harus menjadi bagian dari rencana pengujian pendahuluan.

Langkah-langkah implementasi

- Tinjau daftar lengkap AWS layanan yang dapat digunakan dalam fase desain aplikasi.
- Tinjau batas kuota lunak dan batas kuota keras untuk semua layanan ini. Tidak semua batas ditunjukkan di konsol Service Quotas. Beberapa layanan menggambarkan batasan ini di lokasi yang lain.
- Saat Anda mendesain aplikasi Anda, lakukan peninjauan terhadap pendorong teknologi dan bisnis beban kerja Anda, seperti hasil bisnis, kasus penggunaan, sistem yang dependen, target ketersediaan, dan objek pemulihan bencana. Biarkan pendorong teknologi dan bisnis Anda memandu proses untuk mengidentifikasi sistem terdistribusi yang tepat untuk beban kerja Anda.
- Analisis beban layanan di berbagai Wilayah dan akun. Banyak batas keras yang berbasis wilayah untuk layanan. Tetapi, beberapa batas berbasis akun.

- Analisis arsitektur ketangguhan untuk penggunaan sumber daya selama terjadi kegagalan zona dan kegagalan Wilayah. Jika kemajuan desain multi-Wilayah menggunakan pendekatan aktif/aktif, aktif/pasif – panas, aktif/pasif - dingin, dan aktif/pasif - pilot light, kasus-kasus kegagalan ini akan menyebabkan penggunaan yang lebih tinggi. Hal ini akan menimbulkan potensi kasus penggunaan untuk mencapai batas keras.

Sumber daya

Praktik-praktik terbaik terkait:

- REL01-BP01 Sadar akan kuota dan kendala layanan
- REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah
- REL01-BP04 Memantau dan mengelola kuota
- REL01-BP05 Mengotomatiskan manajemen kuota
- REL01-BP06 Pastikan ada kesenjangan yang cukup antara kuota saat ini dan penggunaan maksimum untuk mengakomodasi failover
- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan

Dokumen terkait:

- AWS Pilar Keandalan Well-Architected Framework: Ketersediaan
- AWS Service Quotas (sebelumnya disebut sebagai batas layanan)
- AWS Trusted Advisor Pemeriksaan Praktik Terbaik (lihat bagian Batas Layanan)
- AWS batasi monitor pada AWS jawaban
- Batas EC2 Layanan Amazon
- Apa itu Service Quotas?
- Cara Meminta Peningkatan Kuota
- Titik akhir dan kuota layanan (service quotas)
- Panduan Pengguna Service Quotas

- Monitor Kuota untuk AWS
- AWS Batas Isolasi Kesalahan
- Ketersediaan dengan redundansi
- AWS untuk Data
- Apa itu Integrasi Berkelanjutan?
- Apa itu Pengiriman Berkelanjutan?
- APNMitra: mitra yang dapat membantu manajemen konfigurasi
- Mengelola siklus hidup akun di lingkungan SaaS account-per-tenant AWS
- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Lihat AWS Trusted Advisor rekomendasi dalam skala besar dengan AWS Organizations
- Mengotomatisasi Peningkatan Batas Layanan dan Dukungan Perusahaan dengan AWS Control Tower
- Tindakan, sumber daya, dan kunci syarat untuk layanan Service Quotas

Video terkait:

- AWS Live re:Inforce 2019 - Service Quotas
- Melihat dan Mengelola Kuota untuk AWS Layanan Menggunakan Service Quotas
- AWS IAMKuota Demo
- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil

Alat terkait:

- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- DevOpsGuru Amazon
- AWS Config
- AWS Trusted Advisor
- AWS CDK

- AWS Systems Manager
- AWS Marketplace

REL01-BP04 Memantau dan mengelola kuota

Evaluasi potensi penggunaan Anda dan tingkatkan kuota dengan semestinya, sehingga terjadi pertumbuhan penggunaan sesuai rencana.

Hasil yang diinginkan: Sistem aktif dan otomatis yang mengelola dan memantau telah di-deploy. Solusi-solusi operasi ini memastikan ambang batas penggunaan kuota hampir dicapai. Hal ini akan secara proaktif diperbaiki berdasarkan perubahan kuota yang diminta.

Anti-pola umum:

- Tidak mengonfigurasi pemantauan untuk memeriksa ambang batas kuota layanan (service quotas)
- Tidak mengonfigurasi pemantauan untuk batas keras, meskipun nilai-nilai tersebut tidak dapat diubah.
- Berasumsi bahwa jumlah waktu yang diperlukan untuk meminta dan mendapatkan perubahan kuota lunak adalah segera atau singkat.
- Mengonfigurasi alarm untuk ketika kuota layanan (service quotas) sudah dekat, namun tidak memiliki proses untuk merespons pemberitahuan.
- Hanya mengonfigurasi alarm untuk layanan yang didukung oleh Service AWS Quotas dan tidak memantau layanan lain. AWS
- Tidak mempertimbangkan manajemen kuota untuk beberapa desain ketangguhan Wilayah, seperti pendekatan aktif/aktif, aktif/pasif - panas, aktif/pasif - dingin, dan aktif/pasif - pilot light.
- Tidak menilai perbedaan kuota antara Wilayah.
- Tidak menilai kebutuhan di setiap Wilayah untuk permintaan peningkatan kuota spesifik.
- Tidak memanfaatkan templat untuk pengelolaan kuota multi Wilayah.

Manfaat menetapkan praktik terbaik ini: Pelacakan otomatis terhadap AWS Service Quotas dan pemantauan penggunaan Anda terhadap kuota tersebut akan memungkinkan Anda untuk melihat kapan Anda mendekati batas kuota. Anda juga dapat menggunakan data pemantauan ini untuk membantu Anda membatasi penurunan kualitas karena kehabisan kuota.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Untuk layanan-layanan yang didukung, Anda dapat memantau kuota Anda dengan mengonfigurasikan berbagai layanan yang berbeda yang dapat menilai dan kemudian mengirimkan pemberitahuan atau alarm. Hal ini dapat membantu Anda memantau penggunaan dan dapat memberitahukan kuota yang akan segera habis kepada Anda. Alarm ini dapat dipanggil dari, fungsi AWS Config Lambda, Amazon CloudWatch, atau dari. AWS Trusted Advisor Anda juga dapat menggunakan filter metrik pada CloudWatch Log untuk mencari dan mengekstrak pola di log untuk menentukan apakah penggunaan mendekati ambang kuota.

Langkah-langkah implementasi

Untuk pemantauan:

- Catat pemakaian sumber daya saat ini (misalnya, bucket atau instans). Gunakan API operasi layanan, seperti Amazon EC2 `DescribeInstances`API, untuk mengumpulkan konsumsi sumber daya saat ini.
- Catat kuota saat ini yang penting dan berlaku untuk layanan dengan menggunakan:
  - AWS Service Quotas
  - AWS Trusted Advisor
  - AWS dokumentasi
  - AWS halaman khusus layanan
  - AWS Command Line Interface (AWS CLI)
  - AWS Cloud Development Kit (AWS CDK)
- Gunakan AWS Service Quotas, AWS layanan yang membantu Anda mengelola kuota untuk lebih dari 250 AWS layanan dari satu lokasi.
- Gunakan batas Trusted Advisor layanan untuk memantau batas layanan Anda saat ini di berbagai ambang batas.
- Gunakan riwayat kuota layanan (konsol atau AWS CLI) untuk memeriksa kenaikan regional.
- Bandingkan perubahan kuota layanan (service quotas) di setiap Wilayah dan setiap akun untuk membuat kesetaraan, jika diperlukan.

Untuk manajemen:

- Otomatis: Siapkan aturan AWS Config khusus untuk memindai kuota layanan di seluruh Wilayah dan membandingkan perbedaannya.

- Otomatis: Siapkan fungsi Lambda terjadwal untuk memindai kuota layanan (service quotas) di berbagai Wilayah dan bandingkan untuk mengetahui perbedaannya.
- Petunjuk: Pindai kuota layanan melalui AWS CLI,API, atau AWS Konsol untuk memindai kuota layanan di seluruh Wilayah dan membandingkan perbedaannya. Laporkan perbedaannya.
- Jika perbedaan kuota diidentifikasi antara Wilayah, mintalah perubahan kuota, jika perlu.
- Tinjau hasil dari semua permintaan.

Sumber daya

Praktik-praktik terbaik terkait:

- REL01-BP01 Sadar akan kuota dan kendala layanan
- REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah
- REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur
- REL01-BP05 Mengotomatiskan manajemen kuota
- REL01-BP06 Pastikan ada kesenjangan yang cukup antara kuota saat ini dan penggunaan maksimum untuk mengakomodasi failover
- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan

Dokumen terkait:

- AWS Pilar Keandalan Well-Architected Framework: Ketersediaan
- AWS Service Quotas (sebelumnya disebut sebagai batas layanan)
- AWS Trusted Advisor Pemeriksaan Praktik Terbaik (lihat bagian Batas Layanan)
- AWS batasi monitor pada AWS jawaban
- Batas EC2 Layanan Amazon
- Apa itu Service Quotas?
- Cara Meminta Peningkatan Kuota

- Titik akhir dan kuota layanan (service quotas)
- Panduan Pengguna Service Quotas
- Monitor Kuota untuk AWS
- AWS Batas Isolasi Kesalahan
- Ketersediaan dengan redundansi
- AWS untuk Data
- Apa itu Integrasi Berkelanjutan?
- Apa itu Pengiriman Berkelanjutan?
- APNMitra: mitra yang dapat membantu manajemen konfigurasi
- Mengelola siklus hidup akun di lingkungan SaaS account-per-tenant AWS
- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Lihat AWS Trusted Advisor rekomendasi dalam skala besar dengan AWS Organizations
- Mengotomatisasi Peningkatan Batas Layanan dan Dukungan Perusahaan dengan AWS Control Tower
- Tindakan, sumber daya, dan kunci syarat untuk layanan Service Quotas

Video terkait:

- AWS Live re: Inforce 2019 - Service Quotas
- Melihat dan Mengelola Kuota untuk AWS Layanan Menggunakan Service Quotas
- AWS IAMKuota Demo
- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil

Alat terkait:

- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- DevOpsGuru Amazon
- AWS Config

- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager
- AWS Marketplace

REL01-BP05 Mengotomatiskan manajemen kuota

Implementasikan alat untuk memperingatkan Anda saat ambang batas terlampaui. Anda dapat mengotomatiskan permintaan peningkatan kuota dengan menggunakan Service AWS Quotas. APIs

Jika Anda mengintegrasikan Database Manajemen Konfigurasi (CMDB) atau sistem tiket dengan Service Quotas, Anda dapat mengotomatiskan pelacakan permintaan peningkatan kuota dan kuota saat ini. Selain itu AWS SDK, Service Quotas menawarkan otomatisasi menggunakan AWS Command Line Interface ()AWS CLI.

Anti-pola umum:

- Melacak kuota dan penggunaan dalam spreadsheet.
- Menjalankan laporan mengenai penggunaan harian, mingguan, bulanan, dan kemudian membandingkan penggunaan terhadap kuota.

Manfaat menetapkan praktik terbaik ini: Pelacakan otomatis kuota AWS layanan dan pemantauan penggunaan Anda terhadap kuota tersebut memungkinkan Anda untuk melihat kapan Anda mendekati kuota. Anda dapat mengatur otomatisasi untuk membantu Anda meminta peningkatan kuota saat diperlukan. Anda dapat mempertimbangkan pengurangan kuota saat penggunaan Anda cenderung tidak selaras untuk benar-benar mengoptimalkan manfaat dari pengurangan risiko (dalam kasus peretasan kredensial) dan penghematan biaya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Siapkan pemantauan otomatis Menerapkan alat yang digunakan SDKs untuk mengingatkan Anda saat ambang sedang didekati.
  - Gunakan Service Quotas dan tambahkan layanan dengan solusi pemantauan kuota otomatis, seperti AWS Limit Monitor atau penawaran dari. AWS Marketplace
    - Apa itu Service Quotas?

    - Monitor Kuota aktif AWS - Solusi AWS
- Siapkan respons otomatis berdasarkan ambang kuota, menggunakan Amazon dan Service SNS AWS Quotas. APIs
- Uji otomatisasi.
    - Konfigurasikan ambang batas.
    - Integrasikan dengan peristiwa perubahan dari AWS Config, pipeline penerapan EventBridge, Amazon, atau pihak ketiga.
    - Atur ambang batas kuota rendah secara artifisial untuk menguji respons.
    - Atur operasi otomatis untuk mengambil tindakan yang sesuai berdasarkan notifikasi dan hubungi AWS Support jika diperlukan.
    - Memulai peristiwa perubahan secara manual.
    - Jalankan game day untuk menguji proses perubahan peningkatan kuota.

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu manajemen konfigurasi
- AWS Marketplace: CMDB produk yang membantu melacak batas
- AWS Service Quotas (sebelumnya disebut batas layanan)
- AWS Trusted Advisor Pemeriksaan Praktik Terbaik (lihat bagian Batas Layanan)
- Monitor Kuota aktif AWS - Solusi AWS
- Batas EC2 Layanan Amazon
- Apa itu Service Quotas?

Video terkait:

- AWS Live re: Inforce 2019 - Service Quotas

REL01-BP06 Pastikan ada kesenjangan yang cukup antara kuota saat ini dan penggunaan maksimum untuk mengakomodasi failover

Artikel ini akan menjelaskan kepada Anda tentang cara menjaga ruang antara kuota sumber daya dan penggunaan Anda, dan bagaimana hal itu dapat memberikan manfaat bagi organisasi Anda.

Setelah Anda selesai menggunakan sebuah sumber daya, kuota penggunaan tersebut dapat terus memperhitungkan sumber daya itu. Hal ini dapat mengakibatkan terjadinya sumber daya yang gagal atau tidak dapat diakses. Hindari kegagalan sumber daya dengan memastikan apakah kuota meliputi sumber daya yang tumpang tindih dan tidak dapat diakses serta penggantinya. Pertimbangkan kasus penggunaan seperti kegagalan jaringan, kegagalan Zona Ketersediaan, atau kegagalan Wilayah ketika menghitung selisih ini.

Hasil yang diinginkan: Kegagalan kecil atau besar dalam sumber daya atau aksesibilitas sumber daya dapat tercakup dalam ambang batas layanan yang berlaku saat ini. Kegagalan zona, kegagalan jaringan, atau bahkan kegagalan Wilayah telah dipertimbangkan dalam pembuatan rencana sumber daya.

Anti-pola umum:

- Mengatur kuota layanan (service quotas) berdasarkan kebutuhan saat ini tanpa memperhitungkan skenario failover.
- Tidak mempertimbangkan pengguna utama stabilitas statis ketika menghitung kuota puncak untuk sebuah layanan.
- Tidak mempertimbangkan potensi kemungkinan sumber daya yang tidak dapat diakses dalam menghitung kuota total yang diperlukan untuk masing-masing Wilayah.
- Tidak mempertimbangkan batas isolasi kesalahan AWS layanan untuk beberapa layanan dan potensi pola penggunaan abnormal mereka.

Manfaat menerapkan praktik terbaik ini: Ketika terjadi peristiwa gangguan layanan yang memengaruhi ketersediaan aplikasi, gunakan cloud untuk menerapkan strategi-strategi untuk pulih dari peristiwa ini. Contoh strategi adalah membuat sumber daya tambahan untuk menggantikan sumber daya yang tidak dapat diakses untuk mengakomodasi kondisi failover tanpa menghabiskan batas layanan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Ketika melakukan evaluasi batas kuota, Anda juga harus mempertimbangkan kasus-kasus failover yang dapat terjadi karena adanya degradasi. Pertimbangkan kasus-kasus failover berikut ini.

- Terganggu atau tidak dapat diaksesVPC.
- Subnet yang tidak dapat diakses.

- Zona Ketersediaan yang mengalami degradasi dan memengaruhi aksesibilitas sumber daya.
- Berbagai rute titik keluar dan masuk atau rute jaringan yang diblokir atau berubah.
- Wilayah terdegradasi yang berdampak pada aksesibilitas sumber daya.
- Subset sumber daya yang dipengaruhi oleh kegagalan di Wilayah atau Zona Ketersediaan.

Keputusan untuk failover adalah keputusan yang bersifat unik untuk setiap situasi dan pelanggan, karena dampak bisnisnya bisa sangat berbeda. Atasi perencanaan kapasitas sumber daya di lokasi failover dan kuota sumber daya sebelum Anda memutuskan untuk melakukan failover aplikasi atau layanan.

Anda juga harus mempertimbangkan puncak aktivitas yang lebih tinggi dari normal saat meninjau kuota untuk setiap layanan. Puncak ini mungkin terkait dengan sumber daya yang tidak dapat diakses karena jaringan atau izin, tetapi masih aktif. Sumber daya aktif yang tidak dihentikan dihitung untuk memenuhi batas kuota layanan (service quotas).

Langkah-langkah implementasi

- Pertahankan ruang antara kuota layanan (service quotas) saat ini dan penggunaan maksimum untuk mengakomodasi failover atau hilangnya kemampuan akses.
- Tentukan service quotas Anda. Memperhitungkan pola deployment yang khas, persyaratan ketersediaan, dan pertumbuhan konsumsi.
- Minta peningkatan kuota, jika perlu. Antisipasi waktu tunggu untuk permintaan kenaikan kuota.
- Tentukan persyaratan keandalan (juga disebut sebagai jumlah angka sembilan Anda).
- Memahami skenario kesalahan potensial seperti hilangnya komponen, Zona Ketersediaan, atau Wilayah.
- Tetapkan metodologi deployment Anda (misalnya canary, blue/green, red/black, atau rolling).
- Sertakan buffer yang sesuai untuk batas saat ini. Contoh buffer bisa 15%.
- Sertakan perhitungan untuk stabilitas statis (Zona dan Wilayah), apabila sesuai.
- Rencanakan peningkatan pemakaian dan pantau tren pemakaian Anda.
- Pertimbangkan dampak stabilitas statis untuk beban kerja Anda yang paling penting. Lakukan penilaian terhadap sumber daya yang sesuai dengan sebuah sistem stabil secara statis di semua Wilayah dan Zona Ketersediaan.
- Pertimbangkan untuk menggunakan Reservasi Kapasitas Sesuai Permintaan untuk menjadwalkan kapasitas sebelum failover. Hal ini bisa menjadi strategi yang bermanfaat untuk

mengimplementasikan penjadwalan bisnis yang paling penting guna mengurangi potensi risiko mendapatkan kuantitas dan jenis sumber daya yang benar selama failover.

Sumber daya

Praktik-praktik terbaik terkait:

- REL01-BP01 Sadar akan kuota dan kendala layanan
- REL01-BP02 Mengelola kuota layanan di seluruh akun dan wilayah
- REL01-BP03 Mengakomodasi kuota dan kendala layanan tetap melalui arsitektur
- REL01-BP04 Memantau dan mengelola kuota
- REL01-BP05 Mengotomatiskan manajemen kuota
- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan

Dokumen terkait:

- AWS Pilar Keandalan Well-Architected Framework: Ketersediaan
- AWS Service Quotas (sebelumnya disebut sebagai batas layanan)
- AWS Trusted Advisor Pemeriksaan Praktik Terbaik (lihat bagian Batas Layanan)
- AWS batasi monitor pada AWS jawaban
- Batas EC2 Layanan Amazon
- Apa itu Service Quotas?
- Cara Meminta Peningkatan Kuota
- Titik akhir dan kuota layanan (service quotas)
- Panduan Pengguna Service Quotas
- Monitor Kuota untuk AWS
- AWS Batas Isolasi Kesalahan

- Ketersediaan dengan redundansi
- AWS untuk Data
- Apa itu Integrasi Berkelanjutan?
- Apa itu Pengiriman Berkelanjutan?
- APNMitra: mitra yang dapat membantu manajemen konfigurasi
- Mengelola siklus hidup akun di lingkungan SaaS account-per-tenant AWS
- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Lihat AWS Trusted Advisor rekomendasi dalam skala besar dengan AWS Organizations
- Mengotomatisasi Peningkatan Batas Layanan dan Dukungan Perusahaan dengan AWS Control Tower
- Tindakan, sumber daya, dan kunci syarat untuk layanan Service Quotas

Video terkait:

- AWS Live re:Inforce 2019 - Service Quotas
- Melihat dan Mengelola Kuota untuk AWS Layanan Menggunakan Service Quotas
- AWS IAMKuota Demo
- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil

Alat terkait:

- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- DevOpsGuru Amazon
- AWS Config
- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager

- AWS Marketplace

## REL2. Bagaimana cara merencanakan topologi jaringan Anda?

Sering kali beban kerja ada di beberapa lingkungan. Ini termasuk beberapa lingkungan cloud (baik yang dapat diakses publik maupun privat) dan kemungkinan infrastruktur pusat data Anda yang ada sekarang. Rencana harus mencakup pertimbangan-pertimbangan jaringan seperti konektivitas di dalam dan antarsistem, pengelolaan alamat IP publik, pengelolaan alamat IP privat, dan resolusi nama domain.

Praktik terbaik

- REL02-BP01 Gunakan konektivitas jaringan yang sangat tersedia untuk titik akhir publik beban kerja Anda
- REL02-BP02 Menyediakan konektivitas redundan antara jaringan pribadi di lingkungan cloud dan lokal
- REL02-BP03 Memastikan akun alokasi subnet IP untuk ekspansi dan ketersediaan
- REL02-BP04 Lebih suka topologi daripada mesh hub-and-spoke many-to-many
- REL02-BP05 Menerapkan rentang alamat IP pribadi yang tidak tumpang tindih di semua ruang alamat pribadi tempat mereka terhubung

REL02-BP01 Gunakan konektivitas jaringan yang sangat tersedia untuk titik akhir publik beban kerja Anda

Membangun konektivitas jaringan yang sangat tersedia ke titik akhir publik beban kerja Anda dapat membantu Anda mengurangi waktu henti karena hilangnya konektivitas dan meningkatkan ketersediaan dan SLA beban kerja Anda. Untuk mencapai hal ini, gunakan jaringan pengiriman konten (CDNs), API gateway, load balancing, atau proxy terbalik yang sangat tersediaDNS.

Hasil yang diinginkan: Sangat penting bagi Anda untuk merencanakan, membangun, dan mengoperasionalkan konektivitas jaringan dengan ketersediaan tinggi untuk titik akhir publik Anda. Jika beban kerja Anda menjadi tidak terjangkau karena hilangnya konektivitas, meskipun beban kerja Anda beroperasi dan tersedia, para pelanggan Anda akan menganggap sistem Anda tidak berfungsi (down). Dengan menggabungkan konektivitas jaringan yang tangguh dan memiliki ketersediaan yang tinggi untuk titik akhir publik beban kerja Anda, bersama dengan arsitektur yang tangguh untuk beban kerja itu sendiri, Anda dapat memberikan tingkat layanan dan ketersediaan sebaik mungkin untuk para pelanggan Anda.

AWS Global Accelerator, Amazon CloudFront, Amazon API Gateway, AWS Lambda Function URLs AWS AppSync APIs, dan Elastic Load Balancing (ELB) semuanya menyediakan titik akhir publik yang sangat tersedia. Amazon Route 53 menyediakan DNS layanan yang sangat tersedia untuk resolusi nama domain untuk memverifikasi bahwa alamat titik akhir publik Anda dapat diselesaikan.

Anda juga dapat mengevaluasi peralatan AWS Marketplace perangkat lunak untuk load balancing dan proxy.

Anti-pola umum:

- Merancang beban kerja yang sangat tersedia tanpa perencanaan DNS dan konektivitas jaringan untuk ketersediaan tinggi.
- Menggunakan alamat internet publik pada instance atau wadah individu dan mengelola konektivitas dengan DNS mereka.
- Menggunakan alamat IP, bukan nama domain untuk mencari layanan.
- Tidak menguji skenario di mana konektivitas ke titik akhir publik Anda hilang.
- Tidak menganalisis pola distribusi dan kebutuhan throughput jaringan.
- Tidak menguji dan merencanakan skenario di mana konektivitas jaringan internet ke titik akhir publik beban kerja Anda mungkin mengalami gangguan.
- Memberikan konten (seperti halaman web, aset statis, atau file media) ke area-area geografis besar dan tidak menggunakan CDN.
- Tidak merencanakan serangan penolakan layanan (DDoS) terdistribusi. DDoSSerangan berisiko mematikan lalu lintas yang sah dan menurunkan ketersediaan untuk pengguna Anda.

Manfaat menerapkan praktik terbaik ini: Membuat desain untuk konektivitas jaringan dengan ketersediaan tinggi dan tangguh akan memastikan bahwa beban kerja Anda dapat diakses dan tersedia bagi para pengguna Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Inti dari pembangunan konektivitas jaringan dengan ketersediaan tinggi untuk titik akhir publik adalah pengarahan rute lalu lintas. Untuk memverifikasi lalu lintas Anda dapat mencapai titik akhir, DNS harus dapat menyelesaikan nama domain ke alamat IP yang sesuai. Gunakan Sistem Nama Domain (DNS) yang sangat tersedia dan dapat diskalakan seperti Amazon Route 53 untuk mengelola DNS catatan domain Anda. Anda juga dapat menggunakan pemeriksaan kondisi yang disediakan oleh

Amazon Route 53. Pemeriksaan kesehatan memverifikasi bahwa aplikasi Anda dapat dijangkau, tersedia, dan fungsional, dan dapat diatur sedemikian rupa sehingga meniru perilaku pengguna Anda, seperti meminta halaman web atau yang spesifik. URL Jika terjadi kegagalan, Amazon Route 53 merespons permintaan DNS resolusi dan mengarahkan lalu lintas ke titik akhir yang sehat saja. Anda juga dapat mempertimbangkan untuk menggunakan kemampuan Perutean Berbasis Geo DNS dan Latensi yang ditawarkan oleh Amazon Route 53.

Untuk memverifikasi bahwa beban kerja Anda sendiri sangat tersedia, gunakan Elastic Load Balancing ELB (). Amazon Route 53 dapat digunakan untuk menargetkan lalu lintas keELB, yang mendistribusikan lalu lintas ke instans komputasi target. Anda juga dapat menggunakan Amazon API Gateway bersama dengan AWS Lambda solusi tanpa server. Pelanggan juga dapat menjalankan beban kerja dalam beberapa Wilayah AWS. Dengan pola aktif/aktif multi-situs, beban kerja dapat melayani lalu lintas yang datang dari beberapa Wilayah. Dengan pola aktif/pasif multi-situs, beban kerja dapat melayani lalu lintas dari wilayah aktif sementara data akan direplikasi ke wilayah sekunder dan menjadi aktif jika terjadi kegagalan di wilayah primer. Pemeriksaan kesehatan Route 53 kemudian dapat digunakan untuk mengontrol DNS failover dari titik akhir mana pun di Wilayah primer ke titik akhir di Wilayah sekunder, memverifikasi bahwa beban kerja Anda dapat dijangkau dan tersedia untuk pengguna Anda.

Amazon CloudFront menyediakan kemudahan API untuk mendistribusikan konten dengan latensi rendah dan kecepatan transfer data tinggi dengan melayani permintaan menggunakan jaringan lokasi edge di seluruh dunia. Jaringan pengiriman konten (CDNs) melayani pelanggan dengan menyajikan konten yang terletak atau di-cache di lokasi yang dekat dengan pengguna. Ini juga meningkatkan ketersediaan aplikasi Anda karena beban untuk konten digeser dari server Anda ke CloudFront lokasi tepi. Lokasi edge dan cache edge regional menyimpan cache salinan konten Anda berada dekat dengan penonton Anda sehingga konten dapat diambil dengan cepat, konten lebih mudah dijangkau, dan beban kerja memiliki ketersediaan yang lebih tinggi.

Untuk beban kerja dengan pengguna yang tersebar secara geografis, AWS Global Accelerator membantu Anda meningkatkan ketersediaan dan kinerja aplikasi. AWS Global Accelerator menyediakan alamat IP statis Anycast yang berfungsi sebagai titik masuk tetap ke aplikasi Anda yang dihosting dalam satu atau lebih Wilayah AWS. Hal ini memungkinkan lalu lintas masuk ke jaringan AWS global sedekat mungkin dengan pengguna Anda, meningkatkan jangkauan dan ketersediaan beban kerja Anda. AWS Global Accelerator juga memantau kesehatan titik akhir aplikasi Anda dengan menggunakanTCP,HTTP, dan pemeriksaan HTTPS kesehatan. Setiap perubahan yang terjadi pada kondisi atau konfigurasi titik akhir Anda akan mengizinkan pengarahan ulang lalu lintas pengguna ke titik akhir yang memiliki kondisi kesehatan yang bagus yang memberikan ketersediaan dan performa terbaik bagi pengguna Anda. Selain itu, AWS Global Accelerator memiliki desain isolasi

kesalahan yang menggunakan dua IPv4 alamat statis yang dilayani oleh zona jaringan independen yang meningkatkan ketersediaan aplikasi Anda.

Untuk membantu melindungi pelanggan dari DDoS serangan, AWS sediakan AWS Shield Standard. Shield Standard secara otomatis dihidupkan dan melindungi dari serangan infrastruktur umum (lapisan 3 dan 4) SYN UDP seperti/banjir dan serangan refleksi untuk mendukung ketersediaan aplikasi Anda yang tinggi. AWS Untuk perlindungan tambahan terhadap serangan yang lebih canggih dan lebih besar (seperti UDP banjir), serangan kelelahan status (seperti TCP SYN banjir), dan untuk membantu melindungi aplikasi Anda yang berjalan di Amazon Elastic Compute Cloud (Amazon), Elastic EC2 Load Balancing ()ELB, Amazon, dan Route 53, Anda dapat CloudFront mempertimbangkan untuk AWS Global Accelerator menggunakannya. AWS Shield Advanced Untuk perlindungan terhadap serangan lapisan Aplikasi seperti HTTP POST atau GET banjir, gunakan AWS WAF. AWS WAF dapat menggunakan alamat IP, HTTP header, HTTP badan, URI string, SQL injeksi, dan kondisi skrip lintas situs untuk menentukan apakah permintaan harus diblokir atau diizinkan.

Langkah-langkah implementasi

1. Siapkan sangat tersediaDNS: Amazon Route 53 adalah layanan web sistem nama domain (DNS) yang sangat tersedia dan dapat diskalakan. Route 53 menghubungkan permintaan pengguna ke aplikasi internet yang berjalan di AWS atau di tempat. Untuk informasi selengkapnya, lihat mengonfigurasi Amazon Route 53 sebagai DNS layanan Anda.
2. Siapkan pemeriksaan kondisi: Ketika menggunakan Route 53, verifikasi bahwa hanya target dengan kondisi bagus yang dapat diselesaikan. Mulailah dengan membuat pemeriksaan kesehatan Route 53 dan mengonfigurasi DNS failover. Aspek-aspek berikut penting untuk Anda pertimbangkan ketika mempersiapkan pemeriksaan kondisi:
   a. Bagaimana Amazon Route 53 menentukan apakah pemeriksaan kondisi hasilnya sehat atau tidak
   b. Membuat, memperbarui, dan menghapus pemeriksaan kondisi
   c. Memantau status pemeriksaan kondisi dan mendapatkan notifikasi
   d. Praktik terbaik untuk Amazon Route 53 DNS
3. Hubungkan DNS layanan Anda ke titik akhir Anda.
   a. Saat menggunakan Penyeimbangan Beban Elastis sebagai target untuk lalu lintas Anda, buat catatan alias menggunakan Amazon Route 53 yang mengarah ke titik akhir regional penyeimbang beban Anda. Selama pembuatan catatan alias, atur opsi Evaluasi kondisi target ke Ya.

   b. Untuk beban kerja tanpa server atau pribadi APIs saat API Gateway digunakan, gunakan Route 53 untuk mengarahkan lalu lintas ke Gateway. API

4. Buat keputusan mengenai jaringan pengiriman konten.
   a. Untuk mengirimkan konten menggunakan lokasi tepi yang lebih dekat dengan pengguna, mulailah dengan memahami cara CloudFront mengirimkan konten.
   b. Mulailah dengan CloudFront distribusi sederhana. CloudFront kemudian tahu dari mana Anda ingin konten dikirim, dan detail tentang cara melacak dan mengelola pengiriman konten. Aspek-aspek berikut penting untuk dipahami dan dipertimbangkan saat mengatur CloudFront distribusi:
      i. Cara kerja caching dengan lokasi CloudFront tepi
      ii. Meningkatkan proporsi permintaan yang disajikan langsung dari CloudFront cache (rasio hit cache)
      iii. Menggunakan Amazon CloudFront Origin Shield
      iv. Mengoptimalkan ketersediaan tinggi dengan failover CloudFront asal
5. Siapkan perlindungan lapisan aplikasi: AWS WAF membantu Anda melindungi dari eksploitasi web umum dan bot yang dapat memengaruhi ketersediaan, membahayakan keamanan, atau mengkonsumsi sumber daya yang berlebihan. Untuk mendapatkan pemahaman yang lebih dalam, tinjau cara AWS WAF kerja dan kapan Anda siap menerapkan perlindungan dari HTTP POST AND GET banjir lapisan aplikasi, tinjau Memulai dengan. AWS WAF Anda juga dapat menggunakan AWS WAF dengan CloudFront melihat dokumentasi tentang cara AWS WAF bekerja dengan CloudFront fitur Amazon.
6. Siapkan DDoS perlindungan tambahan: Secara default, semua AWS pelanggan menerima perlindungan dari DDoS serangan lapisan jaringan dan transport yang umum dan paling sering terjadi yang menargetkan situs web atau aplikasi Anda tanpa AWS Shield Standard biaya tambahan. Untuk perlindungan tambahan terhadap aplikasi yang menghadap ke internet yang berjalan di AmazonEC2, Elastic Load Balancing, CloudFront Amazon, dan AWS Global Accelerator Amazon Route 53, Anda AWS Shield Advanceddapat mempertimbangkan dan meninjau DDoS contoh arsitektur tangguh. Untuk melindungi beban kerja dan titik akhir publik Anda dari DDoS serangan, tinjau Memulai. AWS Shield Advanced

Sumber daya

Praktik-praktik terbaik terkait:

- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL10-BP02 Pilih lokasi yang sesuai untuk penyebaran multi-lokasi Anda

- REL11-BP04 Mengandalkan bidang data dan bukan bidang kontrol selama pemulihan
- REL11-BP06 Kirim pemberitahuan saat acara memengaruhi ketersediaan

Dokumen terkait:

- APNMitra: mitra yang dapat membantu merencanakan jaringan Anda
- AWS Marketplace untuk Infrastruktur Jaringan
- Apa itu AWS Global Accelerator?
- Apa itu Amazon CloudFront?
- Apa itu Amazon Route 53?
- Apa itu Penyeimbangan Beban Elastis?
- Kemampuan Konektivitas Jaringan - Membangun Fondasi Cloud Anda
- Apa itu Amazon API Gateway?
- Apa AWS WAF, AWS Shield, dan AWS Firewall Manager?
- Apa itu Pengontrol Pemulihan Aplikasi Amazon?
- Konfigurasikan pemeriksaan kesehatan khusus untuk DNS failover

Video terkait:

- AWS re:invent 2022 - Tingkatkan kinerja dan ketersediaan dengan AWS Global Accelerator
- AWS re: invent 2020: Manajemen lalu lintas global dengan Amazon Route 53
- AWS Re:invent 2022 - Mengoperasikan aplikasi Multi-AZ yang sangat tersedia
- AWS re:invent 2022 - Selami infrastruktur jaringan AWS
- AWS re:invent 2022 - Membangun jaringan tangguh

Contoh terkait:

- Pemulihan Bencana dengan Pengontrol Pemulihan Aplikasi Amazon (ARC)
- Lokakarya Keandalan
- AWS Global Accelerator Lokakarya

REL02-BP02 Menyediakan konektivitas redundan antara jaringan pribadi di lingkungan cloud dan lokal

Implementasikan redundansi dalam koneksi Anda antara jaringan pribadi di cloud dan lingkungan on-premise untuk mencapai ketahanan konektivitas. Ini dapat dicapai dengan melakukan deployment dua atau lebih tautan dan jalur lalu lintas, sehingga menjaga konektivitas jika terjadi kegagalan jaringan.

Anti-pola umum:

- Anda bergantung hanya pada satu koneksi jaringan, yang akan menciptakan satu titik kegagalan.
- Anda hanya menggunakan satu VPN terowongan atau beberapa terowongan yang berakhir di Availability Zone yang sama.
- Anda mengandalkan satu ISP untuk VPN konektivitas, yang dapat menyebabkan kegagalan total selama ISP pemadaman.
- Tidak menerapkan protokol perutean dinamis sepertiBGP, yang sangat penting untuk mengalihkan lalu lintas selama gangguan jaringan.
- Anda mengabaikan keterbatasan bandwidth VPN terowongan dan melebih-lebihkan kemampuan cadangannya.

Manfaat menerapkan praktik terbaik ini: Dengan mengimplementasikan konektivitas redundan antara lingkungan cloud dan lingkungan perusahaan atau on-premise Anda, maka layanan-layanan dependen yang ada di antara dua lingkungan tersebut dapat berkomunikasi dengan andal.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Saat menggunakan AWS Direct Connect untuk menghubungkan jaringan lokal AWS, Anda dapat mencapai ketahanan jaringan maksimum (SLA99,99%) dengan menggunakan koneksi terpisah yang berakhir pada perangkat berbeda di lebih dari satu lokasi lokal dan lebih dari satu lokasi. AWS Direct Connect Topologi ini menawarkan Anda ketahanan terhadap kegagalan perangkat, masalah konektivitas, dan pemadaman (outage) lokasi total. Atau, Anda dapat mencapai ketahanan tinggi (SLA99,9%) dengan menggunakan dua koneksi individual ke beberapa lokasi (setiap lokasi lokal terhubung ke satu lokasi Direct Connect). Pendekatan ini akan melindungi Anda dari gangguan konektivitas yang disebabkan oleh pemotongan serat atau kegagalan perangkat dan membantu Anda mengurangi kegagalan lokasi total. The AWS Direct Connect Resiliency Toolkit dapat membantu dalam merancang topologi Anda. AWS Direct Connect

Anda juga dapat mempertimbangkan untuk AWS Site-to-Site VPN mengakhiri AWS Transit Gateway sebagai cadangan hemat biaya untuk koneksi utama AWS Direct Connect Anda. Pengaturan ini memungkinkan perutean multipath (ECMP) dengan biaya yang sama di beberapa VPN terowongan, memungkinkan throughput hingga 50Gbps, meskipun setiap terowongan dibatasi pada 1,25 Gbps. VPN Penting untuk dicatat, bagaimanapun, itu masih AWS Direct Connect merupakan pilihan paling efektif untuk meminimalkan gangguan jaringan dan menyediakan konektivitas yang stabil.

Saat menggunakan VPNs melalui internet untuk menghubungkan lingkungan cloud ke pusat data lokal, konfigurasikan dua VPN terowongan sebagai bagian dari satu site-to-site VPN koneksi. Setiap terowongan harus berakhir di Zona Ketersediaan yang berbeda untuk mendapatkan ketersediaan yang tinggi dan harus menggunakan perangkat keras redundan untuk mencegah kegagalan perangkat on-premise. Selain itu, pertimbangkan beberapa koneksi internet dari berbagai penyedia layanan internet (ISPs) di lokasi lokal Anda untuk menghindari gangguan VPN konektivitas total karena satu ISP pemadaman. Memilih ISPs dengan perutean dan infrastruktur yang beragam, terutama yang memiliki jalur fisik terpisah ke AWS titik akhir, memberikan ketersediaan konektivitas yang tinggi.

Selain redundansi fisik dengan beberapa AWS Direct Connect koneksi dan beberapa VPN terowongan (atau kombinasi keduanya), menerapkan perutean dinamis Border Gateway Protocol (BGP) juga penting. Dynamic BGP menyediakan pengalihan lalu lintas otomatis dari satu jalur ke jalur lain berdasarkan kondisi jaringan real-time dan kebijakan yang dikonfigurasi. Perilaku dinamis ini sangat bermanfaat dalam menjaga ketersediaan jaringan dan kontinuitas layanan jika ada kegagalan tautan atau jaringan yang terjadi. Jalur alternatif dipilih dengan cepat, sehingga dapat meningkatkan ketahanan dan keandalan jaringan.

Langkah-langkah implementasi

- Akuisisi konektivitas yang sangat tersedia antara AWS dan lingkungan lokal Anda.
  - Gunakan beberapa AWS Direct Connect koneksi atau VPN terowongan antara jaringan pribadi yang digunakan secara terpisah.
  - Gunakan beberapa AWS Direct Connect lokasi untuk ketersediaan tinggi.
  - Jika menggunakan beberapa Wilayah AWS, buat redundansi setidaknya di dua di antaranya.
- Gunakan AWS Transit Gateway, jika memungkinkan, untuk mengakhiri VPNkoneksi Anda.
- Evaluasi AWS Marketplace peralatan untuk mengakhiri VPNs atau memperluas SD- WAN ke AWS. Jika Anda menggunakan AWS Marketplace peralatan, gunakan instans redundan untuk ketersediaan tinggi di Availability Zone yang berbeda.
- Sediakan koneksi redundan ke lingkungan on-premise Anda.

- Anda mungkin memerlukan koneksi redundan ke beberapa Wilayah AWS untuk mencapai kebutuhan ketersediaan Anda.
- Menggunakan AWS Direct Connect Resiliency Toolkit untuk memulai.

Sumber daya

Dokumen terkait:

- AWS Direct Connect Rekomendasi Ketahanan
- Menggunakan Site-to-Site VPN Koneksi Redundan untuk Menyediakan Failover
- Kebijakan dan komunitas perutean BGP
- Konfigurasi Aktif/Aktif dan Aktif/Pasif di AWS Direct Connect
- APNMitra: mitra yang dapat membantu merencanakan jaringan Anda
- AWS Marketplace untuk Infrastruktur Jaringan
- Laporan Resmi Pilihan Konektivitas Amazon Virtual Private Cloud
- Membangun Infrastruktur Multi VPC AWS Jaringan yang Skalabel dan Aman
- Menggunakan Site-to-Site VPN koneksi redundan untuk menyediakan failover
- Menggunakan AWS Direct Connect Resiliency Toolkit untuk memulai
- VPCLayanan Endpoint dan VPC Endpoint ()AWS PrivateLink
- Apa itu AmazonVPC?
- Apa itu gateway transit?
- Apa itu AWS Site-to-Site VPN?
- Bekerja dengan gateway Direct Connect

Video terkait:

- AWS re: invent 2018: VPC Desain Lanjutan dan Kemampuan Baru untuk Amazon VPC
- AWS Re: invent 2019: arsitektur AWS Transit Gateway referensi untuk banyak orang VPCs

REL02-BP03 Memastikan akun alokasi subnet IP untuk ekspansi dan ketersediaan

Rentang alamat VPC IP Amazon harus cukup besar untuk mengakomodasi persyaratan beban kerja, termasuk anjak piutang dalam ekspansi masa depan dan alokasi alamat IP ke subnet di seluruh Availability Zone. Ini termasuk penyeimbang beban, EC2 instance, dan aplikasi berbasis kontainer.

Ketika Anda merencanakan topologi jaringan Anda, langkah pertama yang harus dilakukan adalah menetapkan ruang alamat IP itu sendiri. Rentang alamat IP pribadi (mengikuti pedoman RFC 1918) harus dialokasikan untuk masing-masing. VPC Akomodasikan persyaratan berikut sebagai bagian dari proses ini:

- Izinkan ruang alamat IP untuk lebih dari satu VPC per Wilayah.
- Dalam aVPC, berikan ruang untuk beberapa subnet sehingga Anda dapat mencakup beberapa Availability Zone.
- Pertimbangkan untuk meninggalkan ruang CIDR blok yang tidak terpakai dalam ekspansi VPC for future.
- Pastikan ada ruang alamat IP untuk memenuhi kebutuhan armada sementara EC2 instans Amazon yang mungkin Anda gunakan, seperti Armada Spot untuk pembelajaran mesin, kluster AmazonEMR, atau kluster Amazon Redshift. Pertimbangan serupa harus diberikan pada klaster Kubernetes, seperti Amazon Elastic Kubernetes Service (EKSAmazon), karena setiap pod Kubernetes diberi alamat routable dari blok secara default. VPC CIDR
- Perhatikan bahwa empat alamat IP pertama dan alamat IP terakhir di setiap CIDR blok subnet dicadangkan dan tidak tersedia untuk Anda gunakan.
- Perhatikan bahwa VPC CIDR blok awal yang dialokasikan untuk Anda VPC tidak dapat diubah atau dihapus, tetapi Anda dapat menambahkan blok non-tumpang tindih CIDR tambahan ke blok. VPC Subnet IPv4 CIDRs tidak dapat diubah, namun IPv6 CIDRs bisa.
- VPCCIDRBlok terbesar yang mungkin adalah /16, dan yang terkecil adalah /28.
- Pertimbangkan jaringan lain yang terhubung (VPC, lokal, atau penyedia cloud lainnya) dan pastikan ruang alamat IP yang tidak tumpang tindih. Untuk informasi selengkapnya, lihat REL02-BP05 Menegakkan rentang alamat IP pribadi yang tidak tumpang tindih di semua ruang alamat pribadi tempat mereka terhubung.

Hasil yang diinginkan: Sebuah subnet IP yang dapat diskalakan dapat membantu Anda mengakomodasi pertumbuhan yang mungkin terjadi di masa depan dan menghindari pemborosan yang tidak perlu.

Anti-pola umum:

- Gagal mempertimbangkan pertumbuhan masa depan, menghasilkan CIDR blok yang terlalu kecil dan membutuhkan konfigurasi ulang, berpotensi menyebabkan downtime.
- Salah memperkirakan jumlah alamat IP yang dapat digunakan oleh satu penyeimbang beban elastis.

- Melakukan deployment banyak penyeimbang beban lalu lintas tinggi ke subnet yang sama
- Menggunakan mekanisme penskalaan otomatis tetapi gagal memantau pemakaian alamat IP.
- Mendefinisikan CIDR rentang yang terlalu besar jauh melampaui ekspektasi pertumbuhan masa depan, yang dapat menyebabkan kesulitan mengintip dengan jaringan lain dengan rentang alamat yang tumpang tindih.

Manfaat menerapkan praktik terbaik ini: Ini akan memastikan bahwa Anda dapat mengakomodasi pertumbuhan beban kerja Anda dan akan terus memberikan ketersediaan saat Anda menaikkan skala.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Rencanakan jaringan Anda untuk mengakomodasi pertumbuhan, kepatuhan terhadap peraturan, dan integrasi dengan jaringan yang lain. Pertumbuhan mungkin saja lebih besar dari yang diperkirakan, kepatuhan terhadap peraturan dapat berubah, dan koneksi jaringan privat atau akuisisi bisa jadi akan sulit diimplementasikan tanpa melakukan perencanaan yang baik.

- Pilih yang relevan Akun AWS dan Wilayah berdasarkan persyaratan layanan, latensi, peraturan, dan pemulihan bencana (DR) Anda.
- Identifikasi kebutuhan Anda untuk VPC penyebaran regional.
- Identifikasi ukuranVPCs.
  - Tentukan apakah Anda akan menggunakan VPC multi-konektivitas.
    - Apa Itu Gateway Transit?
    - VPCKonektivitas Multi Wilayah Tunggal
  - Tentukan apakah Anda membutuhkan jaringan terpisah (segregated) untuk persyaratan berdasarkan regulasi.
  - Buat VPCs dengan CIDR blok berukuran tepat untuk mengakomodasi kebutuhan Anda saat ini dan masa depan.
    - Jika Anda memiliki proyeksi pertumbuhan yang tidak diketahui, Anda mungkin ingin melakukan kesalahan di sisi CIDR blok yang lebih besar untuk mengurangi potensi konfigurasi ulang di masa depan
  - Pertimbangkan untuk menggunakan IPv6pengalamatan untuk subnet sebagai bagian dari dual-stack. VPC IPv6sangat cocok untuk digunakan dalam subnet pribadi yang berisi armada instance atau wadah fana yang seharusnya membutuhkan sejumlah besar alamat. IPv4

Sumber daya

Praktik terbaik Well-Architected terkait:

- REL02-BP05 Menerapkan rentang alamat IP pribadi yang tidak tumpang tindih di semua ruang alamat pribadi tempat mereka terhubung

Dokumen terkait:

- APNMitra: mitra yang dapat membantu merencanakan jaringan Anda
- AWS Marketplace untuk Infrastruktur Jaringan
- Laporan Resmi Pilihan Konektivitas Amazon Virtual Private Cloud
- Konektivitas jaringan ketersediaan tinggi (HA) beberapa pusat data
- VPCKonektivitas Multi Wilayah Tunggal
- Apa itu AmazonVPC?
- IPv6pada AWS
- IPv6pada arsitektur referensi
- Amazon Elastic Kubernetes Service meluncurkan dukungan IPv6
- Rekomendasi untuk Anda VPC - Classic Load Balancers
- Subnet Zona Ketersediaan - Penyeimbang Beban Aplikasi
- Availability Zone - Network Load Balancers

Video terkait:

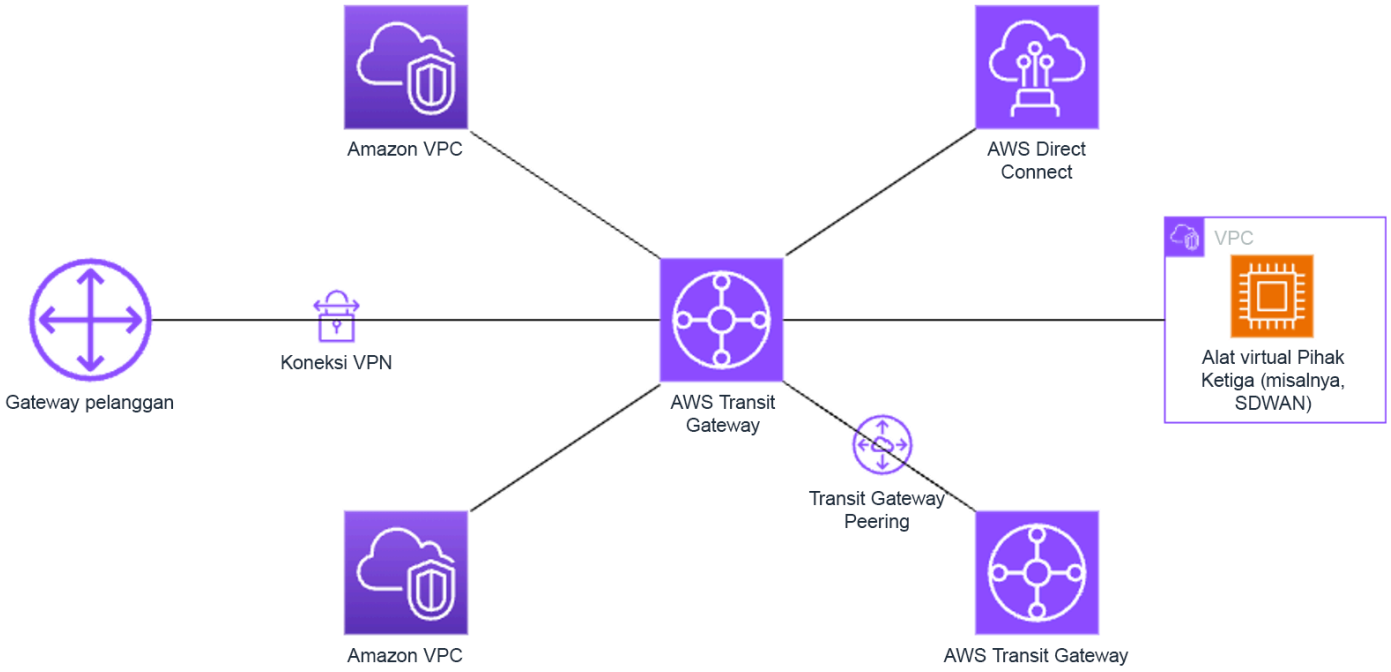
- AWS re: invent 2018: VPC Desain Lanjutan dan Kemampuan Baru untuk Amazon VPC (03) NET3
- AWS Re: invent 2019: arsitektur AWS Transit Gateway referensi untuk banyak orang (06-R1) VPCs NET4
- AWS Re:invent 2023: AWS Siap untuk apa selanjutnya? Merancang jaringan untuk pertumbuhan dan fleksibilitas (NET310)

REL02-BP04 Lebih suka topologi daripada mesh hub-and-spoke many-to-many

Saat menghubungkan beberapa jaringan pribadi, seperti Virtual Private Clouds (VPCs) dan jaringan lokal, pilihlah hub-and-spoke topologi daripada jaringan tersambung. Tidak seperti topologi meshed, di mana setiap jaringan terhubung langsung ke yang lain dan meningkatkan kompleksitas dan

overhead manajemen, hub-and-spoke arsitektur memusatkan koneksi melalui satu hub. Sentralisasi ini menyederhanakan struktur jaringan dan meningkatkan operabilitas, skalabilitas, dan kontrolnya.

AWS Transit Gateway adalah layanan terkelola, terukur, dan sangat tersedia yang dirancang untuk pembangunan hub-and-spoke jaringan di. AWS Layanan ini berfungsi sebagai hub pusat jaringan Anda yang menyediakan segmentasi jaringan, perutean tersentralisasi, dan koneksi yang disederhanakan ke lingkungan cloud dan on-premise. Gambar berikut menggambarkan bagaimana Anda dapat menggunakan AWS Transit Gateway untuk membangun hub-and-spoke topologi Anda.




Anti-pola umum:

- Anda terlalu memperumit kebijakan perutean dalam hub-and-spoke arsitektur, yang mengurangi efisiensi jaringan dan mempersulit pemecahan masalah dan manajemen proaktif.
- Segmentasi berbasis perutean yang tidak memadai di dalam hub dapat menyebabkan kerentanan, yang berpotensi untuk mengekspos jaringan ke akses yang tidak sah.
- Tanpa optimalisasi yang cermat, lalu lintas yang dirutekan melalui hub dapat menyebabkan biaya transfer data yang menjadi lebih tinggi, terutama untuk lalu lintas yang melewati beberapa Zona Ketersediaan dan Wilayah. Strategi manajemen lalu lintas yang efektif sangat penting untuk mengontrol pengeluaran.

Manfaat membangun praktik terbaik ini: Ketika jumlah jaringan yang terhubung meningkat, manajemen dan perluasan konektivitas tersambung menjadi semakin menantang. AWS Transit Gateway menawarkan hub terkelola yang terukur dan andal untuk konstruksi dan pengoperasian hub-and-spoke topologi Anda. Saat Anda menggunakan AWS Transit Gateway, Anda dapat membuat koneksi dan memusatkan perutean lalu lintas di beberapa jaringan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Rencanakan jaringan Anda.
- Buat Anda AWS Transit Gateway.
- Lampirkan AndaVPCs.
- Jika diperlukan, buat VPN koneksi atau gateway Direct Connect dan kaitkan dengan Transit Gateway.
- Tentukan cara lalu lintas dirutekan antara koneksi yang terhubung VPCs dan lainnya melalui konfigurasi tabel rute Transit Gateway Anda.
- Gunakan Amazon CloudWatch untuk memantau dan menyesuaikan konfigurasi yang diperlukan untuk pengoptimalan kinerja dan biaya.

Sumber daya

Dokumen terkait:

- Apa Itu Gateway Transit?
- Membangun Infrastruktur Multi VPC AWS Jaringan yang Skalabel dan Aman
- Membangun jaringan global menggunakan AWS Transit Gateway peering Inter-Region
- Pilihan Konektivitas Amazon Virtual Private Cloud
- APNMitra: mitra yang dapat membantu merencanakan jaringan Anda
- AWS Marketplace untuk Infrastruktur Jaringan

Video terkait:

- AWS re:invent 2023 - yayasan jaringan AWS
- AWS Re: invent 2023 - Desain canggih VPC dan kemampuan baru

REL02-BP05 Menerapkan rentang alamat IP pribadi yang tidak tumpang tindih di semua ruang alamat pribadi tempat mereka terhubung

Rentang alamat IP masing-masing Anda tidak VPCs boleh tumpang tindih saat diintip, terhubung melalui Transit Gateway, atau terhubung. VPN Hindari konflik alamat IP antara lingkungan VPC dan lokal atau dengan penyedia cloud lain yang Anda gunakan. Selain itu, Anda harus memiliki cara untuk mengalokasikan rentang alamat IP privat ketika dibutuhkan. Sistem manajemen alamat IP (IPAM) dapat membantu mengotomatisasi ini.

Hasil yang diinginkan:

- Tidak ada konflik rentang alamat IP antaraVPCs, lingkungan lokal, atau penyedia cloud lainnya.
- Manajemen alamat IP yang tepat akan memungkinkan penskalaan infrastruktur jaringan yang lebih mudah untuk mengakomodasi pertumbuhan dan perubahan persyaratan jaringan.

Anti-pola umum:

- Menggunakan rentang IP yang sama VPC seperti yang Anda miliki di tempat, di jaringan perusahaan Anda, atau penyedia cloud lainnya
- Tidak melacak rentang IP yang VPCs digunakan untuk menyebarkan beban kerja Anda.
- Mengandalkan proses manajemen alamat IP manual, seperti spreadsheet.
- CIDRBlok berukuran berlebih atau kurang, yang mengakibatkan pemborosan alamat IP atau ruang alamat yang tidak mencukupi untuk beban kerja Anda.

Manfaat menerapkan praktik terbaik ini: Perencanaan aktif jaringan Anda akan memastikan bahwa Anda tidak memiliki beberapa kejadian alamat IP yang sama di jaringan-jaringan yang saling terhubung. Ini akan mencegah timbulnya masalah perutean di bagian beban kerja yang menggunakan aplikasi yang berbeda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

ManfaatkanIPAM, seperti Amazon VPC IP Address Manager, untuk memantau dan mengelola CIDR penggunaan Anda. IPAMsBeberapa juga tersedia dari AWS Marketplace. Evaluasi potensi penggunaan Anda AWS, tambahkan CIDR rentang yang adaVPCs, dan buat VPCs untuk memungkinkan pertumbuhan penggunaan yang direncanakan.

Langkah-langkah implementasi

- Tangkap CIDR konsumsi saat ini (misalnya, VPCs dan subnet).
  - Gunakan API operasi layanan untuk mengumpulkan CIDR konsumsi saat ini.
  - Gunakan Manajer Alamat VPC IP Amazon untuk menemukan sumber daya.
- Catat penggunaan subnet Anda saat ini.
  - Gunakan API operasi layanan untuk mengumpulkan subnet per VPC di setiap Wilayah.
  - Gunakan Manajer Alamat VPC IP Amazon untuk menemukan sumber daya.
- Catat penggunaan saat ini.
- Tentukan apakah Anda telah membuat rentang IP yang tumpang tindih.
- Hitung kapasitas cadangan.
- Identifikasi rentang IP yang tumpang tindih. Anda dapat bermigrasi ke rentang alamat baru atau mempertimbangkan untuk menggunakan teknik seperti NATGateway pribadi atau AWS PrivateLinkjika Anda perlu menghubungkan rentang yang tumpang tindih.

Sumber daya

Praktik-praktik terbaik terkait:

- Melindungi jaringan

Dokumen terkait:

- APNMitra: mitra yang dapat membantu merencanakan jaringan Anda
- AWS Marketplace untuk Infrastruktur Jaringan
- Laporan Resmi Pilihan Konektivitas Amazon Virtual Private Cloud
- Konektivitas jaringan ketersediaan tinggi (HA) beberapa pusat data
- Menghubungkan Jaringan dengan Rentang IP yang Tumpang Tindih
- Apa itu AmazonVPC?
- Apa ituIPAM?

Video terkait:

- AWS RE: invent 2023 - Desain canggih VPC dan kemampuan baru

- AWS Re: invent 2019: arsitektur AWS Transit Gateway referensi untuk banyak orang VPCs
- AWS Re:invent 2023 - Siap untuk apa selanjutnya? Merancang jaringan untuk pertumbuhan dan fleksibilitas
- AWS re:invent 2021 - {New Launch} Kelola alamat IP Anda dalam skala besar AWS

## Arsitektur beban kerja

Pertanyaan

- REL3. Bagaimana cara mendesain arsitektur layanan beban kerja Anda?
- REL4. Bagaimana cara mendesain interaksi di sistem terdistribusi untuk mencegah kegagalan?
- REL5. Bagaimana cara mendesain interaksi di sistem terdistribusi untuk memitigasi atau bertahan dari kegagalan?

### REL3. Bagaimana cara mendesain arsitektur layanan beban kerja Anda?

Bangun beban kerja yang sangat skalabel dan andal menggunakan arsitektur berorientasi layanan (SOA) atau arsitektur layanan mikro. Arsitektur berorientasi layanan (SOA) adalah praktik membuat komponen perangkat lunak dapat digunakan kembali melalui antarmuka layanan. Arsitektur layanan mikro melangkah lebih jauh untuk membuat komponen menjadi lebih kecil dan lebih sederhana.

Praktik terbaik

- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL03-BP02 Membangun layanan yang berfokus pada domain dan fungsionalitas bisnis tertentu
- REL03-BP03 Memberikan kontrak layanan per API

#### REL03-BP01 Pilih cara mengelompokkan beban kerja Anda

Segmentasi beban kerja penting saat menentukan persyaratan ketahanan aplikasi Anda. Arsitektur monolitik harus dihindari jika memungkinkan. Sebagai gantinya, pertimbangkan dengan cermat komponen-komponen aplikasi mana yang dapat dipecah menjadi layanan-layanan mikro. Tergantung pada persyaratan aplikasi Anda, ini mungkin berakhir menjadi kombinasi arsitektur berorientasi layanan (SOA) dengan layanan mikro jika memungkinkan. Beban kerja yang mampu berada dalam kondisi stateless akan lebih mampu di-deploy sebagai layanan mikro.

Hasil yang diinginkan: Beban kerja harus dapat didukung, dapat diskalakan, dan digabungkan secara longgar hingga selonggar mungkin.

Saat membuat pilihan tentang cara melakukan segmentasi terhadap beban kerja Anda, seimbangkan manfaat dengan kerumitannya. Hal yang tepat untuk produk baru yang mengejar jadwal peluncuran pertama akan berbeda dengan hal yang dibutuhkan oleh sebuah beban kerja yang dibangun untuk diskalakan dari awal. Saat memfaktor ulang sebuah monolit yang ada, Anda harus mempertimbangkan seberapa baik aplikasi akan mendukung dekomposisi menuju kondisi stateless. Dengan memecah layanan menjadi bagian-bagian yang lebih kecil, tim-tim kecil yang diberi tanggung jawab khusus akan dapat mengembangkan dan mengelolanya. Namun demikian, layanan yang lebih kecil dapat menimbulkan kompleksitas yang semakin tinggi, antara lain peningkatan latensi, debugging yang lebih kompleks, dan peningkatan beban operasional.

Anti-pola umum:

- Death Star layanan mikro adalah situasi saat komponen atomik menjadi sangat saling bergantung sehingga kegagalan salah satu komponen akan menghasilkan kegagalan yang jauh lebih besar, sehingga komponen ini pun menjadi kaku dan rapuh seperti monolit.

Manfaat menerapkan praktik ini:

- Segmen-segmen yang lebih spesifik akan menghasilkan ketangkasan, fleksibilitas organisasi, dan skalabilitas yang lebih besar.
- Dampak gangguan layanan yang berkurang.
- Komponen-komponen aplikasi mungkin memiliki persyaratan ketersediaan yang berbeda-beda, yang dapat didukung oleh segmentasi yang lebih kecil (atomic segmentation).
- Tanggung jawab yang ditentukan dengan baik untuk tim yang mendukung beban kerja.

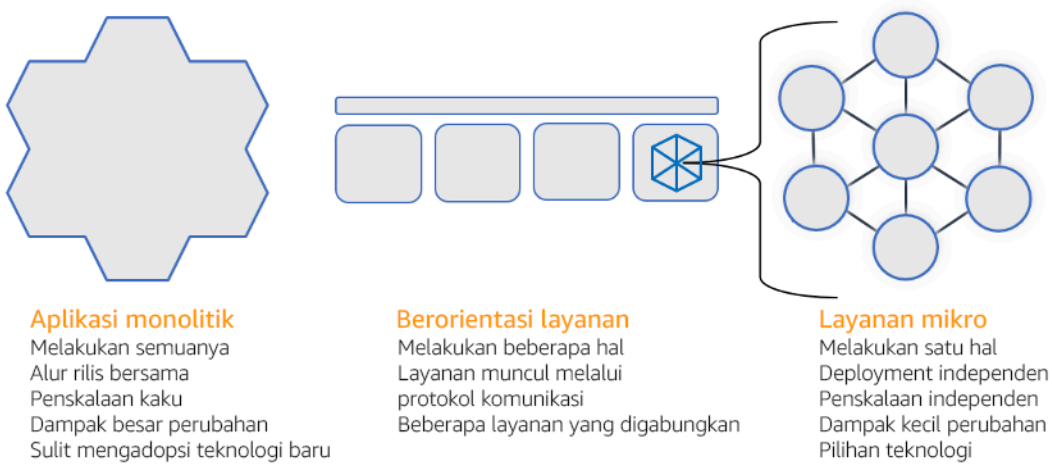
Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pilih jenis arsitektur berdasarkan cara beban kerja disegmentasikan. Pilih arsitektur SOA atau layanan mikro (atau dalam beberapa kasus yang jarang terjadi, arsitektur monolitik). Bahkan jika Anda memilih untuk memulai dengan arsitektur monolit, Anda harus memastikan bahwa itu modular dan pada akhirnya dapat berkembang ke SOA atau layanan mikro sebagai skala produk Anda dengan adopsi pengguna. SOAdan layanan mikro menawarkan segmentasi yang lebih kecil, yang

lebih disukai sebagai arsitektur modern yang dapat diskalakan dan andal, tetapi ada trade-off yang perlu dipertimbangkan, terutama ketika menerapkan arsitektur layanan mikro.

Salah satu tarik ulur utama adalah Anda sekarang memiliki sebuah arsitektur komputasi terdistribusi yang dapat mempersulit dalam memenuhi kebutuhan latensi pengguna dan ada kerumitan tambahan dalam proses debugging dan penelusuran interaksi pengguna. Anda dapat menggunakan AWS X-Ray untuk membantu Anda memecahkan masalah ini. Efek lain yang perlu dipertimbangkan adalah peningkatan kompleksitas operasional seiring meningkatnya jumlah aplikasi yang Anda kelola, yang memerlukan deployment terhadap banyak komponen independensi.




Arsitektur monolitik yang berorientasi layanan, dan arsitektur layanan mikro

Langkah-langkah implementasi

- Tentukan arsitektur yang sesuai untuk memfaktor ulang atau membangun aplikasi Anda. SOAdan layanan mikro menawarkan segmentasi yang lebih kecil, yang lebih disukai sebagai arsitektur modern yang dapat diskalakan dan andal. SOAdapat menjadi kompromi yang baik untuk mencapai segmentasi yang lebih kecil sambil menghindari beberapa kompleksitas layanan mikro. Untuk detail selengkapnya, lihat Kompromi Layanan Mikro.
- Jika dapat diterima beban kerja dan didukung organisasi, maka Anda harus menggunakan sebuah arsitektur layanan mikro untuk mencapai ketangkasan dan keandalan terbaik. Untuk detail selengkapnya, lihat Menerapkan Layanan Mikro di AWS.
- Pertimbangkan untuk mengikuti pola Strangler Fig berikut untuk memfaktorkan ulang monolit menjadi komponen yang lebih kecil. Hal ini melibatkan penggantian komponen aplikasi tertentu secara bertahap dengan aplikasi dan layanan baru. AWS Migration Hub Refactor Spaces bertindak sebagai titik awal untuk memfaktor ulang secara bertahap. Untuk mendapatkan detail

selengkapnya, lihat Memigrasikan beban kerja lama di on-premise dengan lancar menggunakan pola strangler.

- Menerapkan layanan mikro mungkin memerlukan mekanisme penemuan layanan untuk memungkinkan layanan terdistribusi ini berkomunikasi satu sama lain. AWS App Meshdapat digunakan dengan arsitektur berorientasi layanan untuk memberikan penemuan dan akses layanan yang andal. AWS Cloud Mapjuga dapat digunakan untuk penemuan layanan DNS berbasis dinamis.
- Jika Anda bermigrasi dari monolit ke, Amazon SOA MQ dapat membantu menjembatani kesenjangan sebagai bus layanan saat mendesain ulang aplikasi lama di cloud.
- Untuk monolit yang ada dengan satu basis data bersama, pilihlah cara mengatur ulang data menjadi segmen yang lebih kecil. Segmentasi ini dapat didasarkan pada unit bisnis, pola akses, atau struktur data. Pada titik ini dalam proses refactoring, Anda harus memilih untuk bergerak maju dengan jenis database relasional atau non-relasional (No). SQL Untuk detail selengkapnya, lihat Dari SQL ke No SQL.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- REL03-BP02 Membangun layanan yang berfokus pada domain dan fungsionalitas bisnis tertentu

Dokumen terkait:

- Amazon API Gateway: Mengonfigurasi REST API Menggunakan Buka API
- Apa itu Arsitektur Berorientasi Layanan?
- Konteks Terikat (pola sentral di Desain yang Didorong Domain)
- Menerapkan Layanan Mikro pada AWS
- Kompensasi Layanan Mikro
- Layanan mikro - definisi dari istilah arsitektur baru ini
- Microservices pada AWS
- Apa itu AWS App Mesh?

Contoh terkait:

- Lokakarya Modernisasi Aplikasi Iteratif

Video terkait:

- Memberikan Keunggulan dengan Layanan Mikro AWS

REL03-BP02 Membangun layanan yang berfokus pada domain dan fungsionalitas bisnis tertentu

Arsitektur berorientasi layanan (SOA) mendefinisikan layanan dengan fungsi yang digambarkan dengan baik yang ditentukan oleh kebutuhan bisnis. Layanan mikro menggunakan model domain dan konteks yang dibatasi untuk menarik batas-batas layanan di sepanjang batas konteks bisnis. Berfokus pada domain dan fungsionalitas bisnis dapat membantu tim untuk menentukan persyaratan keandalan sendiri untuk layanan mereka. Konteks yang dibatasi mengisolasi dan memisahkan logika bisnis, sehingga memungkinkan tim memiliki penalaran yang lebih baik tentang bagaimana menangani kegagalan.

Hasil yang diinginkan: Para rekayasawan dan pemangku kepentingan bisnis bersama-sama menetapkan konteks yang dibatasi dan menggunakannya untuk merancang sebuah sistem sebagai layanan yang memenuhi fungsi-fungsi bisnis tertentu. Tim-tim ini menggunakan praktik-praktik yang telah lazim seperti event storming untuk menentukan persyaratan. Aplikasi-aplikasi baru dirancang sebagai batasan-batasan layanan yang ditetapkan dengan baik dan penggabungan longgar. Monolit yang ada didekomposisi menjadi konteks terbatas dan desain sistem bergerak menuju atau arsitektur layanan mikro. SOA Ketika arsitektur monolit difaktorkan ulang, pendekatan-pendekatan lazim yang ditetapkan seperti konteks gelembung dan pola penguraian monolit diterapkan.

Layanan-layanan yang berorientasi domain dijalankan sebagai satu atau beberapa proses yang statusnya tidak sama. Layanan-layanan tersebut secara independen memberikan respons terhadap fluktuasi permintaan yang terjadi dan menangani skenario kesalahan dengan berpatokan pada persyaratan khusus domain.

Anti-pola umum:

- Tim dibentuk berdasarkan domain-domain teknis tertentu seperti UI dan UX, perangkat lunak perantara (middleware), atau basis data, tidak dibentuk berdasarkan domain bisnis tertentu.
- Aplikasi melibatkan tanggung jawab domain. Layanan-layanan yang mencakup konteks yang dibatasi bisa lebih sulit untuk dipelihara, memerlukan upaya pengujian yang lebih besar, dan memerlukan banyak tim domain untuk berpartisipasi dalam pembaruan perangkat lunak.

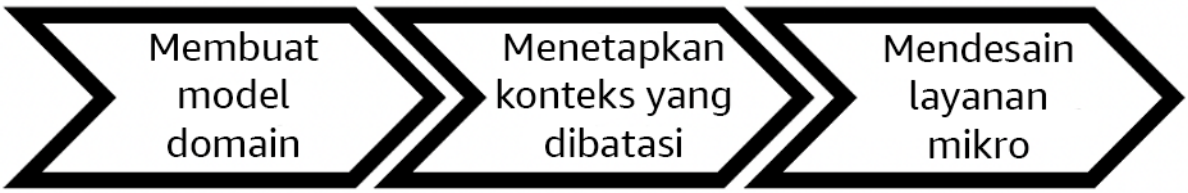
- Dependensi domain, seperti pustaka entitas domain, dibagikan di seluruh layanan sehingga adanya perubahan terhadap satu domain layanan mengharuskan dilakukannya perubahan pada domain layanan lainnya
- Kontrak layanan dan logika bisnis tidak mengekspresikan entitas dalam bahasa domain yang umum dan konsisten, sehingga dapat menghasilkan lapisan-lapisan terjemahan yang membuat sistem semakin rumit dan upaya debugging semakin meningkat.

Manfaat menerapkan praktik terbaik ini: Aplikasi dirancang sebagai layanan independen yang dibatasi oleh domain bisnis dan menggunakan bahasa bisnis yang umum. Layanan-layanan dapat diuji dan dapat di-deploy secara independen. Layanan-layanan memenuhi persyaratan ketahanan khusus domain untuk domain yang diterapkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Domain-driven design (DDD) adalah pendekatan dasar merancang dan membangun perangkat lunak di sekitar domain bisnis. Menggunakan kerangka kerja yang ada sekarang akan memudahkan Anda dalam membuat layanan yang berfokus pada domain bisnis. Saat bekerja dengan aplikasi monolitik yang ada, Anda dapat memanfaatkan pola-pola penguraian yang menyediakan teknik-teknik yang sudah lazim dan ditetapkan untuk melakukan modernisasi terhadap aplikasi hingga menjadi layanan.




Desain berbasis domain

Langkah-langkah implementasi

- Tim dapat mengadakan event storming untuk mengidentifikasi peristiwa, perintah, agregat, dan domain secara cepat dalam format catatan tempel ringan.
- Setelah entitas dan fungsi domain dibentuk dalam sebuah konteks domain, Anda dapat membagi domain Anda menjadi layanan menggunakan konteks terikat, di mana entitas yang berbagi fitur dan atribut serupa dikelompokkan bersama. Dengan model yang dibagi-bagi ke dalam konteks, muncul sebuah templat untuk membatasi layanan mikro.

    - Misalnya, entitas-entitas yang dalam situs web Amazon.com dapat meliputi paket, pengantaran, jadwal, harga, diskon, dan mata uang.
    - Paket, pengantaran, dan jadwal dikelompokkan ke dalam konteks pengiriman, sedangkan harga, diskon, dan mata uang dikelompokkan ke dalam konteks harga.
- Mengurai monolit menjadi layanan mikro menguraikan pola untuk memfaktor ulang layanan mikro. Menggunakan pola-pola penguraian berdasarkan kemampuan bisnis, subdomain, atau transaksi selaras dengan pendekatan berbasis domain.
- Teknik taktis seperti konteks gelembung memungkinkan Anda untuk memperkenalkan aplikasi yang ada atau lama tanpa penulisan ulang DDD di muka dan komitmen penuh untuk. DDD Dalam sebuah pendekatan konteks gelembung, sebuah konteks terikat kecil dibuat dengan menggunakan pemetaan dan koordinasi layanan, atau lapisan anti-korupsi, yang melindungi model domain yang baru didefinisikan dari pengaruh eksternal.

Setelah tim melakukan analisis domain dan entitas yang ditentukan serta kontrak layanan, mereka dapat memanfaatkan AWS layanan untuk mengimplementasikan desain berbasis domain mereka sebagai layanan berbasis cloud.

- Mulailah pengembangan Anda dengan menentukan pengujian yang menggunakan aturan-aturan bisnis domain Anda. Pengembangan berbasis tes (TDD) dan pengembangan berbasis perilaku (BDD) membantu tim menjaga layanan tetap fokus pada pemecahan masalah bisnis.
- Pilih layanan AWS yang paling sesuai dengan persyaratan domain bisnis dan arsitektur layanan mikro Anda:
    - AWS Tanpa server memungkinkan tim Anda fokus pada logika domain tertentu alih-alih mengelola server dan infrastruktur.
    - Kontainer di AWS menyederhanakan pengelolaan infrastruktur Anda, sehingga Anda dapat fokus pada persyaratan domain Anda.
    - Basis data yang dibuat khusus dapat membantu Anda mencocokkan persyaratan domain Anda dengan jenis basis data yang paling sesuai.
- Membangun arsitektur heksagonal di AWS menguraikan kerangka kerja untuk membangun logika bisnis menjadi layanan yang bekerja mundur dari domain bisnis untuk memenuhi persyaratan fungsional dan kemudian melampirkan adaptor integrasi. Pola yang memisahkan detail antarmuka dari logika bisnis dengan AWS layanan membantu tim fokus pada fungsionalitas domain dan meningkatkan kualitas perangkat lunak.

Sumber daya

Praktik-praktik terbaik terkait:

- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL03-BP03 Memberikan kontrak layanan per API

Dokumen terkait:

- AWS Layanan Mikro
- Menerapkan Layanan Mikro pada AWS
- Cara memecah Monolit menjadi Layanan-layanan Mikro
- Memulai DDD ketika Dikelilingi oleh Sistem Warisan
- Desain Berbasis Domain: Mengatasi Kompleksitas di Dalam Inti Perangkat Lunak
- Membangun arsitektur heksagonal di AWS
- Menguraikan monolit menjadi layanan mikro
- Event Storming
- Pesan Antara Konteks-Konteks Terikat
- Layanan mikro
- Pengembangan berbasis pengujian
- Pengembangan berbasis perilaku

Contoh terkait:

- Merancang Cloud Native Microservices di AWS (dariDDD/EventStormingWorkshop)

Alat terkait:

- AWS Cloud Database
- Tanpa server di AWS
- Kontainer di AWS

REL03-BP03 Memberikan kontrak layanan per API

Kontrak layanan adalah perjanjian terdokumentasi antara API produsen dan konsumen yang didefinisikan dalam definisi yang dapat dibaca mesinAPI. Strategi pembuatan versi kontrak memungkinkan konsumen untuk terus menggunakan yang ada API dan memigrasikan aplikasi mereka ke yang lebih baru API ketika mereka siap. Deployment oleh produsen dapat terjadi kapan saja, selama kontrak dipatuhi. Tim layanan dapat menggunakan tumpukan teknologi pilihan mereka untuk memenuhi API kontrak.

Hasil yang diinginkan: Aplikasi yang dibangun dengan arsitektur berorientasi layanan atau layanan mikro dapat beroperasi secara independen sambil memiliki ketergantungan runtime terintegrasi. Perubahan yang diterapkan ke API konsumen atau produsen tidak mengganggu stabilitas sistem secara keseluruhan ketika kedua belah pihak mengikuti kontrak bersamaAPI. Komponen yang berkomunikasi melalui layanan APIs dapat melakukan rilis fungsional independen, peningkatan ke dependensi runtime, atau gagal ke situs pemulihan bencana (DR) dengan sedikit atau tanpa dampak satu sama lain. Selain itu, layanan-layanan diskret dapat menyesuaikan skala secara independen dengan menyerap permintaan sumber daya tanpa mengharuskan layanan lain untuk menyesuaikan skala (menskalakan) secara serempak.

Anti-pola umum:

- Membuat layanan APIs tanpa skema yang diketik dengan kuat. Hal ini mengakibatkan hal APIs itu tidak dapat digunakan untuk menghasilkan API binding dan payload yang tidak dapat divalidasi secara terprogram.
- Tidak mengadopsi strategi pembuatan versi, yang memaksa API konsumen untuk memperbarui dan merilis atau gagal ketika kontrak layanan berkembang.
- Pesan-pesan kesalahan yang membocorkan detail implementasi layanan yang mendasari, bukan menggambarkan kegagalan integrasi dalam bahasa dan konteks domain.
- Tidak menggunakan API kontrak untuk mengembangkan kasus uji dan API implementasi tiruan untuk memungkinkan pengujian independen komponen layanan.

Manfaat membangun praktik terbaik ini: Sistem terdistribusi yang terdiri dari komponen yang berkomunikasi melalui kontrak API layanan dapat meningkatkan keandalan. Pengembang dapat menangkap potensi masalah di awal proses pengembangan dengan pemeriksaan tipe selama kompilasi untuk memverifikasi bahwa permintaan dan tanggapan mengikuti API kontrak dan bidang wajib ada. APIkontrak menyediakan antarmuka pendokumentasian diri yang jelas untuk APIs dan penyedia interoperabilitas yang lebih baik antara sistem yang berbeda dan bahasa pemrograman.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Setelah Anda mengidentifikasi domain bisnis dan menentukan segmentasi beban kerja Anda, Anda dapat mengembangkan layanan Anda. APIs Pertama, tentukan kontrak layanan yang dapat dibaca mesin untukAPIs, dan kemudian terapkan strategi pembuatan versi. API Ketika Anda siap untuk mengintegrasikan layanan melalui protokol umum seperti, REST GraphQL, atau peristiwa asinkron, Anda dapat menggabungkan AWS layanan ke dalam arsitektur Anda untuk mengintegrasikan komponen Anda dengan kontrak yang diketik dengan kuat. API

AWS layanan untuk API kontras layanan

Gabungkan AWS layanan termasuk Amazon API Gateway AWS AppSync, dan Amazon EventBridge ke dalam arsitektur Anda untuk menggunakan kontrak API layanan dalam aplikasi Anda. Amazon API Gateway membantu Anda berintegrasi dengan AWS layanan asli langsung dan layanan web lainnya. APIGateway mendukung APIspesifikasi dan pembuatan versi Terbuka. AWS AppSync adalah titik akhir GraphQL terkelola yang Anda konfigurasikan dengan mendefinisikan skema GraphQL untuk menentukan antarmuka layanan untuk kueri, mutasi, dan langganan. Amazon EventBridge menggunakan skema acara untuk menentukan peristiwa dan menghasilkan binding kode untuk acara Anda.

Langkah-langkah implementasi

- Pertama, tentukan kontrak untuk AndaAPI. Kontrak akan mengekspresikan kemampuan dan API juga mendefinisikan objek dan bidang data yang diketik dengan kuat untuk API input dan output.
- Saat Anda mengonfigurasi APIs di API Gateway, Anda dapat mengimpor dan mengekspor API Spesifikasi Terbuka untuk titik akhir Anda.
  - Mengimpor API definisi Terbuka menyederhanakan pembuatan Anda API dan dapat diintegrasikan dengan AWS infrastruktur sebagai alat kode seperti dan. AWS Serverless Application ModelAWS Cloud Development Kit (AWS CDK)
  - Mengekspor API definisi menyederhanakan integrasi dengan alat API pengujian dan memberikan spesifikasi integrasi kepada konsumen layanan.
- Anda dapat menentukan dan mengelola APIs GraphQL AWS AppSync dengan mendefinisikan file skema GraphQL untuk menghasilkan antarmuka kontrak Anda dan menyederhanakan interaksi dengan model kompleks, beberapa tabel database, atau layanan lama. REST

- AWS Amplifyproyek yang terintegrasi dengan AWS AppSync menghasilkan file JavaScript kueri yang diketik kuat untuk digunakan dalam aplikasi Anda serta pustaka klien AWS AppSync GraphQL untuk tabel Amazon DynamoDB.
- Saat Anda menggunakan peristiwa layanan dari Amazon EventBridge, peristiwa mematuhi skema yang sudah ada di registri skema atau yang Anda tentukan dengan Spesifikasi TerbukaAPI. Dengan sebuah skema yang ditentukan dalam registri tersebut, Anda juga dapat menghasilkan pengikatan klien (client binding) dari kontrak skema tersebut untuk mengintegrasikan kode Anda dengan peristiwa.
- Memperluas atau versi AndaAPI. Memperluas API adalah opsi yang lebih sederhana saat menambahkan bidang yang dapat dikonfigurasi dengan bidang opsional atau nilai default untuk bidang wajib.
  - JSONkontrak berbasis untuk protokol seperti dan REST GraphQL dapat menjadi cocok untuk perpanjangan kontrak.
  - XMLKontrak berbasis untuk protokol seperti SOAP harus diuji dengan konsumen jasa untuk menentukan kelayakan perpanjangan kontrak.
- Saat membuat versiAPI, pertimbangkan untuk menerapkan versi proxy di mana fasad digunakan untuk mendukung versi sehingga logika dapat dipertahankan dalam satu basis kode.
  - Dengan API Gateway Anda dapat menggunakan pemetaan permintaan dan respons untuk menyederhanakan penyerapan perubahan kontrak dengan membuat fasad untuk memberikan nilai default untuk bidang baru atau untuk menghapus bidang yang dihapus dari permintaan atau respons. Dengan pendekatan ini, layanan-layanan yang mendasari dapat mempertahankan satu basis kode tunggal.

Sumber daya

Praktik-praktik terbaik terkait:

- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL03-BP02 Membangun layanan yang berfokus pada domain dan fungsionalitas bisnis tertentu
- REL04-BP02 Menerapkan dependensi yang digabungkan secara longgar
- REL05-BP03 Kontrol dan batasi panggilan coba lagi
- REL05-BP05 Mengatur batas waktu klien

Dokumen terkait:

- Apa Itu API (Antarmuka Pemrograman Aplikasi)?
- Menerapkan Layanan Mikro pada AWS
- Kompensasi Layanan Mikro
- Layanan mikro - definisi dari istilah arsitektur baru ini
- Microservices pada AWS
- Bekerja dengan ekstensi API Gateway untuk Buka API
- Terbuka API -Spesifikasi
- GraphQL: Skema dan Jenis
- EventBridge Ikatan kode Amazon

Contoh terkait:

- Amazon API Gateway: Mengonfigurasi REST API Menggunakan Buka API
- Amazon API Gateway ke aplikasi Amazon CRUD DynamoDB menggunakan Open API
- Pola integrasi aplikasi modern di era tanpa server: Integrasi Layanan API Gateway
- Menerapkan versi API Gateway berbasis header dengan Amazon CloudFront
- AWS AppSync: Membangun aplikasi klien

Video terkait:

- Menggunakan Open API in AWS SAM untuk mengelola API Gateway

Alat terkait:

- APIGerbang Amazon
- AWS AppSync
- Amazon EventBridge

## REL4. Bagaimana cara mendesain interaksi di sistem terdistribusi untuk mencegah kegagalan?

Sistem terdistribusi mengandalkan jaringan komunikasi untuk membuat interkoneksi komponen, seperti server atau layanan. Beban kerja Anda harus beroperasi secara andal terlepas latensi

atau hilangnya data yang terjadi di jaringan-jaringan ini. Komponen dari sistem terdistribusi harus beroperasi dengan cara yang tidak secara negatif memengaruhi beban kerja atau komponen-komponen lain. Praktik terbaik ini mencegah kegagalan dan meningkatkan waktu rata-rata antara kegagalan (MTBF).

Praktik terbaik

- REL04-BP01 Identifikasi jenis sistem terdistribusi yang Anda andalkan
- REL04-BP02 Menerapkan dependensi yang digabungkan secara longgar
- REL04-BP03 Lakukan pekerjaan konstan
- REL04-BP04 Jadikan semua tanggapan idempoten

REL04-BP01 Identifikasi jenis sistem terdistribusi yang Anda andalkan

Sistem terdistribusi bisa bersifat sinkron, asinkron, atau batch. Sistem sinkron harus memproses permintaan secepat mungkin dan berkomunikasi satu sama lain dengan membuat permintaan sinkron dan panggilan respons menggunakan protokol HTTP /S,REST, atau remote procedure call (RPC). Sistem tidak selaras berkomunikasi satu sama lain dengan bertukar data secara asinkron melalui sebuah layanan perantara tanpa melakukan penggabungan (coupling) terhadap masing-masing sistem. Sistem batch menerima data input dalam jumlah besar, menjalankan proses-proses data otomatis tanpa campur tangan manusia, dan menghasilkan data output.

Hasil yang diinginkan: Merancang desain sebuah beban kerja yang berinteraksi secara efektif dengan dependensi selaras, tidak selaras, dan batch.

Anti-pola umum:

- Beban kerja menunggu respons dari dependensinya tanpa batas waktu, yang dapat menyebabkan klien beban kerja mengalami habis waktu, tanpa mengetahui apakah permintaannya telah diterima atau tidak.
- Beban kerja menggunakan sebuah rantai sistem dependen yang memanggil satu sama lain dengan selaras. Hal ini mengharuskan setiap sistem untuk tersedia dan berhasil memproses sebuah permintaan sebelum seluruh rantai sistem dapat berhasil, sehingga menyebabkan perilaku dan ketersediaan secara keseluruhan menjadi rapuh.
- Beban kerja berkomunikasi dengan dependensinya secara tak selaras dan mengandalkan konsep pengiriman pesan yang dijamin persis satu kali, padahal sering kali pesan duplikat masih memungkinkan untuk diterima.

- Beban kerja tidak menggunakan alat-alat penjadwalan batch yang sesuai dan memungkinkan pelaksanaan pekerjaan batch yang sama secara bersamaan.

Manfaat menerapkan praktik terbaik ini: Sudah menjadi hal yang umum untuk beban kerja tertentu untuk menerapkan satu atau beberapa gaya komunikasi antara selaras, tak selaras, dan batch. Praktik terbaik ini akan membantu Anda untuk mengidentifikasi kompromi-kompromi berbeda yang terkait dengan setiap gaya komunikasi untuk membuat beban kerja Anda dapat memberikan toleransi terhadap gangguan pada dependensinya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Bagian-bagian berikutnya berisi panduan implementasi umum dan spesifik untuk masing-masing jenis dependensi.

Panduan umum

- Pastikan bahwa tujuan tingkat layanan kinerja dan keandalan (SLOs) yang ditawarkan dependensi Anda memenuhi persyaratan kinerja dan keandalan beban kerja Anda.
- Gunakan layanan observabilitas AWS untuk memantau waktu respons dan tingkat kesalahan untuk memastikan bahwa dependensi Anda sedang menyediakan layanan pada tingkat yang dibutuhkan oleh beban kerja Anda.
- Identifikasi potensi-potensi tantangan yang mungkin dihadapi beban kerja Anda saat berkomunikasi dengan dependensinya. Sistem terdistribusi datang dengan berbagai tantangan yang dapat meningkatkan kompleksitas arsitektur, beban operasional, dan biaya. Tantangan-tantangan yang biasanya dihadapi mencakup latensi, gangguan jaringan, kehilangan data, penskalaan, dan jeda replikasi data.
- Terapkan penanganan dan pencatatan log kesalahan yang kuat untuk membantu Anda memecahkan masalah saat dependensi Anda mengalami masalah.

Dependensi sinkron

Dalam komunikasi yang selaras, beban kerja Anda mengirimkan permintaan ke dependensinya dan memblokir operasi yang menunggu sebuah respons. Ketika dependensinya menerima permintaan, dependensi tersebut akan mencoba menanganinya sesegera mungkin dan mengembalikan respons ke beban kerja Anda. Tantangan besar pada komunikasi selaras adalah jenis komunikasi

ini menyebabkan terjadinya penggabungan temporal, yang mengharuskan beban kerja Anda dan dependensinya untuk tersedia pada saat yang bersamaan. Ketika beban kerja Anda perlu berkomunikasi secara selaras dengan dependensinya, pertimbangkan panduan berikut:

- Beban kerja Anda seharusnya tidak bergantung pada beberapa dependensi selaras untuk melakukan satu fungsi tunggal. Rangkaian dependensi ini akan meningkatkan kerapuhan secara keseluruhan karena semua dependensi di jalurnya harus tersedia agar permintaan berhasil diselesaikan.
- Ketika sebuah dependensi berada dalam kondisi tidak sehat atau tidak tersedia, tentukan penanganan kesalahan dan strategi coba lagi. Hindari penggunaan perilaku bimodal. Perilaku bimodal adalah ketika beban kerja Anda menunjukkan perilaku yang berbeda dalam mode normal dan mode kegagalan. Untuk detail lebih lanjut tentang perilaku bimodal, lihat REL11-BP05 Gunakan stabilitas statis untuk mencegah perilaku bimodal.
- Harap diingat bahwa gagal cepat (fail fast) lebih baik daripada membuat beban kerja Anda menunggu. Misalnya, Panduan Pengembang AWS Lambda menjelaskan cara menangani percobaan ulang dan kegagalan saat Anda menginvokasi fungsi Lambda.
- Tetapkan batas waktu saat beban kerja Anda memanggil dependensinya. Teknik ini menghindari waktu tunggu yang terlalu lama atau waktu tunggu respons tanpa batas. Untuk diskusi bermanfaat tentang masalah ini, lihat Menyetel setelan SDK HTTP permintaan AWS Java untuk aplikasi Amazon DynamoDB yang sadar latensi.
- Minimalkan jumlah panggilan yang dilakukan dari beban kerja Anda ke dependensinya untuk memenuhi satu permintaan tunggal. Memiliki banyak panggilan (chatty) di antara keduanya dapat meningkatkan penggabungan dan latensi.

### Dependensi tidak selaras

Untuk memisahkan beban kerja Anda dari dependensinya secara sementara, keduanya harus berkomunikasi secara tidak selaras. Jika menggunakan pendekatan tidak selaras, beban kerja Anda dapat melanjutkan pemrosesan lain tanpa harus menunggu dependensinya, atau rangkaian dependensinya, untuk mengirimkan sebuah respons.

Ketika beban kerja Anda perlu berkomunikasi secara tidak selaras dengan dependensinya, pertimbangkan panduan berikut:

- Tentukan apakah akan menggunakan perpesanan atau streaming peristiwa berdasarkan kasus penggunaan dan kebutuhan Anda. Layanan perpesanan akan memungkinkan beban kerja Anda untuk berkomunikasi dengan dependensinya dengan mengirim dan menerima pesan melalui

broker pesan. Streaming acara akan memungkinkan beban kerja Anda dan dependensinya untuk menggunakan layanan streaming untuk mempublikasikan dan berlangganan acara, disampaikan sebagai aliran data berkelanjutan, yang perlu diproses sesegera mungkin.

- Perpesanan dan streaming peristiwa menangani pesan secara berbeda sehingga Anda perlu mengambil keputusan-keputusan kompromi berdasarkan:
  - Prioritas pesan: broker pesan dapat memproses pesan prioritas tinggi lebih dahulu dari pesan normal. Dalam streaming peristiwa, semua pesan memiliki prioritas yang sama.
  - Konsumsi pesan: broker pesan memastikan bahwa konsumen menerima pesan. Pemakai streaming peristiwa harus terus melacak pesan terakhir yang telah mereka baca.
  - Pemesanan pesan: dengan pesan, menerima pesan dalam urutan yang tepat mereka dikirim tidak dijamin kecuali Anda menggunakan pendekatan first-in-first-out (FIFO). Streaming peristiwa selalu akan mempertahankan urutan sesuai urutan data ketika dibuat.
  - Penghapusan pesan: dengan layanan pengiriman pesan, konsumen harus menghapus pesan setelah memprosesnya. Layanan streaming peristiwa menambahkan pesan tersebut ke sebuah aliran dan tetap berada di sana sampai periode penyimpanan pesan berakhir. Kebijakan penghapusan ini menjadikan streaming peristiwa cocok untuk pemutaran ulang pesan.
- Tentukan bagaimana beban kerja Anda mengetahui kapan dependensinya menyelesaikan pekerjaan. Sebagai contoh, saat beban kerja Anda menginvokasi fungsi Lambda asinkron, Lambda akan menempatkan peristiwa dalam antrean dan mengembalikan respons sukses tanpa menyertakan informasi tambahan. Setelah pemrosesan selesai, fungsi Lambda dapat mengirim hasilnya ke sebuah tujuan, dapat dikonfigurasi berdasarkan keberhasilan atau kegagalan.
- Bangun beban kerja Anda untuk menangani pesan duplikat dengan memanfaatkan idempotensi. Idempotensi adalah ketika hasil beban kerja Anda tidak berubah bahkan jika beban kerja Anda dihasilkan lebih dari sekali untuk pesan yang sama. Penting untuk menunjukkan bahwa layanan pengiriman pesan atau streaming akan mengirim ulang pesan jika terjadi kegagalan jaringan atau jika ada pengakuan bahwa pesan belum diterima.
- Jika beban kerja Anda tidak mendapatkan respons dari dependensinya, maka permintaan perlu dikirimkan kembali. Pertimbangkan untuk membatasi jumlah percobaan ulang untuk mempertahankan beban kerjaCPU, memori, dan sumber daya jaringan Anda untuk menangani permintaan lainnya. Dokumentasi AWS Lambda menunjukkan cara menangani kesalahan untuk panggilan asinkron.
- Manfaatkan alat-alat observabilitas, debugging, dan pelacakan yang sesuai untuk mengelola dan mengoperasikan komunikasi tidak selaras dari beban kerja Anda dengan dependensinya. Anda dapat menggunakan Amazon CloudWatch untuk memantau layanan pengiriman pesan dan

streaming acara. Anda juga dapat menginstrumentasikan beban kerja Anda dengan AWS X-Ray untuk dengan cepat mendapatkan wawasan untuk pemecahan masalah.

Dependensi Batch

Sistem batch mengambil data input, memulai serangkaian pekerjaan untuk memprosesnya, dan kemudian menghasilkan beberapa data output, tanpa intervensi manual. Tergantung ukuran data, pekerjaan dapat berjalan dari hitungan menit hingga, dalam beberapa kasus, beberapa hari. Ketika beban kerja Anda berkomunikasi dengan dependensinya, pertimbangkan untuk menggunakan panduan berikut:

- Tentukan rentang periode ketika beban kerja Anda harus menjalankan tugas batch. Beban kerja Anda dapat mengatur pola pengulangan untuk menginvokasi sebuah sistem batch, misalnya, setiap jam atau pada akhir setiap bulan.
- Tentukan lokasi input data dan output data yang diproses. Pilih layanan penyimpanan, seperti Amazon Simple Storage Services (Amazon S3), Amazon Elastic File System (AmazonEFS), dan Amazon FSx for Lustre, yang memungkinkan beban kerja Anda membaca dan menulis file dalam skala besar.
- Jika beban kerja Anda perlu memanggil beberapa pekerjaan batch, Anda dapat memanfaatkan AWS Step Functionsuntuk menyederhanakan orkestrasi pekerjaan batch yang berjalan di dalam atau di tempat. AWS Proyek sampel ini mendemonstrasikan orkestrasi pekerjaan batch dengan menggunakan Step Functions, AWS Batch, dan Lambda.
- Lakukan pemantauan terhadap pekerjaan batch untuk mencari kelainan, seperti pekerjaan yang membutuhkan waktu lebih lama dari yang seharusnya. Anda dapat menggunakan alat seperti CloudWatchWawasan Kontainer untuk memantau AWS Batch lingkungan dan pekerjaan. Dalam keadaan ini, beban kerja Anda akan menghentikan pekerjaan berikutnya dari awal dan memberitahukan pengecualian kepada staf yang relevan.

Sumber daya

Dokumen terkait:

- AWS Cloud Operasi: Pemantauan dan Pengamatan
- Amazon Builders' Library: Tantangan dengan sistem terdistribusi
- REL11-BP05 Gunakan stabilitas statis untuk mencegah perilaku bimodal

- AWS Lambda Panduan Pengembang: Penanganan kesalahan dan percobaan ulang otomatis di AWS Lambda
- Menyetel pengaturan SDK HTTP permintaan AWS Java untuk aplikasi Amazon DynamoDB yang sadar latensi
- Perpesanan AWS
- Apa itu data streaming?
- AWS Lambda Panduan Pengembang: Pemanggilan asinkron
- Layanan Antrian Sederhana AmazonFAQ: FIFO antrian
- Panduan Pengembang Amazon Kinesis Data Streams: Menangani Rekaman Duplikat
- Panduan Pengembang Layanan Antrian Sederhana Amazon: CloudWatch Metrik yang tersedia untuk Amazon SQS
- Panduan Pengembang Amazon Kinesis Data Streams: Memantau Layanan Amazon Kinesis Data Streams dengan Amazon CloudWatch
- AWS X-Ray Panduan Pengembang: AWS X-Ray konsep
- AWS Sampel pada GitHub: Fungsi AWS langkah Aplikasi Orkestrator Kompleks
- AWS Batch Panduan Pengguna: AWS Batch CloudWatch Wawasan Kontainer

Video terkait:

- AWS Summit SF 2022 - Observabilitas tumpukan penuh dan pemantauan aplikasi dengan AWS (0) COP31

Alat terkait:

- Amazon CloudWatch
- CloudWatch Log Amazon
- AWS X-Ray
- Amazon Simple Storage Service (Amazon S3)
- Amazon Elastic File System (AmazonEFS)
- Amazon FSx untuk Lustre
- AWS Step Functions
- AWS Batch

REL04-BP02 Menerapkan dependensi yang digabungkan secara longgar

Dependensi seperti sistem pengantrean, sistem streaming, alur kerja, dan penyeimbang beban digabungkan secara longgar. Penggabungan longgar membantu memisahkan perilaku suatu komponen dari komponen lainnya yang bergantung pada komponen tersebut, sehingga meningkatkan ketahanan dan ketangkasan.

Memisahkan dependensi, seperti sistem antrean, sistem streaming, dan alur kerja, membantu meminimalkan dampak perubahan atau kegagalan pada suatu sistem. Pemisahan ini akan mengisolasi perilaku komponen dari mempengaruhi orang lain yang bergantung padanya, meningkatkan ketahanan dan kelincahan.

Dalam sistem penggabungan erat (tightly coupled), perubahan pada satu komponen dapat menyebabkan perubahan pada komponen lain yang bergantung padanya, yang mengakibatkan penurunan performa di semua komponen. Penggabungan longgar menghilangkan dependensi ini, sehingga komponen-komponen yang bergantung hanya perlu mengetahui antarmuka versi terbaru dan yang dipublikasikan. Mengimplementasikan penggabungan longgar antar dependensi akan memisahkan kegagalan pada salah satu dependensi agar tidak memengaruhi dependensi yang lain.

Penggabungan longgar akan memungkinkan Anda untuk mengubah kode atau menambahkan fitur ke sebuah komponen sekaligus meminimalkan risiko pada komponen lain yang bergantung pada komponen tersebut. Hal ini juga memungkinkan ketahanan hingga tingkatan terkecil (granular) pada tingkat komponen sehingga Anda dapat menambahkan skala (scale-out) atau bahkan mengubah implementasi yang mendasari dependensi.

Agar makin meningkatkan ketahanan melalui penggabungan longgar, buatlah interaksi-interaksi komponen tak selaras, jika memungkinkan. Model ini cocok untuk interaksi apa pun yang tidak memerlukan respons langsung dan di mana pengakuan bahwa permintaan telah terdaftar sudah dianggap cukup. Ini melibatkan satu komponen yang menghasilkan peristiwa dan komponen-komponen lain yang menggunakannya. Kedua komponen tidak terintegrasi melalui point-to-point interaksi langsung tetapi biasanya melalui lapisan penyimpanan tahan lama menengah, seperti SQS antrian Amazon, platform data streaming seperti Amazon Kinesis, atau. AWS Step Functions

Gambar 4: Dependensi seperti sistem pengantrean dan penyeimbang beban digabungkan dengan longgar

Amazon SQS mengantri dan hanya AWS Step Functions dua cara untuk menambahkan lapisan perantara untuk kopling longgar. Arsitektur berbasis peristiwa juga dapat dibangun menggunakan AWS Cloud Amazon EventBridge, yang dapat mengabstraksi klien (produsen acara) dari layanan

yang mereka andalkan (konsumen acara). Amazon Simple Notification Service (AmazonSNS) adalah solusi efektif ketika Anda membutuhkan throughput tinggi, berbasis push, perpesanan. many-to-many Menggunakan SNS topik Amazon, sistem penerbit Anda dapat menyebarkan pesan ke sejumlah besar titik akhir pelanggan untuk pemrosesan paralel.

Meskipun antrean menawarkan sejumlah manfaat, di sebagian besar sistem waktu nyata yang keras, permintaan yang lebih lama dari waktu ambang batas (sering kali dalam hitungan detik) harus dianggap basi (klien telah menyerah dan sudah tidak menunggu respons lagi), dan tidak diproses. Dengan begitu, permintaan yang lebih baru (dan kemungkinan masih valid) dapat diproses sebagai gantinya.

Hasil yang diinginkan: Menerapkan dependensi yang digabungkan dengan longgar akan memungkinkan Anda untuk meminimalkan peluang kegagalan ke tingkat komponen, dan akan membantu Anda untuk mendiagnosis dan menyelesaikan masalah. Cara ini juga dapat menyederhanakan siklus pengembangan, sehingga memungkinkan tim untuk menerapkan perubahan-perubahan pada tingkat modular tanpa memengaruhi performa komponen-komponen lain yang bergantung padanya. Pendekatan ini memberikan kemampuan untuk menambahkan skala (scale-out) pada tingkat komponen berdasarkan kebutuhan sumber daya, serta pemanfaatan komponen yang berkontribusi terhadap efektivitas biaya.

Anti-pola umum:

- Melakukan deployment beban kerja monolitik.
- Langsung memanggil APIs antara tingkatan beban kerja tanpa kemampuan failover atau pemrosesan permintaan asinkron.
- Penggabungan erat dengan menggunakan data bersama. Sistem-sistem yang digabungkan dengan longgar sebaiknya tidak berbagi data melalui basis data bersama atau bentuk penyimpanan data yang digabungkan secara erat, yang dapat menimbulkan kembali penggabungan erat dan akan menghambat skalabilitas.
- Mengabaikan tekanan balik. Beban kerja Anda harus memiliki kemampuan untuk memperlambat atau menghentikan data yang masuk ketika sebuah komponen tidak dapat memprosesnya dengan kecepatan yang sama.

Manfaat menerapkan praktik terbaik ini: Penggabungan longgar akan membantu Anda untuk memisahkan perilaku suatu komponen dari komponen lainnya yang bergantung pada komponen tersebut, sehingga akan meningkatkan ketahanan dan ketangkasan. Kegagalan di salah satu komponen dipisahkan dari komponen lain.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Implementasikan dependensi yang digabungkan dengan longgar. Ada berbagai solusi yang memungkinkan Anda untuk membangun aplikasi yang digabungkan dengan longgar. Ini termasuk layanan untuk menerapkan antrian yang dikelola sepenuhnya, alur kerja otomatis, bereaksi terhadap peristiwa, dan APIs antara lain yang dapat membantu mengisolasi perilaku komponen dari komponen lain, dan dengan demikian meningkatkan ketahanan dan kelincahan.

- Bangun arsitektur berbasis peristiwa: Amazon EventBridge membantu Anda membangun arsitektur berbasis peristiwa yang digabungkan dan didistribusikan secara longgar.
- Menerapkan antrian dalam sistem terdistribusi: Anda dapat menggunakan Amazon Simple Queue Service (AmazonSQS) untuk mengintegrasikan dan memisahkan sistem terdistribusi.
- Kontainerisasi komponen sebagai layanan mikro: Layanan mikro memungkinkan tim untuk membangun aplikasi yang terdiri dari komponen independen kecil yang berkomunikasi melalui definisi yang terdefinisi dengan baik. APIs Amazon Elastic Container Service (AmazonECS), dan Amazon Elastic Kubernetes Service (EKSAmazon) dapat membantu Anda memulai lebih cepat dengan kontainer.
- Kelola alur kerja dengan Step Functions: Step Functions membantu Anda mengoordinasikan beberapa AWS layanan ke dalam alur kerja yang fleksibel.
- Manfaatkan arsitektur perpesanan berlangganan publikasi (pub/sub): Amazon Simple Notification Service (AmazonSNS) menyediakan pengiriman pesan dari penerbit ke pelanggan (juga dikenal sebagai produsen dan konsumen).

Langkah-langkah implementasi

- Komponen dalam sebuah arsitektur yang didorong peristiwa dimulai oleh peristiwa. Peristiwa adalah tindakan-tindakan yang terjadi dalam sebuah sistem, seperti pengguna menambahkan item ke keranjang. Ketika suatu tindakan berhasil, sebuah peristiwa dihasilkan, yang akan menggerakkan komponen berikutnya dalam sistem tersebut.
  - Membangun Aplikasi Event Driven dengan Amazon EventBridge
  - AWS re:invent 2022 - Merancang Integrasi Berbasis Acara menggunakan Amazon EventBridge
- Sistem olah pesan terdistribusi memiliki tiga bagian utama yang perlu diimplementasikan untuk sebuah arsitektur berbasis antrean. Mereka termasuk komponen sistem terdistribusi, antrian yang digunakan untuk decoupling (didistribusikan di SQS server Amazon), dan pesan dalam

antrian. Sistem seperti ini biasanya memiliki produsen yang memulai pesan ke dalam antrean, dan konsumen yang menerima pesan dari antrean tersebut. Antrian menyimpan pesan di beberapa SQS server Amazon untuk redundansi.

  - SQSArsitektur Amazon dasar
  - Kirim Pesan Antara Aplikasi Terdistribusi dengan Amazon Simple Queue Service
- Layanan mikro, jika dimanfaatkan dengan baik, akan meningkatkan pemeliharaan dan mendongkrak skalabilitas, karena komponen-komponen yang digabungkan dengan longgar dikelola oleh tim independen. Hal ini juga akan memungkinkan isolasi perilaku ke satu komponen jika terjadi perubahan.
  - Menerapkan Layanan Mikro pada AWS
  - Mari Merancang! Merancang arsitektur layanan mikro dengan kontainer
- Dengan AWS Step Functions Anda dapat membangun aplikasi terdistribusi, mengotomatiskan proses, mengatur layanan mikro, antara lain. Melakukan orkestrasi beberapa komponen ke dalam sebuah alur kerja otomatis akan memungkinkan Anda untuk memisahkan dependensi dalam aplikasi Anda.
  - Buat Alur Kerja Tanpa Server dengan dan AWS Step FunctionsAWS Lambda
  - Memulai dengan AWS Step Functions

Sumber daya

Dokumen terkait:

- AmazonEC2: Memastikan Idempotensi
- Amazon Builders' Library: Tantangan dengan sistem terdistribusi
- Amazon Builders' Library: Keandalan, kerja konstan, dan pilihan yang tepat
- Apa itu Amazon EventBridge?
- Apa Itu Amazon Simple Queue Service?
- Putus dengan monolit Anda
- Mengatur Layanan Mikro Berbasis Antrian dengan dan Amazon AWS Step Functions SQS
- SQSArsitektur Amazon dasar
- Arsitektur Berbasis Antrian

Video terkait:

- AWS KTT New York 2019: Pengantar Arsitektur Berbasis Acara dan Amazon EventBridge (05) MAD2
- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil ARC337 (termasuk kopling longgar, kerja konstan, stabilitas statis)
- AWS re:invent 2019: Pindah ke arsitektur berbasis acara (08) SVS3
- AWS re:invent 2019: Aplikasi berbasis peristiwa tanpa server yang dapat diskalakan menggunakan Amazon dan Lambda SQS
- AWS re:invent 2022 - Merancang integrasi berbasis acara menggunakan Amazon EventBridge
- AWS RE: Invent 2017: Elastic Load Balancing Deep Dive dan Praktik Terbaik

REL04-BP03 Lakukan pekerjaan konstan

Sistem dapat gagal mengalami kegagalan saat ada perubahan besar dan cepat pada beban. Misalnya, jika beban kerja Anda sedang melakukan pemeriksaan kondisi yang memantau kondisi dari ribuan server, beban kerja Anda harus mengirimkan payload berukuran sama (snapshot penuh berisi status saat ini) setiap saat. Saat tidak ada server yang gagal, atau semuanya gagal, sistem pemeriksaan kondisi melakukan tugas konstan tanpa perubahan besar dan cepat.

Misalnya, jika sistem pemeriksaan kondisi sedang memantau 100.000 server, dengan tingkat kegagalan server normal yang ringan, maka beban yang ditanggung kecil. Namun demikian, jika ada sebuah peristiwa besar yang membuat separuh server menjadi tidak sehat, maka sistem pemeriksaan kondisi akan kewalahan untuk memperbarui sistem notifikasi dan menyampaikan status ke kliennya. Jadi alih-alih sistem pemeriksaan kondisi harus mengirim snapshot lengkap dari keadaan saat ini setiap kali. 100.000 status kesehatan server, masing-masing diwakili oleh satu bit, dan itu hanya akan menjadi muatan sebesar 12,5 KB. Saat tidak ada server yang gagal, atau semuanya gagal, sistem pemeriksaan kondisi akan melakukan tugas konstan, dan perubahan yang besar dan cepat bukanlah ancaman untuk stabilitas sistem. Seperti inilah Amazon Route 53 menangani pemeriksaan kondisi untuk titik akhir (seperti alamat IP) untuk menentukan bagaimana pengguna akhir dirutekan ke sana.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

- Lakukan tugas konstan sehingga sistem tidak gagal saat terdapat perubahan beban yang besar dan cepat.

- Implementasikan dependensi yang digabungkan dengan longgar. Dependensi seperti sistem pengantrean, sistem streaming, alur kerja, dan penyeimbang beban digabungkan secara longgar. Penggabungan longgar membantu memisahkan perilaku suatu komponen dari komponen lainnya yang bergantung pada komponen tersebut, sehingga meningkatkan ketahanan dan ketangkasan.
  - Amazon Builders' Library: Keandalan, kerja konstan, dan pilihan yang tepat
  - AWS Re: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil ARC337 (termasuk pekerjaan konstan)
    - Untuk contoh sistem pemeriksaan kondisi yang memantau 100.000 server, lakukan rekayasa beban kerja sehingga ukuran payload tetap sama berapa pun jumlah keberhasilan atau kegagalan yang terjadi.

Sumber daya

Dokumen terkait:

- AmazonEC2: Memastikan Idempotensi
- Amazon Builders' Library: Tantangan dengan sistem terdistribusi
- Amazon Builders' Library: Keandalan, kerja konstan, dan pilihan yang tepat

Video terkait:

- AWS KTT New York 2019: Pengantar Arsitektur yang Digerakkan oleh Acara dan Amazon EventBridge (05) MAD2
- AWS Re: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil ARC337 (termasuk pekerjaan konstan)
- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil ARC337 (termasuk kopling longgar, kerja konstan, stabilitas statis)
- AWS re:invent 2019: Pindah ke arsitektur berbasis acara (08) SVS3

REL04-BP04 Jadikan semua tanggapan idempoten

Layanan idempoten menjanjikan setiap permintaan diselesaikan tepat satu kali, sehingga pembuatan beberapa permintaan yang sama memiliki efek yang sama seperti membuat satu permintaan. Layanan idempoten memudahkan klien untuk mengimplementasikan percobaan ulang tanpa takut permintaan akan salah diproses beberapa kali. Untuk melakukan ini, klien dapat mengeluarkan API

permintaan dengan token idempotensi—token yang sama digunakan setiap kali permintaan diulang. Layanan idempoten API menggunakan token untuk mengembalikan respons yang identik dengan respons yang dikembalikan saat pertama kali permintaan selesai.

Dalam sebuah sistem terdistribusi, mudah untuk melakukan tindakan paling banyak satu kali (klien hanya membuat satu permintaan), atau setidaknya satu kali (tetap mengirimkan permintaan sampai klien mendapat konfirmasi berhasil). Tetapi sulit untuk menjamin suatu tindakan idempoten, yang berarti itu dilakukan benar-benar sekali, sehingga membuat beberapa permintaan yang identik memiliki efek yang sama seperti membuat satu permintaan. Menggunakan token idempotensi diAPIs, layanan dapat menerima permintaan mutasi satu kali atau lebih tanpa membuat catatan duplikat atau efek samping.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Membuat semua respons menjadi idempoten. Layanan idempoten menjanjikan setiap permintaan diselesaikan tepat satu kali, sehingga pembuatan beberapa permintaan yang sama memiliki efek yang sama seperti membuat satu permintaan.
  - Klien dapat mengeluarkan API permintaan dengan token idempotensi—token yang sama digunakan setiap kali permintaan diulang. Layanan idempoten API menggunakan token untuk mengembalikan respons yang identik dengan respons yang dikembalikan saat pertama kali permintaan selesai.
    - AmazonEC2: Memastikan Idempotensi

Sumber daya

Dokumen terkait:

- AmazonEC2: Memastikan Idempotensi
- Amazon Builders' Library: Tantangan dengan sistem terdistribusi
- Amazon Builders' Library: Keandalan, kerja konstan, dan pilihan yang tepat

Video terkait:

- AWS KTT New York 2019: Pengantar Arsitektur Berbasis Acara dan Amazon EventBridge (05) MAD2

- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil ARC337 (termasuk kopling longgar, kerja konstan, stabilitas statis)
- AWS re:invent 2019: Pindah ke arsitektur berbasis acara (08) SVS3

## REL5. Bagaimana cara mendesain interaksi di sistem terdistribusi untuk memitigasi atau bertahan dari kegagalan?

Sistem terdistribusi mengandalkan jaringan komunikasi untuk membuat interkoneksi komponen (seperti server atau layanan). Beban kerja Anda harus beroperasi secara andal terlepas latensi atau hilangnya data pada jaringan-jaringan ini. Komponen dari sistem terdistribusi harus beroperasi dengan cara yang tidak secara negatif memengaruhi beban kerja atau komponen-komponen lain. Berbagai praktik terbaik ini memungkinkan beban kerja bertahan dari stres atau kegagalan, lebih cepat pulih darinya, dan memitigasi dampak gangguan tersebut. Hasilnya ditingkatkan waktu rata-rata untuk pemulihan (MTTR).

Praktik terbaik

- REL05-BP01 Menerapkan degradasi anggun untuk mengubah dependensi keras yang berlaku menjadi dependensi lunak
- REL05-BP02 Permintaan Throttle
- REL05-BP03 Kontrol dan batasi panggilan coba lagi
- REL05-BP04 Gagal cepat dan batasi antrian
- REL05-BP05 Mengatur batas waktu klien
- REL05-BP06 Membuat sistem tanpa kewarganegaraan jika memungkinkan
- REL05-BP07 Menerapkan tuas darurat

### REL05-BP01 Menerapkan degradasi anggun untuk mengubah dependensi keras yang berlaku menjadi dependensi lunak

Komponen aplikasi harus terus menjalankan fungsi intinya bahkan jika dependensi menjadi tidak tersedia. Komponen mungkin menyajikan data yang sedikit basi, data alternatif, atau bahkan tidak menyajikan data sama sekali. Hal ini memastikan fungsi sistem secara keseluruhan hanya terhambat secara minimum oleh kegagalan lokal sekaligus memberikan nilai bisnis utama.

Hasil yang diinginkan: Saat dependensi sebuah komponen tidak optimum, komponen tersebut masih dapat berfungsi, meskipun terbatas atau terdegradasi. Mode-mode kegagalan komponen harus dipandang sebagai operasi normal. Alur kerja harus dirancang dengan desain sedemikian

rupa sehingga kegagalan tersebut tidak menyebabkan kegagalan total atau setidaknya hanya menyebabkan keadaan yang dapat diprediksi dan dapat dipulihkan.

Anti-pola umum:

- Tidak mengidentifikasi fungsi bisnis inti yang dibutuhkan. Tidak menguji bahwa komponen berfungsi bahkan selama kegagalan dependensi.
- Tidak menyajikan data jika terjadi kesalahan atau ketika hanya ada satu dari beberapa dependensi yang tidak tersedia dan hasil sebagian masih dapat dikembalikan.
- Menciptakan sebuah keadaan yang tidak konsisten ketika transaksi mengalami gagal sebagian.
- Tidak memiliki cara alternatif untuk mengakses tempat penyimpanan parameter pusat.
- Membatalkan atau mengosongkan status lokal sebagai akibat dari penyegaran yang gagal tanpa mempertimbangkan konsekuensi yang ditimbulkan oleh tindakan tersebut.

Manfaat menerapkan praktik terbaik ini: Degradasi bertahap (graceful degradation) akan meningkatkan ketersediaan sistem secara keseluruhan dan mempertahankan fungsionalitas dari fungsi-fungsi yang paling penting, bahkan selama terjadi kegagalan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Menerapkan degradasi yang tepat akan membantu Anda meminimalkan dampak kegagalan dependensi yang terjadi pada fungsi komponen. Idealnya, sebuah komponen mendeteksi kegagalan-kegagalan dependensi dan menanganinya dengan cara yang berdampak minim pada pelanggan atau komponen lain.

Merancang arsitektur untuk degradasi yang tepat berarti mempertimbangkan potensi mode kegagalan selama desain dependensi. Untuk setiap mode kegagalan, miliki cara untuk menghadirkan sebagian besar atau setidaknya fungsionalitas yang paling penting dari komponen kepada pemanggil atau pelanggan. Pertimbangan-pertimbangan ini dapat menjadi persyaratan tambahan yang dapat diuji dan diverifikasi. Idealnya, sebuah komponen harus mampu menjalankan fungsi intinya dengan cara yang dapat diterima bahkan ketika satu atau beberapa dependensi gagal.

Ini bukan hanya pembahasan teknis, melainkan juga pembahasan bisnis. Semua persyaratan bisnis penting dan harus dipenuhi, jika memungkinkan. Namun demikian, menanyakan apa yang seharusnya terjadi ketika tidak semua persyaratan tersebut dapat dipenuhi adalah hal yang wajar. Suatu sistem dapat dirancang agar tersedia dan konsisten, tetapi dalam keadaan yang

mengharuskan salah satu persyaratan untuk dikorbankan, mana yang lebih penting? Untuk pemrosesan pembayaran, jawabannya mungkin adalah konsistensi. Untuk aplikasi waktu nyata, jawabannya mungkin adalah ketersediaan. Untuk sebuah situs web yang digunakan langsung oleh pelanggan, jawabannya mungkin tergantung pada ekspektasi pelanggan.

Seberapa pentingnya, ini tergantung persyaratan komponen dan apa yang seharusnya dianggap sebagai fungsi intinya. Sebagai contoh:

- Situs web ecommerce mungkin akan menampilkan data dari berbagai sistem, misalnya rekomendasi yang dipersonalisasi, produk dengan peringkat tertinggi, dan status pesanan pelanggan di halaman arahan. Ketika salah satu sistem hulu gagal, masih masuk akal untuk menampilkan semua daripada menampilkan halaman kesalahan kepada pelanggan.
- Sebuah komponen yang menjalankan penulisan batch masih dapat melanjutkan pemrosesan batch jika salah satu operasi mengalami kegagalan. Implementasi mekanisme percobaan ulang harus sederhana. Hal ini dapat dilakukan dengan mengembalikan informasi tentang operasi-operasi yang berhasil, yang telah gagal, dan mengapa operasi-operasi tersebut gagal ke pemanggil, atau dengan menempatkan permintaan yang gagal ke dalam antrean surat mati untuk mengimplementasikan percobaan ulang tidak selaras. Informasi tentang operasi-operasi yang gagal juga harus dibuatkan log.
- Sebuah sistem yang memproses transaksi harus memastikan bahwa semua pembaruan individual dijalankan atau tidak sama sekali. Untuk transaksi-transaksi terdistribusi, pola saga dapat digunakan untuk kembali ke operasi sebelumnya jika operasi selanjutnya dari transaksi yang sama mengalami kegagalan. Di sini, fungsi intinya adalah menjaga konsistensi.
- Sistem-sistem time-critical harus mampu menangani dependensi yang tidak memberikan respons secara tepat waktu. Dalam kasus-kasus ini, pola pemutus sirkuit dapat digunakan. Ketika respons dari sebuah dependensi mulai mencapai batas waktu, sistem dapat beralih ke keadaan ditutup di mana tidak ada panggilan tambahan yang dibuat.
- Sebuah aplikasi dapat membaca parameter dari tempat penyimpanan parameter. Membuat citra kontainer dengan serangkaian parameter default akan membantu agar apabila tempat penyimpanan parameter tidak tersedia citra tersebut dapat digunakan.

Perlu diperhatikan bahwa jalur-jalur yang diambil jika terjadi kegagalan komponen perlu diuji dan harus jauh lebih sederhana daripada jalur-jalur utama. Umumnya, strategi fallback harus dihindari.

Langkah-langkah implementasi

Identifikasi dependensi eksternal dan internal. Pertimbangkan jenis-jenis kegagalan yang bisa terjadi di dalamnya. Pikirkan tentang cara-cara yang dapat meminimalkan dampak negatif terhadap pelanggan serta sistem hulu dan hilir selama kegagalan-kegagalan tersebut.

Berikut ini adalah daftar dependensi dan cara melakukan degradasi yang tepat ketika dependensi mengalami kegagalan:

1. Kegagalan dependensi parsial: Sebuah komponen dapat melakukan beberapa permintaan ke sistem-sistem hilir, baik beberapa permintaan ke satu sistem atau satu permintaan ke beberapa sistem. Tergantung konteks bisnis, mungkin ada berbagai cara penanganan yang sesuai (untuk detail lebih lanjut, silakan lihat contoh-contoh sebelumnya dalam Panduan implementasi).
2. Sistem hilir tidak dapat memproses permintaan karena beban tinggi: Jika permintaan ke sistem hilir terus-menerus gagal, sebaiknya Anda tidak mencoba lagi. Tindakan ini dapat menciptakan beban tambahan pada sistem yang sudah mengalami kelebihan beban dan mempersulit pemulihan. Pola pemutus sirkuit dapat digunakan di sini, yang memantau kegagalan panggilan ke sistem hilir. Jika ada banyak panggilan yang mengalami kegagalan, permintaan akan berhenti dikirimkan ke sistem hilir dan hanya sesekali panggilan dibiarkan masuk untuk menguji apakah sistem hilir sudah tersedia kembali.
3. Gudang parameter tidak tersedia: Untuk mengubah tempat penyimpanan parameter, caching dependensi lunak atau sane default yang disertakan di dalam image kontainer atau mesin dapat digunakan. Perhatikan bahwa default ini perlu disimpan up-to-date dan disertakan dalam rangkaian pengujian.
4. Layanan pemantauan atau dependensi non-fungsional lainnya tidak tersedia: Jika sebuah komponen sebentar-sebentar tidak dapat mengirim log, metrik, atau jejak ke layanan pemantauan pusat, langkah terbaiknya sering kali adalah tetap menjalankan fungsi-fungsi bisnis seperti biasa. Diam-diam tidak membuat log atau mendorong metrik dalam waktu yang lama sering kali tidak dapat diterima. Selain itu, beberapa kasus penggunaan mungkin akan memerlukan entri audit lengkap untuk memenuhi persyaratan-persyaratan kepatuhan.
5. Instans primer basis data relasional mungkin tidak tersedia: Amazon Relational Database Service, seperti hampir semua database relasional, hanya dapat memiliki satu contoh penulis utama. Hal ini akan menciptakan satu titik kegagalan untuk beban kerja tulis dan menjadikan penskalaan menjadi lebih sulit. Hal ini dapat diatasi sebagiannya dengan menggunakan konfigurasi Multi-AZ untuk mendapatkan ketersediaan tinggi atau Amazon Aurora Nirserver untuk mendapatkan penskalaan yang lebih baik. Untuk persyaratan-persyaratan ketersediaan yang sangat tinggi, ada baiknya untuk tidak bergantung pada penulis utama sama sekali. Untuk kueri yang hanya

membaca, replika baca dapat digunakan, yang memberikan redundansi dan kemampuan untuk melakukan penambahan skala (scale-out), bukan hanya scale-up. Tulis dapat di-buffer, misalnya dalam antrean Amazon Simple Queue Service, sehingga permintaan tulis dari pelanggan masih dapat diterima bahkan jika penulis utama tidak tersedia untuk sementara.

Sumber daya

Dokumen terkait:

- Amazon API Gateway: API Permintaan Throttle untuk Throughput yang Lebih Baik
- CircuitBreaker(merangkum Circuit Breaker dari “Release It!” buku)
- Error Retries dan Exponential Backoff di AWS
- Michael Nygard “Release It! Rancang dan Lakukan Deployment Perangkat Lunak yang Siap Diproduksi”
- Amazon Builders' Library: Menghindari fallback dalam sistem terdistribusi
- Amazon Builders' Library: Menghindari backlog antrean yang tidak dapat diatasi
- Amazon Builders' Library: Tantangan dan strategi caching
- Amazon Builders' Library: Batas waktu, percobaan ulang, dan penundaan dengan jitter

Video terkait:

- Coba lagi, backoff, dan jitter: AWS re:Invent 2019: Memperkenalkan Perpustakaan Pembangun Amazon () DOP328

Contoh terkait:

- Lab Well-Architected: Level 300: Mengimplementasikan Pemeriksaan Kondisi dan Mengelola Dependensi untuk Meningkatkan Keandalan

REL05-BP02 Permintaan Throttle

Batasi permintaan untuk mengurangi keletihan sumber daya karena peningkatan permintaan yang tidak terduga. Permintaan di bawah laju pembatasan diproses sementara permintaan di atas batas yang ditentukan ditolak dengan pesan balasan yang menunjukkan permintaan telah dibatasi.

Hasil yang diinginkan: Lonjakan volume dalam jumlah besar baik dari peningkatan lalu lintas pelanggan yang naik tiba-tiba, serangan membanjir, atau banjir percobaan ulang akan diminimalkan dengan throttling permintaan, sehingga beban kerja dapat melanjutkan pemrosesan volume permintaan normal yang didukung.

Anti-pola umum:

- APIthrottle endpoint tidak diimplementasikan atau dibiarkan pada nilai default tanpa mempertimbangkan volume yang diharapkan.
- APItitik akhir tidak diuji beban atau batas pelambatan tidak diuji.
- Lakukan throttling terhadap angka permintaan tanpa mempertimbangkan ukuran atau kompleksitas permintaan.
- Melakukan uji laju permintaan maksimum atau uji ukuran permintaan maksimum, tetapi tidak menguji keduanya bersama-sama.
- Sumber daya tidak disediakan untuk batas yang sama yang ditetapkan dalam pengujian.
- Rencana penggunaan belum dikonfigurasi atau dipertimbangkan untuk konsumen aplikasi ke aplikasi (A2A)API.
- Tidak ada konfigurasi pengaturan konkurensi maksimum pada konsumen antrean yang mengalami penskalaan horizontal.
- Pembatasan tingkat untuk setiap alamat IP belum diimplementasikan.

Manfaat menerapkan praktik terbaik ini: Beban kerja yang menetapkan batas throttling dapat beroperasi secara normal dan berhasil memproses beban permintaan yang diterima saat terjadi lonjakan volume yang tidak terduga. Lonjakan permintaan dan antrian yang tiba-tiba atau berkelanjutan dibatasi APIs dan tidak menghabiskan sumber daya pemrosesan permintaan. Batas tarif membatasi pemohon individu sehingga volume lalu lintas yang tinggi dari satu alamat IP atau API konsumen tidak akan menghabiskan sumber daya berdampak pada konsumen lain.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Layanan-layanan harus dirancang untuk memproses kapasitas permintaan yang diketahui; kapasitas ini dapat ditetapkan melalui pengujian beban. Jika laju kedatangan permintaan sudah melampaui batas, maka respons yang tepat menandakan bahwa permintaan telah mengalami throttling. Hal ini akan memungkinkan konsumen untuk menangani kesalahan dan mencoba ulang di lain waktu.

Saat layanan-layanan Anda memerlukan implementasi throttling, pertimbangkan untuk mengimplementasikan algoritme bucket token, yang menghitung satu token sebagai satu permintaan. Token diisi ulang dengan laju throttling per detik dan dikosongkan secara tidak selaras dengan satu token per permintaan.


Algoritme bucket token.

Amazon API Gateway mengimplementasikan algoritma token bucket sesuai dengan batas akun dan wilayah dan dapat dikonfigurasi per klien dengan paket penggunaan. Selain itu, Amazon Simple Queue Service (AmazonSQS) dan Amazon Kinesis dapat menyangga permintaan untuk memperlancar tingkat permintaan, dan memungkinkan tingkat pembatasan yang lebih tinggi untuk permintaan yang dapat ditangani. Terakhir, Anda dapat menerapkan pembatasan laju dengan AWS WAFmembatasi API konsumen tertentu yang menghasilkan beban yang luar biasa tinggi.

Langkah-langkah implementasi

Anda dapat mengonfigurasi API Gateway dengan batas pelambatan untuk `429 Too Many Requests` kesalahan Anda APIs dan mengembalikan saat batas terlampaui. Anda dapat menggunakan AWS WAF dengan titik akhir AWS AppSync dan API Gateway Anda untuk mengaktifkan pembatasan tarif berdasarkan per alamat IP. Selain itu, apabila sistem Anda dapat memberikan toleransi terhadap pemrosesan tidak selaras, Anda dapat memasukkan pesan ke dalam antrean atau aliran guna mempercepat respons terhadap klien layanan, yang memungkinkan Anda untuk melakukan lonjakan ke tingkat throttling yang lebih tinggi.

Dengan pemrosesan asinkron, ketika Anda telah mengonfigurasi Amazon SQS sebagai sumber peristiwa AWS Lambda, Anda dapat mengonfigurasi konkurensi maksimum untuk menghindari

tingkat kejadian tinggi dari penggunaan kuota eksekusi bersamaan akun yang tersedia yang diperlukan untuk layanan lain di beban kerja atau akun Anda.

Meskipun API Gateway menyediakan implementasi token bucket yang terkelola, jika Anda tidak dapat menggunakan API Gateway, Anda dapat memanfaatkan implementasi sumber terbuka khusus bahasa (lihat contoh terkait di Sumber Daya) dari keranjang token untuk layanan Anda.

- Memahami dan mengonfigurasi batas pembatasan API Gateway di tingkat akun per wilayah, API per tahap, dan API kunci per level paket penggunaan.
- Terapkan aturan pembatasan AWS WAF tarif ke API Gateway dan AWS AppSync titik akhir untuk melindungi dari banjir dan memblokir bahaya. IPs Aturan pembatasan tarif juga dapat dikonfigurasi pada AWS AppSync API kunci untuk konsumen A2A.
- Pertimbangkan apakah Anda memerlukan lebih banyak kontrol pelambatan daripada pembatasan laju untuk AWS AppSync APIs, dan jika demikian, konfigurasikan API Gateway di depan titik akhir Anda AWS AppSync .
- Saat SQS antrian Amazon disiapkan sebagai pemicu bagi konsumen antrian Lambda, tetapkan konkurensi maksimum ke nilai yang cukup diproses untuk memenuhi tujuan tingkat layanan Anda tetapi tidak menggunakan batas konkurensi yang memengaruhi fungsi Lambda lainnya. Pertimbangkan untuk menetapkan konkurensi cadangan pada fungsi Lambda lain di akun dan wilayah yang sama saat Anda menggunakan antrean dengan Lambda.
- Gunakan API Gateway dengan integrasi layanan asli ke Amazon SQS atau Kinesis untuk menyangga permintaan.
- Jika Anda tidak dapat menggunakan API Gateway, lihat pustaka khusus bahasa untuk mengimplementasikan algoritma token bucket untuk beban kerja Anda. Periksa bagian contoh dan lakukan riset sendiri untuk menemukan pustaka yang sesuai.
- Uji batas yang ingin Anda tetapkan, atau yang ingin Anda izinkan untuk ditingkatkan, dan buatlah dokumentasi dari batas-batas yang diuji.
- Jangan tingkatkan batas melebihi apa yang Anda tetapkan dalam pengujian. Saat meningkatkan batas, pastikan bahwa sumber daya yang disediakan sudah setara atau lebih besar daripada yang ada dalam skenario pengujian sebelum menerapkan peningkatan.

Sumber daya

Praktik-praktik terbaik terkait:

- REL04-BP03 Lakukan pekerjaan konstan

- REL05-BP03 Kontrol dan batasi panggilan coba lagi

Dokumen terkait:

- Amazon API Gateway: API Permintaan Throttle untuk Throughput yang Lebih Baik
- AWS WAF: Pernyataan aturan berbasis laju
- Memperkenalkan konkurensi maksimum AWS Lambda saat menggunakan Amazon SQS sebagai sumber acara
- AWS Lambda: Konkurensi Maksimum

Contoh terkait:

- Tiga aturan AWS WAF berbasis tarif yang paling penting
- Java Bucket4j
- Bucket token Python
- Bucket token Node
- . NETPembatasan Tingkat Threading Sistem

Video terkait:

- Menerapkan praktik terbaik keamanan API GraphQL dengan AWS AppSync

Alat terkait:

- APIGerbang Amazon
- AWS AppSync
- Amazon SQS
- Amazon Kinesis
- AWS WAF

### REL05-BP03 Kontrol dan batasi panggilan coba lagi

Gunakan penundaan eksponensial untuk mencoba ulang permintaan dengan interval yang makin lama antara setiap percobaan ulang. Terapkan jitter antara percobaan ulang untuk mengacak interval percobaan ulang. Batasi jumlah percobaan ulang maksimum.

Hasil yang diinginkan: Komponen khas dalam sistem perangkat lunak terdistribusi termasuk server, penyeimbang beban, database, dan server. DNS Selama operasi normal, komponen-komponen ini dapat merespons permintaan yang memiliki kesalahan yang bersifat sementara atau terbatas, dan juga kesalahan yang persisten terlepas dari percobaan ulang. Ketika klien membuat permintaan ke layanan, permintaan tersebut mengonsumsi sumber daya termasuk memori, thread, koneksi, port, atau sumber daya terbatas lainnya. Mengontrol dan membatasi percobaan ulang adalah strategi untuk melepaskan dan meminimalkan konsumsi sumber daya sehingga komponen sistem yang ada di bawah tekanan tidak kewalahan.

Ketika permintaan klien mengalami batas waktu atau menerima respons kesalahan, mereka harus menentukan apakah akan mencoba lagi atau tidak. Jika mereka mencoba lagi, mereka melakukannya dengan penundaan eksponensial dengan jitter dan nilai coba ulang maksimum. Karena itu, layanan dan proses backend mendapat kelonggaran beban dan waktu untuk pulih secara mandiri, sehingga hal itu akan mengakibatkan terjadinya pemulihan yang lebih cepat dan pelayanan permintaan yang berhasil.

Anti-pola umum:

- Mengimplementasikan percobaan ulang tanpa menambahkan penundaan eksponensial, jitter, dan nilai coba ulang maksimum. Penundaan dan jitter membantu menghindari lonjakan lalu lintas semu yang disebabkan percobaan ulang yang dikoordinasikan secara tidak sengaja pada interval umum.
- Menerapkan percobaan ulang tanpa menguji efeknya atau dengan asumsi percobaan ulang sudah dibangun ke dalam skenario coba lagi SDK tanpa pengujian.
- Tidak memahami kode kesalahan yang dipublikasikan dari dependensi, yang menyebabkan percobaan ulang semua kesalahan, termasuk kesalahan dengan penyebab yang dengan jelas menunjukkan tidak adanya izin, kesalahan konfigurasi, atau kondisi-kondisi lain yang jelas tidak akan terselesaikan tanpa intervensi manual.
- Tidak menangani praktik-praktik observabilitas, termasuk pemantauan dan peringatan tentang kegagalan layanan berulang sehingga masalah yang mendasari dapat diketahui dan diatasi.
- Mengembangkan mekanisme-mekanisme percobaan ulang kustom saat kemampuan coba ulang bawaan atau pihak ketiga sudah mencukupi.

- Melakukan percobaan ulang pada beberapa lapisan tumpukan aplikasi dengan cara yang makin memperparah upaya-upaya percobaan ulang sehingga makin menyita sumber daya yang sedang berada dalam badai percobaan ulang. Pastikan bahwa Anda memahami bagaimana kesalahan-kesalahan ini memengaruhi aplikasi Anda dan dependensi yang Anda andalkan, dan kemudian terapkan percobaan ulang hanya pada satu tingkat.
- Melakukan percobaan ulang pada panggilan layanan yang tidak idempoten, sehingga menyebabkan efek samping yang tidak terduga seperti hasil-hasil ganda.

Manfaat menerapkan praktik terbaik ini: Percobaan ulang akan membantu klien memperoleh hasil yang diinginkan ketika permintaan mengalami kegagalan, tetapi juga dapat menyita lebih banyak waktu server untuk mendapatkan respons berhasil yang mereka inginkan. Ketika kegagalan jarang terjadi atau sementara, percobaan ulang dapat berfungsi dengan baik. Ketika kegagalan disebabkan oleh kelebihan beban sumber daya, melakukan percobaan ulang dapat memperburuk keadaan. Menambahkan penundaan eksponensial dengan jitter ke percobaan ulang klien memungkinkan server pulih ketika kegagalan disebabkan oleh kelebihan beban sumber daya. Jitter menghindarkan penyelarasan permintaan menjadi lonjakan, dan penundaan dapat mengurangi eskalasi beban yang disebabkan oleh penambahan percobaan ulang ke beban permintaan normal. Terakhir, penting bagi Anda untuk mengonfigurasi jumlah coba ulang maksimum atau waktu yang telah berlalu untuk menghindari terciptanya backlog yang dapat menghasilkan kegagalan yang bersifat metastabil.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Mengontrol dan membatasi panggilan percobaan ulang. Gunakan penundaan eksponensial untuk percobaan ulang setelah interval yang makin lama. Masukkan jitter untuk mengacak interval percobaan ulang dan batasi jumlah percobaan ulang maksimum.

Beberapa AWS SDKs mengimplementasikan percobaan ulang dan backoff eksponensial secara default. Gunakan AWS implementasi bawaan ini jika berlaku dalam beban kerja Anda. Implementasikan logika serupa dalam beban kerja Anda saat memanggil layanan yang bersifat idempoten dan apabila percobaan ulang meningkatkan ketersediaan klien Anda. Tentukan batas waktu dan kapan harus berhenti melakukan percobaan ulang berdasarkan kasus penggunaan Anda. Buat dan latih skenario pengujian untuk kasus-kasus penggunaan percobaan ulang tersebut.

Langkah-langkah implementasi

- Tentukan lapisan optimal dalam yang ada dalam tumpukan aplikasi Anda untuk mengimplementasikan percobaan ulang untuk layanan-layanan yang diandalkan aplikasi Anda.

- Sadarilah SDKs yang ada yang menerapkan strategi coba lagi yang telah terbukti dengan backoff eksponensial dan jitter untuk bahasa pilihan Anda, dan dukung ini daripada menulis implementasi coba ulang Anda sendiri.
- Verifikasi bahwa layanan sudah idempoten sebelum Anda menerapkan percobaan ulang. Setelah percobaan ulang diterapkan, pastikan bahwa keduanya diuji dan latihlah secara rutin dalam lingkungan produksi.
- Saat memanggil AWS layananAPIs, gunakan AWS SDKsdan AWS CLIdan pahami opsi konfigurasi coba lagi. Tentukan apakah konfigurasi default cocok untuk kasus penggunaan Anda, uji, dan lakukan penyesuaian sesuai kebutuhan.

Sumber daya

Praktik-praktik terbaik terkait:

- REL04-BP04 Jadikan semua tanggapan idempoten
- REL05-BP02 Permintaan Throttle
- REL05-BP04 Gagal cepat dan batasi antrian
- REL05-BP05 Mengatur batas waktu klien
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan

Dokumen terkait:

- Kesalahan Mencoba Ulang dan Backoff Eksponensial di AWS
- Amazon Builders' Library: Batas waktu, percobaan ulang, dan penundaan dengan jitter
- Penundaan Eksponensial dan Jitter
- Membuat percobaan ulang aman dengan idempoten APIs

Contoh terkait:

- Spring Retry
- Resilience4j Retry

Video terkait:

- Coba lagi, backoff, dan jitter: AWS re:Invent 2019: Memperkenalkan Perpustakaan Pembangun Amazon () DOP328

Alat terkait:

- AWS SDKsdan Alat: Coba lagi perilaku
- AWS Command Line Interface: mencoba AWS CLI lagi

REL05-BP04 Gagal cepat dan batasi antrian

Ketika layanan tidak berhasil merespons permintaan, lakukanlah gagal cepat (fail fast). Hal ini memungkinkan pelepasan sumber daya yang terkait dengan permintaan, dan mengizinkan layanan untuk melakukan pemulihan ketika layanan tersebut kehabisan sumber daya. Gagal cepat (fail fast) adalah pola desain perangkat lunak mapan yang dapat dimanfaatkan untuk membangun beban kerja yang sangat andal di cloud. Antrean juga merupakan pola integrasi korporat yang sudah mapan yang dapat memperlancar beban dan memungkinkan klien untuk melepaskan sumber daya ketika pemrosesan tak selaras dapat ditoleransi. Ketika layanan berhasil memberikan respons dalam kondisi normal tetapi gagal ketika laju permintaan terlalu tinggi, gunakan antrean untuk melakukan buffering permintaan. Namun demikian, jangan sampai ada penumpukan backlog antrean panjang yang dapat mengakibatkan diprosesnya permintaan-permintaan yang telah kedaluwarsa dan telah ditinggalkan oleh klien.

Hasil yang diinginkan: Ketika sistem berebut sumber daya, mengalami waktu habis, pengecualian, atau grey failure (kegagalan samar-samar) yang menyebabkan target tingkat layanan tidak dapat dicapai, strategi gagal cepat (fail fast) akan memungkinkan Anda melakukan pemulihan sistem yang lebih cepat. Sistem yang harus menyerap lonjakan lalu lintas dan dapat mengakomodasi pemrosesan tak selaras dapat meningkatkan keandalan dengan memungkinkan klien untuk secara cepat melepaskan permintaan dengan menggunakan antrean untuk melakukan buffering permintaan ke layanan backend. Ketika melakukan buffering permintaan ke antrean, strategi manajemen antrean diimplementasikan untuk menghindari backlog yang terlalu membebani.

Anti-pola umum:

- Menerapkan antrian pesan tetapi tidak mengonfigurasi antrian huruf mati (DLQ) atau alarm pada DLQ volume untuk mendeteksi ketika sistem gagal.
- Tidak mengukur usia pesan dalam antrean, yakni ukuran latensi untuk mengetahui kapan konsumen antrean tertinggal atau mengalami kesalahan yang menyebabkan percobaan ulang.

- Tidak menghapus pesan-pesan yang menumpuk dari antrean, padahal tidak ada gunanya memproses pesan-pesan tersebut jika kebutuhan bisnis sudah tidak lagi ada.
- Mengkonfigurasi antrian first in first out (FIFO) saat antrian last in first out (LIFO) akan lebih baik melayani kebutuhan klien, misalnya ketika pemesanan yang ketat tidak diperlukan dan pemrosesan backlog menunda semua permintaan baru dan sensitif waktu yang mengakibatkan semua klien mengalami tingkat layanan yang dilanggar.
- Mengekspos antrian internal ke klien alih-alih mengekspos APIs yang mengelola asupan kerja dan menempatkan permintaan ke dalam antrian internal.
- Menggabungkan terlalu banyak jenis permintaan kerja ke dalam satu antrean tunggal yang dapat memperburuk kondisi backlog dengan menyebarkan permintaan sumber daya di seluruh jenis permintaan.
- Memproses permintaan-permintaan yang kompleks dan sederhana dalam antrean yang sama, sehingga mengabaikan perbedaan kebutuhan pemantauan, batas waktu, dan alokasi sumber daya.
- Tidak memvalidasi input atau menggunakan pernyataan untuk mengimplementasikan mekanisme gagal cepat (fail fast) dalam perangkat lunak yang menaikkan pengecualian ke komponen dengan level lebih tinggi yang dapat menangani kesalahan secara mulus.
- Tidak menghapus sumber daya yang rusak dari perutean permintaan, terutama ketika terjadi kegagalan samar-samar yang menunjukkan keberhasilan sekaligus kegagalan akibat crash dan mulai ulang, kegagalan dependensi intermiten, kapasitas yang menurun, atau hilangnya paket jaringan.

Manfaat menerapkan praktik terbaik ini: Sistem yang gagal cepat (fail fast) lebih mudah untuk di-debug dan diperbaiki, dan sering kali dapat mengekspos masalah-masalah dalam pengodean dan konfigurasi sebelum rilis dipublikasikan ke tahap produksi. Sistem yang menggabungkan strategi antrean yang efektif memberikan ketahanan dan keandalan yang lebih baik terhadap lonjakan lalu lintas dan kondisi gangguan sistem intermiten.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Strategi gagal cepat (fail fast) dapat dikodekan ke dalam solusi perangkat lunak serta dikonfigurasi ke dalam infrastruktur. Selain gagal cepat (fail fast), antrean adalah teknik arsitektur yang sederhana namun ampuh untuk memisahkan komponen-komponen sistem dan memperlancar beban. Amazon CloudWatch menyediakan kemampuan untuk memantau dan alarm tentang kegagalan.

Setelah sistem diketahui mengalami kegagalan, strategi mitigasi dapat diinvokasi, termasuk gagal dan menjauh (fail away) dari sumber daya yang terdampak. Ketika sistem menerapkan antrian dengan Amazon SQS dan teknologi antrian lainnya untuk memperlancar beban, mereka harus mempertimbangkan cara mengelola backlog antrian, serta kegagalan konsumsi pesan.

Langkah-langkah implementasi

- Terapkan pernyataan terprogram atau metrik spesifik dalam perangkat lunak Anda dan gunakan itu untuk memperingatkan secara eksplisit tentang masalah sistem. Amazon CloudWatch membantu Anda membuat metrik dan alarm berdasarkan pola log aplikasi dan SDK instrumentasi.
- Gunakan CloudWatch metrik dan alarm untuk menghindari sumber daya yang terganggu yang menambahkan latensi ke pemrosesan atau berulang kali gagal memproses permintaan.
- Gunakan pemrosesan asinkron dengan merancang APIs untuk menerima permintaan dan menambahkan permintaan ke antrian internal menggunakan Amazon SQS dan kemudian menanggapi klien penghasil pesan dengan pesan sukses sehingga klien dapat melepaskan sumber daya dan melanjutkan pekerjaan lain saat konsumen antrian backend memproses permintaan.
- Ukur dan pantau latensi pemrosesan antrian dengan menghasilkan CloudWatch metrik setiap kali Anda mengeluarkan pesan dari antrian dengan membandingkan sekarang dengan stempel waktu pesan.
- Ketika ada kegagalan yang menghambat keberhasilan pemrosesan pesan atau terjadi lonjakan volume lalu lintas sehingga tidak dapat diproses dalam batas perjanjian tingkat layanan, sisihkan lalu lintas yang lebih lama atau berlebih ke antrean spillover. Hal ini akan memungkinkan pemrosesan yang berprioritas pada pekerjaan baru, dan pekerjaan-pekerjaan yang lebih lama ketika kapasitas tersedia. Teknik ini merupakan perkiraan LIFO pemrosesan dan memungkinkan pemrosesan sistem normal untuk semua pekerjaan baru.
- Gunakan antrean surat mati atau redrive untuk memindahkan pesan yang tidak dapat diproses dari backlog ke lokasi yang dapat dicari ulang dan selesaikan di lain waktu
- Coba lagi atau, apabila dapat ditoleransi, singkirkan pesan lama dengan membandingkan sekarang dengan stempel waktu pesan dan membuang pesan yang sudah tidak relevan dengan klien yang membuat permintaan.

Sumber daya

Praktik-praktik terbaik terkait:

- REL04-BP02 Menerapkan dependensi yang digabungkan secara longgar

- REL05-BP02 Permintaan Throttle
- REL05-BP03 Kontrol dan batasi panggilan coba lagi
- REL06-BP02 Tentukan dan hitung metrik (Agregasi)
- REL06-BP07 Memantau end-to-end penelusuran permintaan melalui sistem Anda

Dokumen terkait:

- Menghindari backlog antrean yang terlalu membebani
- Gagal Cepat (Fail Fast)
- Bagaimana saya bisa mencegah peningkatan backlog pesan di SQS antrian Amazon saya?
- Penyeimbangan Beban Elastis: Peralihan Zona
- Pengontrol Pemulihan Aplikasi Amazon: Kontrol perutean untuk failover lalu lintas

Contoh terkait:

- Pola Integrasi Korporat: Saluran Surat Mati

Video terkait:

- AWS Re:invent 2022 - Mengoperasikan aplikasi Multi-AZ yang sangat tersedia

Alat terkait:

- Amazon SQS
- Amazon MQ
- AWS IoT Core
- Amazon CloudWatch

REL05-BP05 Mengatur batas waktu klien

Atur batas waktu secara tepat pada koneksi dan permintaan, verifikasikan waktu tersebut secara sistematis, dan jangan selalu bergantung pada nilai default karena nilai tersebut mengabaikan hal-hal spesifik tentang beban kerja.

Hasil yang Diinginkan: Batas waktu klien harus mempertimbangkan biaya untuk klien, server, dan beban kerja yang berkaitan dengan proses menunggu permintaan yang memerlukan waktu sangat lama untuk diselesaikan. Karena penyebab batas waktu tidak mungkin diketahui secara pasti, maka klien harus menggunakan pengetahuannya mengenai layanan untuk membangun ekspektasi tentang kemungkinan-kemungkinan yang menjadi penyebab dan batas waktu yang tepat

Koneksi klien mengalami habis waktu berdasarkan nilai yang dikonfigurasi. Setelah mengalami batas waktu, klien mengambil keputusan untuk menunda dan mencobanya lagi atau membuka pemutus sirkuit. Pola-pola ini akan mencegah mengeluarkan permintaan yang dapat memperburuk kondisi kesalahan yang menyebabkannya.

Anti-pola umum:

- Tidak mengetahui batas waktu sistem atau batas waktu default.
- Tidak mengetahui waktu penyelesaian permintaan normal.
- Tidak mengetahui kemungkinan-kemungkinan yang menjadi penyebab permintaan membutuhkan waktu yang terlalu lama untuk diselesaikan, atau biaya untuk klien, layanan, atau kinerja beban kerja yang berkaitan dengan proses tunggu penyelesaian ini.
- Tidak mengetahui adanya kemungkinan jaringan rusak yang menyebabkan permintaan mengalami kegagalan hanya sesaat setelah batas waktu tercapai, dan biaya untuk klien dan kinerja beban kerja karena tidak mengadopsi batas waktu yang lebih singkat.
- Tidak menguji skenario batas waktu, baik untuk koneksi maupun permintaan.
- Mengatur batas waktu terlalu tinggi, yang berimbas pada waktu tunggu menjadi lama dan meningkatkan pemanfaatan sumber daya.
- Mengatur batas waktu terlalu rendah, sehingga mengakibatkan terjadinya kegagalan buatan.
- Mengabaikan pola-pola untuk menangani kesalahan-kesalahan batas waktu untuk panggilan jarak jauh seperti pemutus sirkuit dan percobaan ulang.
- Tidak mempertimbangkan pemantauan untuk angka kesalahan panggilan layanan, target latensi di tingkat layanan, dan outlier latensi. Metrik-metrik ini dapat memberikan wawasan tentang batas waktu yang agresif atau permisif

Manfaat menerapkan praktik terbaik ini: Waktu tunggu panggilan jarak jauh dikonfigurasi dan sistem dirancang untuk menangani batas waktu secara perlahan sehingga sumber daya akan dihemat ketika panggilan jarak jauh memberikan respons yang terlalu lambat dan kesalahan batas waktu ditangani secara perlahan oleh klien layanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Atur batas waktu koneksi dan batas waktu permintaan untuk panggilan dependensi layanan apa pun, serta secara umum untuk panggilan apa pun yang dilakukan di seluruh proses. Banyak kerangka kerja yang menawarkan kemampuan batas waktu bawaan, tetapi Anda harus tetap memperhatikan bahwa nilai default bawaan bisa saja tidak terbatas atau lebih tinggi dari nilai yang dapat diterima untuk sasaran layanan Anda. Nilai yang terlalu tinggi dapat mengurangi kegunaan waktu habis karena sumber daya akan terus terpakai saat klien menunggu terjadinya waktu habis. Nilai yang terlalu rendah dapat menyebabkan lalu lintas yang tinggi di backend serta akan meningkatkan latensi karena terlalu banyak permintaan yang dicoba ulang. Dalam beberapa kasus, hal ini dapat menyebabkan penghentian total karena semua permintaan dicoba ulang.

Pertimbangkan hal berikut saat Anda menentukan strategi batas waktu:

- Permintaan mungkin membutuhkan waktu pemrosesan yang lebih lama dari biasanya dikarenakan kontennya, terjadi gangguan pada layanan target, atau kegagalan partisi jaringan.
- Permintaan-permintaan dengan konten yang terlalu mahal dapat mengonsumsi sumber daya server dan klien yang tidak perlu. Dalam hal ini, membatasi waktu dan tidak mencoba ulang permintaan-permintaan tersebut dapat menghemat sumber daya. Layanan juga harus melindungi diri dari konten yang terlalu mahal dengan melakukan throttling dan batas waktu sisi server.
- Permintaan yang memakan waktu terlalu lama karena adanya gangguan layanan dapat diberikan batas waktu dan dicoba ulang. Pertimbangan harus diberikan pada biaya layanan untuk membuat permintaan dan melakuan percobaan ulang, tetapi jika penyebabnya adalah gangguan yang terbatas di suatu tempat, percobaan ulang kemungkinan tidak akan mahal dan akan mengurangi konsumsi sumber daya klien. Batas waktu juga dapat melepaskan sumber daya server, tergantung sifat gangguan.
- Permintaan-permintaan yang membutuhkan waktu penyelesaian yang lama karena permintaan atau respons gagal dikirimkan oleh jaringan dapat diberikan batas waktu dan dicoba ulang. Karena permintaan atau respons tidak dikirimkan, kegagalan akan terjadi, terlepas dari lamanya batas waktu yang dibreikan. Memberikan batas waktu pada kasus ini tidak akan melepaskan sumber daya server, tetapi akan melepaskan sumber daya klien dan meningkatkan kinerja beban kerja.

Manfaatkan pola desain yang sudah mapan seperti percobaan ulang dan pemutus sirkuit untuk menangani batas waktu dengan anggun dan mendukung pendekatan cepat gagal. AWS SDKsdan AWS CLImemungkinkan konfigurasi koneksi dan batas waktu permintaan dan untuk percobaan

ulang dengan backoff dan jitter eksponensial. AWS Lambdafungsi mendukung konfigurasi batas waktu, dan dengan AWS Step Functions, Anda dapat membangun pemutus sirkuit kode rendah yang memanfaatkan integrasi pra-bangun dengan layanan dan. AWS SDKs AWS App Mesh Envoy memberikan kemampuan batas waktu dan pemutus sirkuit.

Langkah-langkah implementasi

- Konfigurasikan batas waktu pada panggilan layanan jarak jauh dan manfaatkan fitur batas waktu bahasa bawaan atau pustaka batas waktu sumber terbuka.
- Saat beban kerja Anda melakukan panggilan dengan AWS SDK, tinjau dokumentasi untuk konfigurasi batas waktu khusus bahasa.
  - Python
  - PHP
  - .NET
  - Ruby
  - Java
  - Go
  - Node.js
  - C++
- Saat menggunakan AWS SDKs atau AWS CLI perintah di beban kerja Anda, konfigurasikan nilai batas waktu default dengan menyetel default AWS konfigurasi untuk dan. `connectTimeoutInMillis` `tlsNegotiationTimeoutInMillis`
- Terapkan opsi baris perintah `cli-connect-timeout` dan `cli-read-timeout` untuk mengontrol AWS CLI perintah satu kali ke AWS layanan.
- Lakukan pemantauan panggilan layanan jarak jauh untuk memeriksa batas waktu, dan atur alarm pada kesalahan persisten sehingga Anda dapat menangani skenario kesalahan secara proaktif.
- Menerapkan CloudWatch Metrik dan deteksi CloudWatch anomali pada tingkat kesalahan panggilan, tujuan tingkat layanan untuk latensi, dan latensi outlier untuk memberikan wawasan tentang mengelola batas waktu yang terlalu agresif atau permisif.
- Konfigurasikan batas waktu pada fungsi Lambda.
- APIKlien gateway harus menerapkan percobaan ulang mereka sendiri saat menangani batas waktu. APIGateway mendukung batas waktu integrasi 50 milidetik hingga 29 detik untuk integrasi hilir dan tidak mencoba lagi saat integrasi meminta batas waktu.

- Implementasikan pemutus sirkuit untuk menghindari pembuatan panggilan jarak jauh ketika waktu habis. Buka sirkuit untuk menghindari kegagalan panggilan dan tutup sirkuit saat panggilan memberikan respons secara normal.
- Untuk beban kerja berbasis kontainer, tinjau fitur App Mesh Envoy untuk memanfaatkan batas waktu bawaan dan pemutus sirkuit.
- Gunakan AWS Step Functions untuk membangun pemutus sirkuit kode rendah untuk panggilan layanan jarak jauh, terutama saat memanggil integrasi Step Functions AWS asli SDKs dan didukung untuk menyederhanakan beban kerja Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- REL05-BP03 Kontrol dan batasi panggilan coba lagi
- REL05-BP04 Gagal cepat dan batasi antrian
- REL06-BP07 Memantau end-to-end penelusuran permintaan melalui sistem Anda

Dokumen terkait:

- AWS SDK: Coba Ulang dan Batas Waktu
- Amazon Builders' Library: Batas waktu, percobaan ulang, dan penundaan dengan jitter
- Kuota Amazon API Gateway dan catatan penting
- AWS Command Line Interface: Opsi baris perintah
- AWS SDK for Java 2.x: Konfigurasikan Batas API Waktu
- AWS Botocore menggunakan objek konfigurasi dan Referensi Config
- AWS SDK for .NET: Percobaan Ulang dan Batas Waktu
- AWS Lambda: Mengonfigurasi opsi fungsi Lambda

Contoh terkait:

- Menggunakan pola pemutus sirkuit dengan AWS Step Functions dan Amazon DynamoDB
- Martin Fowler: CircuitBreaker

Alat terkait:

- AWS SDKs
- AWS Lambda
- Amazon SQS
- AWS Step Functions
- AWS Command Line Interface

REL05-BP06 Membuat sistem tanpa kewarganegaraan jika memungkinkan

Sistem seharusnya tidak memerlukan state, atau seharusnya mengalihkan state sedemikian rupa sehingga di antara permintaan klien yang berbeda, tidak ada dependensi di penyimpanan data lokal di disk dan memori. Hal ini membuat server dapat diganti sesuai keinginan tanpa menimbulkan dampak ketersediaan.

Ketika pengguna atau layanan berinteraksi dengan sebuah aplikasi, mereka sering kali melakukan serangkaian interaksi yang membentuk sebuah sesi. Sesi adalah data unik bagi para pengguna yang lama berada di antara permintaan ketika mereka menggunakan aplikasi. Aplikasi stateless adalah sebuah aplikasi yang tidak memerlukan pengetahuan tentang interaksi sebelumnya dan tidak menyimpan informasi sesi.

Setelah dirancang untuk menjadi stateless, Anda kemudian dapat menggunakan layanan komputasi tanpa server, seperti atau. AWS Lambda AWS Fargate

Selain penggantian server, manfaat lain dari aplikasi stateless adalah mereka dapat menskalakan secara horizontal karena salah satu sumber daya komputasi yang tersedia (seperti EC2 instance dan AWS Lambda fungsi) dapat melayani permintaan apa pun.

Manfaat menerapkan praktik terbaik ini: Sistem yang dirancang untuk menjadi stateless lebih mudah beradaptasi dengan penskalaan horizontal, sehingga akan memungkinkan Anda untuk menambah atau menghapus kapasitas berdasarkan lalu lintas dan permintaan yang berfluktuasi. Sistem ini juga memiliki sifat yang tahan terhadap kegagalan-kegagalan yang terjadi dan memberikan fleksibilitas serta ketangkasan dalam pengembangan aplikasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jadikan aplikasi Anda stateless. Aplikasi stateless memungkinkan penskalaan horizontal dan toleran terhadap kegagalan yang dialami simpul individual. Lakukan analisis terhadap dan pahami komponen-komponen aplikasi Anda yang mempertahankan state di dalam arsitektur. Hal ini akan

membantu Anda menilai dampak potensial dari melakukan transisi ke desain stateless. Arsitektur stateless memisahkan data pengguna dan memindahkan data sesi. Hal ini dapat memberikan fleksibilitas untuk menskalakan setiap komponen secara independen untuk memenuhi permintaan beban kerja yang beragam dan mengoptimalkan pemanfaatan sumber daya.

Langkah-langkah implementasi

- Identifikasi dan pahami komponen-komponen stateful yang ada dalam aplikasi Anda.
- Pisahkan data dengan memisahkan dan mengelola data pengguna dari logika aplikasi inti.
  - Amazon Cognito dapat memisahkan data pengguna dari kode aplikasi dengan menggunakan fitur, seperti kolam identitas, kumpulan pengguna, dan Amazon Cognito Sync.
  - Anda dapat menggunakan AWS Secrets Manager untuk memisahkan data pengguna dengan menyimpan rahasia di lokasi yang aman dan tersentralisasi. Ini berarti kode aplikasi tidak perlu menyimpan rahasia, sehingga membuatnya menjadi lebih aman.
  - Pertimbangkan untuk menggunakan Amazon S3 untuk menyimpan data besar dan tidak terstruktur, seperti gambar dan dokumen. Aplikasi Anda dapat mengambil data tersebut saat diperlukan, sehingga akan menghilangkan kebutuhan untuk menyimpannya di dalam memori.
  - Gunakan Amazon DynamoDB untuk menyimpan informasi seperti profil pengguna. Aplikasi Anda dapat melakukan kueri terhadap data tersebut hampir dalam waktu nyata.
- Pindahkan data sesi ke basis data, cache, atau file eksternal.
  - Amazon ElastiCache, Amazon DynamoDB, Amazon Elastic File System (Amazon), dan EFS Amazon MemoryDB adalah contoh layanan yang dapat Anda AWS gunakan untuk membongkar data sesi.
- Rancang sebuah arsitektur stateless setelah Anda mengidentifikasi state dan data pengguna mana yang perlu dipertahankan dengan solusi penyimpanan pilihan Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan

Dokumen terkait:

- Amazon Builders' Library: Menghindari fallback dalam sistem terdistribusi
- Amazon Builders' Library: Menghindari backlog antrean yang tidak dapat diatasi

- Amazon Builders' Library: Tantangan dan strategi caching
- Praktik Terbaik untuk Tingkat Web Stateless di AWS

REL05-BP07 Menerapkan tuas darurat

Tuas darurat adalah proses cepat yang dapat memitigasi dampak ketersediaan pada beban kerja.

Tuas darurat bekerja dengan cara menonaktifkan, melakukan throttling, atau mengubah perilaku komponen atau dependensi dengan menggunakan mekanisme yang diketahui dan sudah diuji. Hal ini dapat mengurangi gangguan beban kerja yang disebabkan oleh kelelahan sumber daya karena peningkatan permintaan yang terjadi secara tidak terduga dan mengurangi dampak kegagalan pada komponen non-kritis yang ada dalam beban kerja Anda.

Hasil yang diinginkan: Dengan menerapkan tuas-tuas darurat, Anda dapat menetapkan proses yang diketahui baik untuk menjaga ketersediaan komponen penting dalam beban kerja Anda. Beban kerja akan mengalami degradasi secara perlahan (graceful degradation) dan terus menjalankan fungsi-fungsi kritis bisnisnya selama tuas darurat masih dalam keadaan aktif. Untuk detail lebih lanjut tentang degradasi anggun, lihat REL05-BP01 Menerapkan degradasi anggun untuk mengubah dependensi keras yang berlaku menjadi dependensi lunak.

Anti-pola umum:

- Kegagalan dependensi non-kritis akan berdampak pada ketersediaan beban kerja inti Anda.
- Tidak menguji atau memverifikasi perilaku komponen-komponen kritis saat terjadi gangguan komponen non-kritis.
- Tidak ada kriteria yang jelas dan deterministik yang ditentukan untuk pengaktifan atau penonaktifan sebuah tuas darurat.

Manfaat menerapkan praktik terbaik ini: Menerapkan tuas darurat dapat meningkatkan ketersediaan komponen penting dalam beban kerja Anda dengan menyediakan resolver Anda proses-proses yang telah ditetapkan untuk menanggapi lonjakan permintaan yang tidak terduga atau kegagalan dependensi non-kritis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Identifikasi komponen-komponen kritis yang ada dalam beban kerja Anda.

- Buat agar rancangan dan arsitek komponen kritis dalam beban kerja Anda dapat menahan kegagalan komponen non-kritis.
- Lakukan pengujian untuk memvalidasi perilaku komponen-komponen kritis Anda saat terjadi kegagalan komponen non-kritis.
- Tentukan dan pantau metrik atau pemicu yang relevan untuk memulai prosedur tuas darurat.
- Tentukan prosedur (manual atau otomatis) yang mencakup tuas darurat.

Langkah-langkah implementasi

- Identifikasi komponen-komponen kritis bagi bisnis yang ada dalam beban kerja Anda.
  - Setiap komponen teknis yang ada dalam beban kerja Anda harus dipetakan ke fungsi bisnisnya yang relevan dan diberi peringkat sebagai komponen kritis atau non-kritis. Untuk contoh fungsionalitas kritis dan non-kritis yang ada di Amazon, silakan lihat Kapan Saja Bisa Menjadi Hari yang Prima: Bagaimana Penelusuran Pencarian Amazon.com Menggunakan Perekayasaan Chaos untuk Menangani Lebih dari 84K Permintaan Per Detik.
  - Ini merupakan keputusan teknis sekaligus keputusan bisnis, dan berbeda-beda berdasarkan organisasi dan beban kerja.
- Buat agar rancangan dan arsitek komponen kritis dalam beban kerja Anda dapat menahan kegagalan komponen non-kritis.
  - Saat melakukan analisis dependensi, pertimbangkan semua mode kegagalan yang dapat terjadi, dan pastikan bahwa mekanisme tuas darurat Anda memberikan fungsionalitas kritis pada komponen-komponen hilir.
- Lakukan pengujian untuk melakukan validasi terhadap perilaku komponen kritis Anda saat tuas darurat Anda diaktifkan.
  - Hindari perilaku bimodal. Untuk detail lebih lanjut, lihat REL11-BP05 Gunakan stabilitas statis untuk mencegah perilaku bimodal.
- Tentukan, pantau, dan munculkan peringatan pada metrik-metrik yang relevan untuk memulai prosedur tuas darurat.
  - Beban kerja Anda menentukan metrik yang tepat untuk dipantau. Beberapa contoh metrik adalah latensi atau jumlah permintaan yang gagal ke sebuah dependensi.
- Tentukan prosedur yang mencakup tuas darurat, bisa manual atau otomatis.
  - Ini mungkin termasuk mekanisme seperti memuat pelepasan, permintaan throttling, atau melaksanakan degradasi yang tepat.

Sumber daya

Praktik-praktik terbaik terkait:

- REL05-BP01 Menerapkan degradasi anggun untuk mengubah dependensi keras yang berlaku menjadi dependensi lunak
- REL05-BP02 Permintaan Throttle
- REL11-BP05 Gunakan stabilitas statis untuk mencegah perilaku bimodal

Dokumen terkait:

- Melakukan otomatisasi deployment secara aman dan otonom
- Kapan Saja Bisa Menjadi Hari yang Prima: Bagaimana Penelusuran Pencarian Amazon.com Menggunakan Perekayasaan Chaos untuk Menangani Lebih dari 84K Permintaan Per Detik

Video terkait:

- AWS Re: Invent 2020: Keandalan, konsistensi, dan kepercayaan diri melalui kekekalan

## Manajemen perubahan

Pertanyaan

- REL6. Bagaimana cara memantau sumber daya beban kerja Anda?
- REL7. Bagaimana cara mendesain beban kerja Anda untuk beradaptasi dengan perubahan dalam permintaan?
- REL8. Bagaimana cara mengimplementasikan perubahan?

### REL6. Bagaimana cara memantau sumber daya beban kerja Anda?

Log dan metrik merupakan alat yang luar biasa untuk mendapatkan wawasan mengenai kondisi beban kerja Anda. Anda dapat mengonfigurasikan beban kerja Anda untuk memantau log dan metrik serta mengirimkan notifikasi ketika ambang batas terlampaui atau ada peristiwa signifikan yang terjadi. Pemantauan memungkinkan beban kerja Anda untuk mengenali ketika ambang batas kinerja rendah terlampaui atau ada kegagalan yang terjadi, sehingga pemulihan dapat terjadi secara otomatis untuk menanggapinya.

Praktik terbaik

- REL06-BP01 Memantau semua komponen untuk beban kerja (Generasi)
- REL06-BP02 Tentukan dan hitung metrik (Agregasi)
- REL06-BP03 Kirim pemberitahuan (Pemrosesan waktu nyata dan mengkhawatirkan)
- REL06-BP04 Otomatiskan respons (Pemrosesan waktu nyata dan mengkhawatirkan)
- REL06-BP05 Menganalisis log
- REL06-BP06 Melakukan tinjauan secara teratur
- REL06-BP07 Memantau end-to-end penelusuran permintaan melalui sistem Anda

REL06-BP01 Memantau semua komponen untuk beban kerja (Generasi)

Pantau komponen beban kerja dengan Amazon CloudWatch atau alat pihak ketiga. Memantau AWS layanan dengan AWS Health Dashboard.

Semua komponen beban kerja Anda harus dipantau, termasuk front-end, logika bisnis, dan tingkat penyimpanan. Tetapkan metrik utama, jelaskan cara mengekstraknya dari log (jika diperlukan), dan tetapkan ambang batas untuk menginvokasi peristiwa alarm yang sesuai. Pastikan metrik relevan dengan indikator kinerja utama (KPIs) beban kerja Anda, dan gunakan metrik dan log untuk mengidentifikasi tanda-tanda peringatan dini degradasi layanan. Misalnya, metrik yang terkait dengan hasil bisnis seperti jumlah pesanan yang berhasil diproses per menit, dapat menunjukkan masalah beban kerja lebih cepat daripada metrik teknis, seperti CPU Pemanfaatan. Gunakan AWS Health Dasbor untuk tampilan yang dipersonalisasi tentang kinerja dan ketersediaan AWS layanan yang mendasari AWS sumber daya Anda.

Pemantauan di cloud menawarkan peluang-peluang baru. Sebagian besar penyedia cloud telah mengembangkan kait yang dapat disesuaikan dan dapat memberikan wawasan untuk membantu Anda memantau beberapa lapisan beban kerja Anda. AWS Layanan seperti Amazon CloudWatch menerapkan algoritma statistik dan pembelajaran mesin untuk terus menganalisis metrik sistem dan aplikasi, menentukan garis dasar normal, dan anomali permukaan dengan intervensi pengguna minimal. Algoritma deteksi anomali bertanggung jawab atas perubahan-perubahan musiman dan tren metrik.

AWS membuat banyak informasi pemantauan dan log tersedia untuk konsumsi yang dapat digunakan untuk menentukan metrik, change-in-demand proses, dan mengadopsi teknik pembelajaran mesin spesifik beban kerja terlepas dari keahlian ML.

Selain itu, pantau semua titik akhir eksternal Anda untuk memastikan bahwa mereka tidak bergantung pada implementasi dasar Anda. Pemantauan aktif ini dapat dilakukan dengan transaksi sintetis (kadang-kadang disebut sebagai canary pengguna, tetapi jangan disamakan dengan deployment canary) yang secara berkala menjalankan sejumlah tindakan yang cocok dengan tugas-tugas umum yang dilakukan oleh klien dari beban kerja. Buat tugas-tugas ini berdurasi singkat dan pastikan untuk tidak membebani beban kerja Anda saat melakukan pengujian. Amazon CloudWatch Synthetics memungkinkan Anda membuat kenari sintetis untuk memantau titik akhir Anda dan. APIs Anda juga dapat menggabungkan simpul-simpul klien canary sintetis dengan konsol AWS X-Ray untuk mengidentifikasi canary sintetis mana yang mengalami masalah berupa error, fault, atau tingkat throttling untuk jangka waktu yang dipilih.

Hasil yang Diinginkan:

Kumpulkan dan gunakan metrik-metrik kritis dari semua komponen beban kerja untuk memastikan keandalan beban kerja dan pengalaman pengguna yang optimal. Dengan mendeteksi bahwa sebuah beban kerja tidak mencapai hasil bisnis, Anda dapat dengan cepat mengumumkan terjadinya sebuah bencana dan kemudian segera melakukan pemulihan dari insiden.

Anti-pola umum:

- Melakukan pemantauan hanya untuk antarmuka eksternal beban kerja Anda.
- Tidak menghasilkan metrik khusus beban kerja apa pun dan hanya mengandalkan metrik yang diberikan kepada Anda oleh layanan yang digunakan beban kerja Anda. AWS
- Hanya menggunakan metrik teknis dalam beban kerja Anda dan tidak memantau metrik apa pun yang terkait dengan non-teknis yang disumbangkan KPIs oleh beban kerja.
- Mengandalkan lalu lintas produksi dan pemeriksaan kondisi sederhana untuk melakukan pemantauan dan evaluasi terhadap status (state) beban kerja.

Manfaat menerapkan praktik terbaik ini: Dengan memantau semua tingkatan di beban kerja Anda akan memungkinkan Anda untuk dapat lebih cepat mengantisipasi dan menyelesaikan masalah di komponen dalam beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

1. Aktifkan pencatatan log jika tersedia. Data pemantauan harus diperoleh dari semua komponen beban kerja. Aktifkan pencatatan log tambahan, seperti S3 Access Logs, dan mengizinkan beban

kerja Anda untuk mencatat log khusus beban kerja. Kumpulkan metrik untukCPU, I/O jaringan, dan rata-rata I/O disk dari layanan seperti Amazon, Amazon, ECS EKS Amazon, Elastic Load EC2 Balancing, dan Amazon. AWS Auto Scaling EMR Lihat AWS Layanan yang Mempublikasikan CloudWatch Metrik untuk daftar AWS layanan yang memublikasikan metrik. CloudWatch

2. Tinjau semua metrik default dan telusuri celah pengumpulan data apa pun. Setiap layanan menghasilkan metrik default. Dengan mengumpulkan metrik default, Anda dapat lebih memahami dependensi yang terjadi antar komponen beban kerja dan bagaimana keandalan dan kinerja komponen memengaruhi beban kerja tersebut. Anda juga dapat membuat dan mempublikasikan metrik Anda sendiri untuk CloudWatch menggunakan AWS CLI atau. API
3. Evaluasi semua metrik untuk memutuskan mana yang akan diperingatkan untuk setiap AWS layanan di beban kerja Anda. Anda dapat memilih subset metrik yang memiliki dampak besar terhadap keandalan beban kerja. Berfokus pada metrik-metrik dan ambang batas kritis akan memungkinkan Anda untuk menyempurnakan jumlah peringatan dan dapat membantu Anda meminimalkan positif palsu.
4. Tetapkan peringatan dan proses pemulihan beban kerja Anda setelah peringatan diinvokasi. Mendefinisikan lansiran memungkinkan Anda untuk dengan cepat memberi tahu, meningkatkan, dan mengikuti langkah-langkah yang diperlukan untuk pulih dari suatu insiden dan memenuhi Tujuan Waktu Pemulihan yang ditentukan (). RTO Anda dapat menggunakan CloudWatch Alarm Amazon untuk menjalankan alur kerja otomatis dan memulai prosedur pemulihan berdasarkan ambang batas yang ditentukan.
5. Jelajahi penggunaan transaksi sintetis untuk mengumpulkan data yang relevan tentang state beban kerja. Pemantauan sintetis mengikuti rute yang sama dan menjalankan tindakan-tindakan yang sama seperti seorang pelanggan, sehingga memungkinkan Anda untuk terus memverifikasi pengalaman pelanggan Anda bahkan saat Anda tidak memiliki lalu lintas pelanggan apa pun pada beban kerja Anda. Dengan menggunakan transaksi sintetis, Anda dapat menemukan masalah-masalah sebelum para pelanggan Anda menemukannya.

Sumber daya

Praktik-praktik terbaik terkait:

- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan

Dokumen terkait:

- Memulai AWS Health Dashboard Anda — Kesehatan akun Anda

- AWS Layanan yang Mempublikasikan CloudWatch Metrik
- Pencatatan Log Akses Untuk Penyeimbang Beban Jaringan Anda
- Pencatatan Log Akses untuk penyeimbang beban aplikasi Anda
- Mengakses CloudWatch Log Amazon untuk AWS Lambda
- Pencatatan Log Akses Server Amazon S3
- Mengaktifkan Pencatatan Log Akses untuk Penyeimbang Beban Klasik Anda
- Mengekspor data log ke Amazon S3
- Instal CloudWatch agen pada EC2 instans Amazon
- Memublikasikan Metrik Kustom
- Menggunakan CloudWatch Dasbor Amazon
- Menggunakan CloudWatch Metrik Amazon
- Menggunakan Canaries (Amazon CloudWatch Synthetics)
- Apa itu CloudWatch Log Amazon?

Panduan pengguna:

- Membuat jejak
- Memantau metrik memori dan disk untuk instans Amazon EC2 Linux
- Menggunakan CloudWatch Log dengan instance kontainer
- Log Alur VPC
- Apa itu Amazon DevOps Guru?
- Apa itu AWS X-Ray?

Blog terkait:

- Debugging dengan Amazon CloudWatch Synthetics dan AWS X-Ray

Contoh dan lokakarya terkait:

- Lab AWS Well-Architected: Keunggulan Operasional - Pemantauan Dependensi
- Amazon Builders' Library: Instrumentasi sistem terdistribusi untuk visibilitas operasional
- Lokakarya observabilitas

REL06-BP02 Tentukan dan hitung metrik (Agregasi)

Simpan data log dan terapkan filter saat diperlukan untuk menghitung metrik, seperti jumlah peristiwa log tertentu, atau latensi yang dihitung dari stempel waktu peristiwa log.

Amazon CloudWatch dan Amazon S3 berfungsi sebagai lapisan agregasi dan penyimpanan utama. Untuk beberapa layanan, seperti Elastic Load Balancing AWS Auto Scaling dan Elastic Load Balancing, metrik default disediakan secara default untuk CPU beban atau latensi permintaan rata-rata di seluruh cluster atau instance. Untuk layanan streaming, seperti VPC Flow Logs dan AWS CloudTrail, data peristiwa diteruskan ke CloudWatch Log dan Anda perlu menentukan dan menerapkan filter metrik untuk mengekstrak metrik dari data peristiwa. Ini memberi Anda data deret waktu, yang dapat berfungsi sebagai input ke CloudWatch alarm yang Anda tentukan untuk memanggil peringatan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Tetapkan dan hitung metrik (Agregasi). Simpan data log dan terapkan filter saat diperlukan untuk menghitung metrik, seperti jumlah peristiwa log tertentu, atau latensi yang dihitung dari stempel waktu peristiwa log
  - Filter metrik menentukan istilah dan pola yang harus dicari dalam data log saat dikirim ke CloudWatch Log. CloudWatch Log menggunakan filter metrik ini untuk mengubah data log menjadi CloudWatch metrik numerik yang dapat Anda buat grafik atau nyalakan alarm.
    - Mencari dan Menyaring Data Log
  - Gunakan pihak ketiga tepercaya untuk melakukan agregasi log.
    - Ikuti instruksi pihak ketiga. Sebagian besar produk pihak ketiga terintegrasi dengan CloudWatch dan Amazon S3.
  - Beberapa AWS layanan dapat mempublikasikan log langsung ke Amazon S3. Jika kebutuhan utama Anda untuk log adalah penyimpanan di Amazon S3, maka Anda dapat dengan mudah meminta layanan yang memproduksi log untuk mengirimnya langsung ke Amazon S3 tanpa perlu menyiapkan infrastruktur tambahan.
    - Mengirim Log Langsung ke Amazon S3

Sumber daya

Dokumen terkait:

- Kueri Contoh Wawasan CloudWatch Log Amazon
- Debugging dengan Amazon CloudWatch Synthetics dan AWS X-Ray
- Lokakarya Satu Observabilitas
- Mencari dan Menyaring Data Log
- Mengirim Log Langsung ke Amazon S3
- Amazon Builders' Library: Instrumentasi sistem terdistribusi untuk visibilitas operasional

REL06-BP03 Kirim pemberitahuan (Pemrosesan waktu nyata dan mengkhawatirkan)

Ketika organisasi mendeteksi potensi masalah, mereka mengirimkan notifikasi dan peringatan waktu nyata kepada personel dan sistem yang sesuai untuk merespons masalah ini dengan cepat dan efektif.

Hasil yang diinginkan: Respons yang cepat terhadap peristiwa operasional dapat terjadi melalui konfigurasi alarm yang relevan berdasarkan metrik layanan dan aplikasi. Ketika ambang batas alarm dilanggar, personel dan sistem yang sesuai mendapatkan notifikasi sehingga mereka dapat mengatasi masalah-masalah yang mendasarinya.

Anti-pola umum:

- Mengonfigurasi alarm dengan ambang batas yang terlalu tinggi, akan mengakibatkan kegagalan untuk mengirim notifikasi-notifikasi penting.
- Mengonfigurasi alarm dengan ambang batas yang terlalu rendah, akan menyebabkan tidak adanya tindakan atas notifikasi-notifikasi penting karena kebisingan notifikasi yang berlebihan.
- Tidak memperbarui alarm dan ambang batasnya saat penggunaan berubah.
- Untuk alarm yang paling sesuai untuk ditangani melalui tindakan otomatis, mengirim notifikasi ke personel alih-alih membuat tindakan otomatis, akan menyebabkan terjadinya pengiriman notifikasi yang berlebihan.

Manfaat menerapkan praktik terbaik ini: Mengirimkan notifikasi dan pemberitahuan waktu nyata kepada personel dan sistem yang sesuai akan memungkinkan dilakukannya deteksi dini terhadap masalah dan memungkinkan respons yang cepat terhadap insiden operasional.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Beban kerja harus dilengkapi dengan pemrosesan dan peringatan alarm waktu nyata untuk meningkatkan pendeteksian masalah yang dapat memengaruhi ketersediaan aplikasi dan berfungsi sebagai pemicu respons otomatis. Organisasi dapat melakukan pemrosesan dan peringatan alarm waktu nyata dengan menciptakan peringatan dengan metrik yang ditentukan untuk menerima notifikasi setiap kali peristiwa signifikan terjadi atau sebuah metrik melebihi ambang batas.

Amazon CloudWatch memungkinkan Anda membuat alarm metrik dan komposit menggunakan CloudWatch alarm berdasarkan ambang batas statis, deteksi anomali, dan kriteria lainnya. Untuk detail selengkapnya tentang jenis alarm yang dapat Anda konfigurasikan CloudWatch, lihat bagian alarm pada CloudWatch dokumentasi.

Anda dapat membuat tampilan metrik dan peringatan yang disesuaikan dari AWS sumber daya Anda untuk tim Anda menggunakan dasbor. CloudWatch Halaman beranda yang dapat disesuaikan di CloudWatch konsol memungkinkan Anda memantau sumber daya dalam satu tampilan di beberapa Wilayah.

Alarm dapat melakukan satu atau beberapa tindakan, seperti mengirim notifikasi ke SNStopik Amazon, melakukan tindakan Amazon, atau EC2 tindakan Penskalaan EC2Otomatis Amazon, atau membuat OpsItem insiden atau masuk. AWS Systems Manager

Amazon CloudWatch menggunakan Amazon SNS untuk mengirim pemberitahuan ketika alarm berubah status, memberikan pengiriman pesan dari penerbit (produsen) ke pelanggan (konsumen). Untuk detail selengkapnya tentang mengatur SNS notifikasi Amazon, lihat Mengonfigurasi Amazon SNS.

CloudWatch mengirimkan EventBridgeperistiwa setiap kali CloudWatch alarm dibuat, diperbarui, dihapus, atau statusnya berubah. Anda dapat menggunakan EventBridge dengan peristiwa ini untuk membuat aturan yang melakukan tindakan, seperti memberi tahu Anda setiap kali status alarm berubah atau secara otomatis memicu peristiwa di akun Anda menggunakan otomatisasi Systems Manager.

Kapan sebaiknya Anda menggunakan EventBridge atau AmazonSNS?

Keduanya EventBridge dan Amazon SNS dapat digunakan untuk mengembangkan aplikasi berbasis peristiwa, dan pilihan Anda akan tergantung pada kebutuhan spesifik Anda.

Amazon EventBridge direkomendasikan saat Anda ingin membuat aplikasi yang bereaksi terhadap peristiwa dari aplikasi, aplikasi SaaS, dan layanan Anda sendiri. AWS EventBridge adalah satu-

satunya layanan berbasis acara yang terintegrasi langsung dengan mitra SaaS pihak ketiga. EventBridge juga secara otomatis menyerap peristiwa dari lebih dari 200 AWS layanan tanpa mengharuskan pengembang untuk membuat sumber daya apa pun di akun mereka.

EventBridge menggunakan struktur JSON berbasis yang ditentukan untuk acara, dan membantu Anda membuat aturan yang diterapkan di seluruh badan acara untuk memilih acara untuk diteruskan ke target. EventBridge Saat ini mendukung lebih dari 20 AWS layanan sebagai target, termasuk AWS Lambda, Amazon SQS, AmazonSNS, Amazon Kinesis Data Streams, dan Amazon Data Firehose.

Amazon SNS direkomendasikan untuk aplikasi yang membutuhkan fan out tinggi (ribuan atau jutaan titik akhir). Pola umum yang kami lihat adalah bahwa pelanggan menggunakan Amazon SNS sebagai target aturan mereka untuk memfilter peristiwa yang mereka butuhkan, dan menyebar ke beberapa titik akhir.

Pesan tidak terstruktur dan dapat dalam format apa pun. Amazon SNS mendukung penerusan pesan ke enam jenis target yang berbeda, termasuk Lambda, SQS Amazon, /S endpointHTTP,, push selulerSMS, dan email. Latensi SNS khas Amazon di bawah 30 milidetik. Berbagai AWS layanan mengirim SNS pesan Amazon dengan mengonfigurasi layanan untuk melakukannya (lebih dari 30, termasuk Amazon, Amazon S3EC2, dan Amazon RDS).

Langkah-langkah implementasi

1. Buat alarm menggunakan CloudWatch alarm Amazon.
   a. Alarm metrik memonitor CloudWatch metrik tunggal atau ekspresi yang bergantung pada CloudWatch metrik. Alarm memulai satu atau beberapa tindakan berdasarkan nilai metrik atau ekspresi dibandingkan dengan ambang batas selama interval waktu tertentu. Tindakan dapat terdiri dari mengirim pemberitahuan ke SNStopik Amazon, melakukan tindakan Amazon atau EC2 tindakan EC2Auto Scaling Amazon, atau membuat OpsItem atau insiden di. AWS Systems Manager
   b. Sebuah alarm gabungan terdiri dari ekspresi aturan yang mempertimbangkan kondisi alarm dari alarm-alarm lain yang telah Anda buat. Alarm gabungan hanya memasuki status alarm jika semua kondisi aturan terpenuhi. Alarm yang ditentukan dalam ekspresi aturan suatu alarm komposit dapat mencakup alarm-alarm metrik dan alarm gabungan tambahan. Alarm komposit dapat mengirim SNS pemberitahuan Amazon ketika statusnya berubah dan dapat membuat Systems Manager OpsItemsatau insiden saat memasuki status alarm, tetapi mereka tidak dapat melakukan tindakan Amazon EC2 atau Auto Scaling.
2. Siapkan SNSnotifikasi Amazon. Saat membuat CloudWatch alarm, Anda dapat menyertakan SNS topik Amazon untuk mengirim pemberitahuan saat alarm berubah status.

3. Buat aturan EventBridge yang cocok dengan CloudWatch alarm yang ditentukan. Setiap aturan mendukung beberapa target, termasuk fungsi Lambda. Misalnya, Anda dapat menentukan alarm yang dimulai saat ruang disk yang tersedia hampir habis, yang memicu fungsi Lambda melalui EventBridge aturan, untuk membersihkan ruang. Untuk detail lebih lanjut tentang EventBridge target, lihat EventBridge target.

Sumber daya

Praktik terbaik Well-Architected terkait:

- REL06-BP01 Memantau semua komponen untuk beban kerja (Generasi)
- REL06-BP02 Tentukan dan hitung metrik (Agregasi)
- REL12-BP01 Gunakan pedoman untuk menyelidiki kegagalan

Dokumen terkait:

- Amazon CloudWatch
- CloudWatch Wawasan log
- Menggunakan CloudWatch alarm Amazon
- Menggunakan CloudWatch dasbor Amazon
- Menggunakan CloudWatch metrik Amazon
- Menyiapkan SNS notifikasi Amazon
- CloudWatch deteksi anomali
- CloudWatch Perlindungan data log
- Amazon EventBridge
- Amazon Simple Notification Service

Video terkait:

- reinvent 2022 video observabilitas
- AWS re:invent 2022 - Praktik terbaik observabilitas di Amazon

Contoh terkait:

- Lokakarya Satu Observabilitas

- Amazon EventBridge AWS Lambda dengan kontrol umpan balik oleh Amazon CloudWatch Alarms

REL06-BP04 Otomatiskan respons (Pemrosesan waktu nyata dan mengkhawatirkan)

Gunakan otomatisasi untuk melakukan tindakan ketika peristiwa terdeteksi, misalnya, mengganti komponen yang rusak.

Pemrosesan alarm waktu nyata secara otomatis diimplementasikan agar sistem dapat mengambil tindakan-tindakan korektif dengan cepat dan berupaya mencegah terjadinya kegagalan atau penurunan layanan ketika alarm terpicu. Respons otomatis terhadap alarm dapat mencakup penggantian komponen yang mengalami kegagalan, penyesuaian kapasitas komputasi, pengalihan lalu lintas ke host yang dalam kondisi sehat, zona ketersediaan, atau wilayah lain, dan pemberitahuan operator.

Hasil yang diinginkan: Alarm waktu nyata diidentifikasi, dan pemrosesan alarm otomatis diatur untuk meminta tindakan yang tepat yang diambil untuk mempertahankan tujuan tingkat layanan dan perjanjian tingkat layanan (). SLAs Otomatisasi dapat berupa banyak hal, dari aktivitas pemulihan diri sebuah komponen hingga failover seluruh situs.

Anti-pola umum:

- Tidak memiliki inventaris atau katalog alarm waktu nyata utama yang jelas.
- Tidak ada respons otomatis terhadap alarm-alarm kritis (misalnya, penskalaan otomatis berjalan ketika komputasi hampir habis).
- Tindakan respons alarm yang kontradiktif.
- Tidak ada prosedur operasi standar (SOPs) bagi operator untuk mengikuti ketika mereka menerima pemberitahuan peringatan.
- Tidak memantau perubahan konfigurasi, padahal perubahan konfigurasi yang tidak terdeteksi dapat menyebabkan waktu henti terhadap beban kerja.
- Tidak memiliki strategi untuk membatalkan perubahan konfigurasi yang tidak diinginkan.

Manfaat menerapkan praktik terbaik ini: Melakukan otomatisasi atas pemrosesan alarm dapat meningkatkan ketahanan sistem. Sistem mengambil tindakan-tindakan korektif secara otomatis, sehingga akan mengurangi aktivitas manual yang memberi peluang adanya intervensi manusia yang rawan menyebabkan kesalahan. Operasi beban kerja memenuhi tujuan-tujuan ketersediaan, dan mengurangi gangguan layanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Untuk mengelola peringatan secara efektif dan mengotomatiskan responsnya, Anda harus mengkategorikan peringatan berdasarkan tingkat kekritisan dan dampaknya, mendokumentasikan prosedur respons, dan merencanakan respons sebelum menentukan peringkat tugas.

Identifikasi tugas yang membutuhkan tindakan-tindakan tertentu (sering kali diperinci dalam runbook), dan periksa semua runbook dan playbook untuk menentukan tugas mana yang dapat diotomatisasi. Jika tindakan dapat ditentukan, tindakan tersebut sering kali dapat diotomatisasi. Jika tindakan tidak dapat diotomatisasi, dokumentasikan langkah-langkah manual di sebuah SOP dan latih operator pada mereka. Terus cari peluang otomatisasi pada proses-proses yang masih dilakukan secara manual agar Anda dapat membuat dan menerapkan rencana untuk mengotomatiskan respons peringatan.

Langkah-langkah implementasi

1. Buat inventaris alarm: Untuk mendapatkan daftar semua alarm, Anda dapat menggunakan perintah AWS CLImenggunakan Amazon CloudWatch. `describe-alarms` Bergantung pada berapa banyak alarm yang telah Anda atur, Anda mungkin harus menggunakan pagination untuk mengambil subset alarm untuk setiap panggilan, atau sebagai alternatif Anda dapat menggunakan alarm AWS SDK untuk mendapatkan alarm menggunakan panggilan. API
2. Dokumentasikan semua tindakan alarm: Perbarui runbook dengan semua alarm dan tindakannya, terlepas dari apakah runbook itu manual atau otomatis. AWS Systems Manager menyediakan runbook yang telah ditentukan sebelumnya. Untuk informasi lebih lanjut tentang runbook, lihat Bekerja dengan runbook. Untuk detail tentang cara melihat konten runbook, silakan lihat Menampilkan konten runbook.
3. Siapkan dan kelola tindakan alarm: Untuk alarm apa pun yang memerlukan tindakan, tentukan tindakan otomatis menggunakan. CloudWatch SDK Misalnya, Anda dapat mengubah status EC2 instans Amazon secara otomatis berdasarkan CloudWatch alarm dengan membuat dan mengaktifkan tindakan pada alarm atau menonaktifkan tindakan pada alarm.

   Anda juga dapat menggunakan Amazon EventBridge untuk merespons peristiwa sistem secara otomatis, seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Anda dapat membuat aturan untuk menunjukkan peristiwa yang sesuai kepentingan Anda, dan tindakan yang akan diambil ketika peristiwa sesuai dengan aturan. Tindakan yang dapat dimulai secara otomatis termasuk menjalankan AWS Lambdafungsi, menjalankan Amazon, menyampaikan acara ke

EC2Run CommandAmazon Kinesis Data Streams, dan melihat Automate Amazon menggunakan. EC2 EventBridge

4. Prosedur Operasi Standar (SOPs): Berdasarkan komponen aplikasi Anda, AWS Resilience Hubrekomendasikan beberapa SOPtemplat. Anda dapat menggunakan ini SOPs untuk mendokumentasikan semua proses yang harus diikuti operator jika ada peringatan. Anda juga dapat membuat rekomendasi Resilience Hub SOP berdasarkan, di mana Anda memerlukan aplikasi Resilience Hub dengan kebijakan ketahanan terkait, serta penilaian ketahanan historis terhadap aplikasi tersebut. Rekomendasi untuk SOP Anda dihasilkan oleh penilaian ketahanan.

   Resilience Hub bekerja dengan Systems Manager untuk mengotomatiskan langkah-langkah Anda SOPs dengan menyediakan sejumlah SSMdokumen yang dapat Anda gunakan sebagai dasar untuk itu. SOPs Misalnya, Resilience Hub dapat merekomendasikan SOP untuk menambahkan ruang disk berdasarkan dokumen SSM otomatisasi yang ada.
5. Lakukan tindakan otomatis menggunakan Amazon DevOps Guru: Anda dapat menggunakan Amazon DevOps Guru untuk secara otomatis memantau sumber daya aplikasi untuk perilaku anomali dan memberikan rekomendasi yang ditargetkan untuk mempercepat identifikasi masalah dan waktu perbaikan. Dengan DevOps Guru, Anda dapat memantau aliran data operasional dalam waktu dekat dari berbagai sumber termasuk CloudWatch metrik Amazon,, AWS ConfigAWS CloudFormation, dan. AWS X-Ray Anda juga dapat menggunakan DevOps Guru untuk secara otomatis membuat OpsItems OpsCenter dan mengirim acara EventBridge untuk otomatisasi tambahan.

Sumber daya

Praktik-praktik terbaik terkait:

- REL06-BP01 Memantau semua komponen untuk beban kerja (Generasi)
- REL06-BP02 Tentukan dan hitung metrik (Agregasi)
- REL06-BP03 Kirim pemberitahuan (Pemrosesan waktu nyata dan mengkhawatirkan)
- REL08-BP01 Gunakan runbook untuk aktivitas standar seperti penerapan

Dokumen terkait:

- AWS Systems Manager Otomatisasi
- Membuat EventBridge Aturan yang Memicu Peristiwa dari Sumber Daya AWS
- Lokakarya Satu Observabilitas

- Amazon Builders' Library: Instrumentasi sistem terdistribusi untuk visibilitas operasional
- Apa itu Amazon DevOps Guru?
- Bekerja dengan Dokumen Otomatisasi (Playbooks)

Video terkait:

- AWS re:invent 2022 - Praktik terbaik observabilitas di Amazon
- AWS Re:invent 2020: Otomatiskan apa pun dengan AWS Systems Manager
- Pengantar AWS Resilience Hub
- Buat Sistem Tiket Kustom untuk Pemberitahuan Amazon DevOps Guru
- Aktifkan Agregasi Wawasan Multi-Akun dengan Amazon Guru DevOps

Contoh terkait:

- Lokakarya Keandalan
- Workshop Amazon CloudWatch dan Systems Manager

REL06-BP05 Menganalisis log

Kumpulkan riwayat metrik dan file log dan analisis ini untuk mendapatkan wawasan beban kerja dan tren lebih luas.

Amazon CloudWatch Logs Insights mendukung bahasa kueri sederhana namun kuat yang dapat Anda gunakan untuk menganalisis data log. Amazon CloudWatch Logs juga mendukung langganan yang memungkinkan data mengalir dengan mulus ke Amazon S3 tempat Anda dapat menggunakan atau Amazon Athena untuk menanyakan data. Amazon CloudWatch Logs juga mendukung pembuatan kueri dengan berbagai macam format. Lihat Format yang Didukung SerDes dan Data di Panduan Pengguna Amazon Athena untuk informasi selengkapnya. Untuk analisis kumpulan file log besar, Anda dapat menjalankan EMR klaster Amazon untuk menjalankan analisis skala petabyte.

Ada sejumlah alat yang disediakan oleh AWS Mitra dan pihak ketiga yang memungkinkan agregasi, pemrosesan, penyimpanan, dan analitik. Alat-alat ini termasuk New Relic, Splunk, Loggly, Logstash, dan Nagios. CloudHealth Namun demikian, log generasi di luar sistem dan log aplikasi bersifat unik untuk setiap penyedia cloud, dan sering kali bersifat unik untuk masing-masing layanan.

Bagian proses pemantauan yang sering kali tidak diperhatikan adalah manajemen data. Anda harus menentukan persyaratan-persyaratan retensi untuk memantau data, kemudian Anda harus

menerapkan kebijakan siklus hidup yang sesuai. Amazon S3 mendukung manajemen siklus hidup pada tingkat bucket S3. Manajemen siklus hidup ini dapat diterapkan secara berbeda ke jalur-jalur yang berbeda yang ada dalam bucket tersebut. Menjelang akhir siklus hidup, Anda dapat melakukan transisi data ke Amazon S3 Glacier untuk mendapatkan penyimpanan jangka panjang, dan kemudian akan menjadi kedaluwarsa setelah akhir jangka waktu retensi tercapai. Kelas penyimpanan S3 Intelligent-Tiering didesain untuk mengoptimalkan biaya dengan secara otomatis memindahkan data ke tingkat akses yang paling hemat biaya, tanpa memengaruhi performa atau menimbulkan biaya operasional tambahan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- CloudWatch Logs Insights memungkinkan Anda untuk secara interaktif mencari dan menganalisis data log Anda di Amazon CloudWatch Logs.
  - Menganalisis Data Log dengan Wawasan CloudWatch Log
  - Kueri Contoh Wawasan CloudWatch Log Amazon
- Gunakan CloudWatch Log Amazon untuk mengirim log ke Amazon S3 tempat Anda dapat menggunakan atau Amazon Athena untuk menanyakan data.
  - Bagaimana cara menganalisis log akses server Amazon S3 menggunakan Athena?
    - Buat kebijakan siklus hidup S3 untuk bucket log akses server Anda. Konfigurasikan kebijakan siklus hidup untuk secara berkala menghapus file log. Hal ini mengurangi jumlah data yang dianalisis oleh Athena untuk setiap kueri.
      - Bagaimana Cara Membuat Kebijakan Siklus Hidup untuk Bucket S3?

Sumber daya

Dokumen terkait:

- Kueri Contoh Wawasan CloudWatch Log Amazon
- Menganalisis Data Log dengan Wawasan CloudWatch Log
- Debugging dengan Amazon CloudWatch Synthetics dan AWS X-Ray
- Bagaimana Cara Membuat Kebijakan Siklus Hidup untuk Bucket S3?
- Bagaimana cara menganalisis log akses server Amazon S3 menggunakan Athena?
- Lokakarya Satu Observabilitas
- Amazon Builders' Library: Instrumentasi sistem terdistribusi untuk visibilitas operasional

REL06-BP06 Melakukan tinjauan secara teratur

Sering kali tinjau bagaimana pemantauan beban kerja diimplementasikan dan perbarui berdasarkan perubahan dan peristiwa yang signifikan.

Pemantauan yang efektif didorong oleh metrik-metrik bisnis utama. Pastikan metrik-metrik ini diakomodasi di beban kerja Anda seiring dengan berubahnya prioritas bisnis.

Mengaudit pemantauan Anda akan membantu memastikan Anda tahu kapan sebuah aplikasi memenuhi tujuan-tujuan ketersediaannya. Analisis akar masalah (RCA) memerlukan kemampuan untuk menemukan apa yang telah terjadi ketika ada kegagalan. AWS memberikan layanan-layanan yang memungkinkan Anda untuk melacak keadaan layanan Anda saat terjadi sebuah insiden:

- Amazon CloudWatch Logs: Anda dapat menyimpan log Anda di layanan ini dan memeriksa isinya.
- Amazon CloudWatch Logs Insights: Adalah layanan yang dikelola sepenuhnya yang memungkinkan Anda menganalisis log besar dalam hitungan detik. Layanan ini memberikan visualisasi dan kueri cepat dan interaktif kepada Anda.
- AWS Config: Anda dapat melihat infrastruktur AWS apa yang digunakan di berbagai titik waktu yang berbeda.
- AWS CloudTrail: Anda dapat melihat mana yang AWS APIs dipanggil pada jam berapa dan oleh prinsipal apa.

Di AWS, kami mengadakan pertemuan mingguan untuk meninjau kinerja operasional dan untuk berbagi pembelajaran antar tim. Karena ada begitu banyak tim AWS, kami membuat The Wheel untuk memilih beban kerja secara acak untuk ditinjau. Menetapkan irama yang teratur untuk peninjauan performa operasional dan berbagi pengetahuan akan meningkatkan kemampuan Anda untuk mencapai performa tim operasional yang lebih tinggi.

Anti-pola umum:

- Hanya mengumpulkan metrik default.
- Menetapkan strategi pemantauan dan tidak pernah meninjaunya.
- Tidak membahas pemantauan ketika ada deployment perubahan besar.

Manfaat menerapkan praktik terbaik ini: Melakukan peninjauan secara teratur terhadap pemantauan Anda akan membuat Anda mampu melakukan antisipasi terhadap potensi masalah yang mungkin

terjadi, dan bukannya bereaksi terhadap notifikasi ketika masalah yang diantisipasi sesungguhnya terjadi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Buat beberapa dasbor untuk beban kerja. Anda harus memiliki sebuah dasbor tingkat teratas yang berisi metrik-metrik bisnis utama, serta metrik-metrik teknis yang telah Anda identifikasi sebagai metrik paling relevan untuk kondisi beban kerja yang diproyeksikan sesuai dengan penggunaan yang bervariasi. Anda juga harus memiliki dasbor yang dapat diinspeksi untuk berbagai tingkat aplikasi dan dependensi.
  - Menggunakan CloudWatch Dasbor Amazon
- Jadwalkan dan lakukan peninjauan dasbor beban kerja secara teratur. Lakukan inspeksi dasbor secara rutin. Anda mungkin memiliki irama yang berbeda untuk kedalaman inspeksi Anda.
  - Inspeksi apakah ada kecenderungan dalam metrik-metrik tersebut. Bandingkan nilai-nilai metrik dengan nilai-nilai historis untuk melihat apakah ada tren yang mungkin mengindikasikan bahwa sesuatu perlu diselidiki. Contohnya antara lain: meningkatkan latensi, menurunkan fungsi bisnis utama, dan meningkatkan respons-respons kegagalan.
  - Inspeksi apakah ada penyimpangan/anomali dalam metrik-metrik Anda. Rerata atau median dapat menutupi penyimpangan (outlier) dan anomali. Lihat nilai tertinggi dan nilai terendah dalam kerangka waktu dan lakukan investigasi untuk mengetahui penyebab skor yang ekstrem. Saat Anda melanjutkan untuk menghilangkan penyebab-penyebab ini, hal itu akan menurunkan definisi ekstrem Anda akan memungkinkan Anda untuk terus meningkatkan konsistensi performa beban kerja Anda.
  - Cari perubahan-perubahan mendadak dalam perilaku. Perubahan cepat yang terjadi dalam jumlah atau arah metrik dapat menandakan telah ada perubahan dalam aplikasi, atau ada faktor eksternal yang mungkin perlu Anda tambahi dengan metrik-metrik tambahan untuk dilacak.

Sumber daya

Dokumen terkait:

- Kueri Contoh Wawasan CloudWatch Log Amazon
- Debugging dengan Amazon CloudWatch Synthetics dan AWS X-Ray
- Lokakarya Satu Observabilitas
- Amazon Builders' Library: Instrumentasi sistem terdistribusi untuk visibilitas operasional

- Menggunakan CloudWatch Dasbor Amazon

REL06-BP07 Memantau end-to-end penelusuran permintaan melalui sistem Anda

Lacak permintaan yang sedang diproses melalui komponen layanan agar tim produk dapat lebih mudah menganalisis dan menemukan serta memperbaiki masalah dan meningkatkan kinerja.

Hasil yang diinginkan: Beban kerja dengan penelusuran komprehensif di semua komponen mudah di-debug, meningkatkan waktu rata-rata untuk resolusi (MTTR) kesalahan dan latensi dengan menyederhanakan penemuan akar penyebab. End-to-endpenelusuran mengurangi waktu yang diperlukan untuk menemukan komponen yang terkena dampak dan menelusuri akar penyebab kesalahan atau latensi yang terperinci.

Anti-pola umum:

- Penelusuran digunakan untuk beberapa komponen, tidak semuanya. Misalnya, tanpa melacak AWS Lambda, tim mungkin tidak memahami dengan jelas latensi yang disebabkan oleh start dingin dalam beban kerja yang runcing.
- Kenari sintetis atau pemantauan pengguna nyata (RUM) tidak dikonfigurasi dengan penelusuran. Tanpa kenari atauRUM, telemetri interaksi klien dihilangkan dari analisis jejak yang menghasilkan profil kinerja yang tidak lengkap.
- Beban kerja hibrida mencakup alat-alat penelusuran cloud-native dan pihak ketiga, tetapi langkah-langkah belum dilakukan untuk memilih dan sepenuhnya mengintegrasikan solusi penelusuran tunggal. Berdasarkan solusi penelusuran yang dipilih, penelusuran asli cloud SDKs harus digunakan untuk komponen instrumen yang bukan cloud native atau alat pihak ketiga harus dikonfigurasi untuk menelan telemetri jejak asli cloud.

Manfaat menerapkan praktik terbaik ini: Saat tim pengembangan menerima peringatan masalah, mereka dapat melihat gambaran utuh dari interaksi komponen sistem, termasuk korelasi komponen per komponen dengan pembuatan log, kinerja, dan kegagalan. Karena penelusuran memudahkan Anda untuk mengidentifikasi akar masalah secara visual, waktu penyelidikan akar masalah akan menjadi lebih singkat. Tim yang memahami interaksi komponen secara detail dapat mengambil keputusan yang lebih baik dan lebih cepat saat menyelesaikan masalah. Keputusan seperti kapan harus menginvokasi failover pemulihan bencana (DR) atau lokasi terbaik untuk menerapkan strategi penyembuhan mandiri dapat ditingkatkan dengan menganalisis jejak sistem, dan pada akhirnya meningkatkan kepuasan pelanggan terhadap layanan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Tim yang mengoperasikan aplikasi-aplikasi yang terdistribusi dapat menggunakan alat penelusuran untuk membuat sebuah pengidentifikasi korelasi, mengumpulkan jejak permintaan, dan membuat peta layanan dari komponen-komponen yang terhubung. Semua komponen aplikasi harus disertakan dalam jejak-jejak permintaan, termasuk klien layanan, gateway perangkat lunak perantara (middleware) dan bus peristiwa, komponen komputasi, dan penyimpanan, termasuk penyimpanan nilai kunci dan basis data. Sertakan kenari sintetis dan pemantauan pengguna nyata dalam konfigurasi end-to-end penelusuran Anda untuk mengukur interaksi dan latensi klien jarak jauh sehingga Anda dapat mengevaluasi kinerja sistem secara akurat terhadap perjanjian dan sasaran tingkat layanan Anda.

Anda dapat menggunakan AWS X-Raydan layanan instrumentasi Pemantauan CloudWatch Aplikasi Amazon untuk memberikan tampilan lengkap permintaan saat mereka melakukan perjalanan melalui aplikasi Anda. X-Ray mengumpulkan telemetri aplikasi dan memungkinkan Anda memvisualisasikan dan memfilternya di seluruh muatan, fungsi, jejak, layananAPIs, dan dapat dihidupkan untuk komponen sistem tanpa kode atau kode rendah. CloudWatch pemantauan aplikasi termasuk ServiceLens untuk mengintegrasikan jejak Anda dengan metrik, log, dan alarm. CloudWatch Pemantauan aplikasi juga mencakup sintetis untuk memantau titik akhir Anda danAPIs, serta pemantauan pengguna nyata untuk instrumen klien aplikasi web Anda.

Langkah-langkah implementasi

- Gunakan AWS X-Ray di semua layanan asli yang didukung seperti Amazon S3, AWS Lambda, dan Amazon API Gateway. AWS Layanan ini memungkinkan X-Ray dengan sakelar konfigurasi menggunakan infrastruktur sebagai kode, AWS SDKs, atau file. AWS Management Console
- Aplikasi instrumen AWS Distro for Open Telemetry dan X-Ray atau agen pengumpulan pihak ketiga.
- Tinjau Panduan Pengembang AWS X-Ray untuk penerapan khusus bahasa pemrograman. Bagian dokumentasi ini merinci cara instrumen HTTP permintaan, SQL kueri, dan proses lain yang spesifik untuk bahasa pemrograman aplikasi Anda.
- Gunakan penelusuran X-Ray untuk Amazon CloudWatch Synthetic Canaries dan Amazon CloudWatch RUM untuk menganalisis jalur permintaan dari klien pengguna akhir Anda melalui infrastruktur hilir AWS Anda.
- Konfigurasikan CloudWatch metrik dan alarm berdasarkan kesehatan sumber daya dan telemetri kenari sehingga tim diberitahu tentang masalah dengan cepat, dan kemudian dapat menyelami jejak dan peta layanan dengan lebih dalam. ServiceLens

- Aktifkan integrasi X-Ray untuk alat-alat penelusuran pihak ketiga seperti Datadog, New Relic, atau Dynatrace jika Anda menggunakan alat-alat pihak ketiga untuk solusi penelusuran utama Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- REL06-BP01 Memantau semua komponen untuk beban kerja (Generasi)
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan

Dokumen terkait:

- Apa itu AWS X-Ray?
- Amazon CloudWatch: Pemantauan Aplikasi
- Debugging dengan Amazon CloudWatch Synthetics dan AWS X-Ray
- Amazon Builders' Library: Instrumentasi sistem terdistribusi untuk visibilitas operasional
- Integrasi AWS X-Ray dengan layanan lain AWS
- AWS Distro untuk OpenTelemetry dan AWS X-Ray
- Amazon CloudWatch: Menggunakan pemantauan sintetis
- Amazon CloudWatch: Gunakan CloudWatch RUM
- Siapkan kenari CloudWatch sintetis Amazon dan alarm Amazon CloudWatch
- Ketersediaan dan Selanjutnya: Memahami dan Meningkatkan Ketahanan Sistem Terdistribusi AWS

Contoh terkait:

- Lokakarya Satu Observabilitas

Video terkait:

- AWS re:invent 2022 - Cara memantau aplikasi di beberapa akun
- Cara Memantau AWS Aplikasi Anda

Alat terkait:

- AWS X-Ray

- Amazon CloudWatch
- Amazon Route 53

## REL7. Bagaimana cara mendesain beban kerja Anda untuk beradaptasi dengan perubahan dalam permintaan?

Beban kerja yang dapat diskalakan memberikan elastisitas untuk menambahkan atau mengeluarkan sumber daya secara otomatis sehingga sangat sesuai dengan permintaan saat ini pada titik waktu tertentu.

Praktik terbaik

- REL07-BP01 Gunakan otomatisasi saat memperoleh atau menskalakan sumber daya
- REL07-BP02 Memperoleh sumber daya setelah mendeteksi gangguan pada beban kerja
- REL07-BP03 Dapatkan sumber daya setelah deteksi bahwa lebih banyak sumber daya diperlukan untuk beban kerja
- REL07-BP04 Beban uji beban kerja Anda

REL07-BP01 Gunakan otomatisasi saat memperoleh atau menskalakan sumber daya

Saat mengganti sumber daya yang rusak atau menskalakan beban kerja Anda, otomatiskan proses dengan menggunakan AWS layanan terkelola, seperti Amazon S3 dan. AWS Auto Scaling Anda juga dapat menggunakan alat pihak ketiga dan AWS SDKs untuk mengotomatiskan penskalaan.

AWS Layanan terkelola termasuk Amazon S3, Amazon CloudFront,, AWS Auto Scaling, Amazon DynamoDB AWS Lambda,, dan Amazon AWS Fargate Route 53.

AWS Auto Scaling memungkinkan Anda mendeteksi dan mengganti instance yang rusak. Ini juga memungkinkan Anda membuat rencana penskalaan untuk sumber daya termasuk EC2 instans Amazon dan Armada Spot, ECS tugas Amazon, tabel dan indeks Amazon DynamoDB, dan Replika Amazon Aurora.

Saat menskalakan EC2 instans, pastikan Anda menggunakan beberapa Availability Zone (sebaiknya setidaknya tiga) dan menambah atau menghapus kapasitas untuk menjaga keseimbangan di seluruh Availability Zone ini. ECStask atau pod Kubernetes (saat menggunakan Amazon Elastic Kubernetes Service) juga harus didistribusikan ke beberapa Availability Zone.

Saat menggunakan AWS Lambda, skala instance secara otomatis. Setiap kali pemberitahuan acara diterima untuk fungsi Anda, AWS Lambda dengan cepat menempatkan kapasitas gratis dalam

armada komputasi dan menjalankan kode Anda hingga konkurensi yang dialokasikan. Anda harus memastikan bahwa konkurensi yang diperlukan telah dikonfigurasikan di Lambda tertentu, dan di dalam Service Quotas Anda.

Amazon S3 secara otomatis menyesuaikan skalanya untuk menangani tingkat permintaan yang tinggi. Misalnya, aplikasi Anda dapat mencapai setidaknya 3.500PUT/COPYPOST/DELETEatau GET HEAD 5.500/permintaan per detik per awalan dalam sebuah bucket. Tidak ada batas jumlah awalan dalam bucket. Anda dapat meningkatkan kinerja membaca atau menulis dengan cara menyelaraskan pembacaan. Misalnya, jika Anda membuat 10 awalan dalam bucket Amazon S3 untuk paralelisasi bacaan, Anda dapat menskalakan kinerja baca Anda menjadi 55.000 permintaan baca per detik.

Konfigurasikan dan gunakan Amazon CloudFront atau jaringan pengiriman konten tepercaya (CDN). A CDN dapat memberikan waktu respons pengguna akhir yang lebih cepat dan dapat melayani permintaan konten dari cache, sehingga mengurangi kebutuhan untuk menskalakan beban kerja Anda.

Anti-pola umum:

- Mengimplementasikan grup Auto Scaling (penskalaan otomatis) untuk pemulihan otomatis, tetapi tidak mengimplementasikan elastisitas.
- Menggunakan penskalaan otomatis untuk merespons peningkatan lalu lintas yang signifikan.
- Melakukan deployment aplikasi yang sangat stateful, yang menghilangkan opsi elastisitas.

Manfaat menerapkan praktik terbaik ini: Otomatisasi menghilangkan potensi kesalahan manual dalam melakukan deployment dan penonaktifan sumber daya. Otomatisasi menghilangkan risiko-risiko pembengkakan biaya dan penolakan layanan akibat lambatnya respons saat dibutuhkan untuk melakukan deployment atau penonaktifan (decommissioning).

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Konfigurasikan dan gunakan AWS Auto Scaling. Layanan ini memantau aplikasi-aplikasi Anda dan secara otomatis menyesuaikan kapasitas untuk menjaga agar aplikasi-aplikasi tersebut tetap mempunyai kinerja yang stabil dan dapat diprediksi dengan biaya serendah mungkin. Dengan menggunakan AWS Auto Scaling, Anda dapat mengonfigurasi penskalaan aplikasi untuk beberapa sumber daya di beberapa layanan.
  - Apa itu AWS Auto Scaling?

    - Konfigurasikan Auto Scaling pada EC2 instans Amazon dan Armada Spot, tugas Amazon, tabel dan indeks ECS Amazon DynamoDB, Replika Amazon Aurora, dan peralatan sebagaimana berlaku. AWS Marketplace
        - Mengelola kapasitas throughput secara otomatis dengan menggunakan Penskalaan Otomatis DynamoDB
            - Gunakan API operasi layanan untuk menentukan alarm, kebijakan penskalaan, waktu pemanasan, dan waktu pendinginan.
- Gunakan Penyeimbangan Beban Elastis. Penyeimbang beban dapat mendistribusikan beban berdasarkan jalur atau berdasarkan konektivitas jaringan.
    - Apa itu Penyeimbangan Beban Elastis?
        - Penyeimbang Beban Aplikasi dapat mendistribusikan beban berdasarkan jalur.
            - Apa itu Penyeimbang Beban Aplikasi?
                - Konfigurasikan Penyeimbang Beban Aplikasi untuk mendistribusikan lalu lintas ke berbagai beban kerja berdasarkan nama domain.
                - Application Load Balancers dapat digunakan untuk mendistribusikan beban dengan cara yang terintegrasi dengan AWS Auto Scaling untuk mengelola permintaan.
                    - Menggunakan penyeimbang beban dengan grup Auto Scaling
        - Penyeimbang Beban Jaringan dapat mendistribusikan beban berdasarkan koneksi.
            - Apa itu Penyeimbang Beban Jaringan?
                - Konfigurasikan Network Load Balancer untuk mendistribusikan lalu lintas ke beban kerja yang berbeda menggunakanTCP, atau untuk memiliki kumpulan alamat IP yang konstan untuk beban kerja Anda.
                - Network Load Balancers dapat digunakan untuk mendistribusikan beban dengan cara yang terintegrasi dengan AWS Auto Scaling untuk mengelola permintaan.
- Gunakan DNS penyedia yang sangat tersedia. DNSnama memungkinkan pengguna Anda memasukkan nama alih-alih alamat IP untuk mengakses beban kerja Anda dan mendistribusikan informasi ini ke ruang lingkup yang ditentukan, biasanya secara global untuk pengguna beban kerja.
    - Gunakan Amazon Route 53 atau DNS penyedia tepercaya.
        - Apa itu Amazon Route 53?
    - Gunakan Route 53 untuk mengelola CloudFront distribusi dan penyeimbang beban Anda.
        - Tentukan domain dan subdomain yang akan Anda kelola.

    - Buat set rekaman yang sesuai menggunakan ALIAS atau CNAME catatan.
        - Bekerja dengan catatan
- Gunakan jaringan AWS global untuk mengoptimalkan jalur dari pengguna Anda ke aplikasi Anda. AWS Global Accelerator terus memantau kesehatan titik akhir aplikasi Anda dan mengarahkan lalu lintas ke titik akhir yang sehat dalam waktu kurang dari 30 detik.
    - AWS Global Accelerator adalah layanan yang meningkatkan ketersediaan dan kinerja aplikasi Anda dengan pengguna lokal atau global. Ini menyediakan alamat IP statis yang bertindak sebagai titik masuk tetap ke titik akhir aplikasi Anda dalam satu atau beberapa Wilayah AWS, seperti Application Load Balancers, Network Load Balancers, atau instans Amazon. EC2
        - Apa itu Akselerator Global AWS ?
- Konfigurasikan dan gunakan Amazon CloudFront atau jaringan pengiriman konten tepercaya (CDN). Jaringan pengiriman konten dapat memberikan waktu respons pengguna akhir yang lebih cepat serta dapat memenuhi permintaan konten yang dapat mengakibatkan penskalaan yang tidak diperlukan terhadap beban kerja Anda.
    - Apa itu Amazon CloudFront?
        - Konfigurasikan CloudFront distribusi Amazon untuk beban kerja Anda, atau gunakan pihak ketiga. CDN
            - Anda dapat membatasi akses ke beban kerja Anda sehingga hanya dapat diakses CloudFront dengan menggunakan rentang IP untuk CloudFront grup keamanan titik akhir atau kebijakan akses Anda.

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu Anda membuat solusi komputasi otomatis
- AWS Auto Scaling: Cara Kerja Penskalaan
- AWS Marketplace: produk yang dapat digunakan dengan penskalaan otomatis
- Mengelola Kapasitas Throughput Secara Otomatis dengan Menggunakan Penskalaan Otomatis DynamoDB
- Menggunakan penyeimbang beban dengan grup Auto Scaling
- Apa itu Akselerator AWS Global?
- Apa itu EC2 Auto Scaling Amazon?
- Apa itu AWS Auto Scaling?

- Apa itu Amazon CloudFront?
- Apa itu Amazon Route 53?
- Apa itu Penyeimbangan Beban Elastis?
- Apa itu Penyeimbang Beban Jaringan?
- Apa itu Penyeimbang Beban Aplikasi?
- Bekerja dengan catatan

REL07-BP02 Memperoleh sumber daya setelah mendeteksi gangguan pada beban kerja

Skalakan sumber daya secara reaktif saat diperlukan jika ketersediaan terganggu, guna memulihkan ketersediaan beban kerja.

Anda terlebih dahulu harus mengonfigurasi pemeriksaan kondisi dan kriteria pada pemeriksaan ini agar memberikan penanda saat ada ketersediaan yang terganggu karena kurangnya sumber daya. Lalu, beri tahu personel yang bersangkutan untuk menskalakan sumber daya secara manual, atau mulai lakukan otomatisasi untuk menskalakannya secara otomatis.

Skala dapat disesuaikan secara manual untuk beban kerja Anda (misalnya, mengubah jumlah EC2 instance dalam grup Auto Scaling, atau memodifikasi throughput tabel DynamoDB melalui atau). AWS Management Console AWS CLI Namun, otomatisasi harus digunakan bila memungkinkan (silakan lihat Gunakan otomatisasi saat memperoleh atau menskalakan sumber daya).

Hasil yang diinginkan: Aktivitas penskalaan (baik secara otomatis atau manual) dimulai untuk memulihkan ketersediaan setelah mendeteksi adanya kegagalan atau pengalaman pelanggan yang mengalami penurunan kualitas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Terapkan observabilitas dan pemantauan di semua komponen yang ada dalam beban kerja Anda untuk memantau pengalaman pelanggan dan mendeteksi terjadinya kegagalan. Tentukan prosedur, manual atau otomatis, yang menskalakan sumber daya yang diperlukan. o Untuk informasi lebih lanjut, lihat REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan.

Langkah-langkah implementasi

- Tentukan prosedur, baik manual ataupun otomatis, yang menskalakan sumber daya yang dibutuhkan.

- Prosedur penskalaan tergantung pada bagaimana rancangan berbagai komponen yang ada dalam beban kerja Anda.
- Prosedur penskalaan juga bisa berbeda-beda, tergantung pada teknologi dasar yang digunakan.
  - Komponen yang menggunakan AWS Auto Scaling dapat menggunakan rencana penskalaan untuk mengonfigurasi serangkaian instruksi untuk menskalakan sumber daya Anda. Jika Anda bekerja dengan AWS CloudFormation atau menambahkan tag ke AWS sumber daya, Anda dapat menyiapkan rencana penskalaan untuk kumpulan sumber daya yang berbeda per aplikasi. Auto Scaling (penskalaan otomatis) memberikan rekomendasi untuk strategi penyekalaan yang disesuaikan dengan setiap sumber daya. Setelah membuat rencana penskalaan, Auto Scaling (penskalaan otomatis) menggabungkan metode penskalaan dinamik dan penskalaan prediktif secara bersama-sama untuk mendukung strategi penskalaan Anda. Untuk detail selengkapnya, silakan lihat Cara kerja rencana penskalaan.
  - EC2Auto Scaling Amazon memverifikasi bahwa Anda memiliki jumlah EC2 instans Amazon yang benar yang tersedia untuk menangani pemuatan aplikasi Anda. Anda membuat koleksi EC2 instance, yang disebut grup Auto Scaling. Anda dapat menentukan jumlah instans minimum dan maksimum di setiap grup Auto Scaling, dan Amazon Auto EC2 Scaling memastikan bahwa grup Anda tidak pernah berada di bawah atau di atas batas ini. Untuk detail selengkapnya, lihat Apa itu EC2 Auto Scaling Amazon?
  - Penskalaan otomatis Amazon DynamoDB menggunakan layanan Penskalaan Otomatis Aplikasi untuk secara dinamis menyesuaikan kapasitas throughput tersedia untuk merespons pola lalu lintas aktual. Ini memungkinkan tabel atau indeks sekunder global meningkatkan kapasitas baca dan tulis yang disediakan untuk menangani peningkatan lalu lintas tiba-tiba, tanpa throttling. Untuk detail lebih lanjut, silakan lihat Mengelola kapasitas throughput secara otomatis dengan menggunakan penskalaan otomatis DynamoDB.

Sumber daya

Praktik-praktik terbaik terkait:

- REL07-BP01 Gunakan otomatisasi saat memperoleh atau menskalakan sumber daya
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan

Dokumen terkait:

- AWS Auto Scaling: Cara Kerja Penskalaan

- Mengelola Kapasitas Throughput Secara Otomatis dengan Menggunakan Penskalaan Otomatis DynamoDB
- Apa itu EC2 Auto Scaling Amazon?

REL07-BP03 Dapatkan sumber daya setelah deteksi bahwa lebih banyak sumber daya diperlukan untuk beban kerja

Skalakan sumber daya secara proaktif untuk memenuhi permintaan dan menghindari dampak ketersediaan.

Banyak AWS layanan secara otomatis menskalakan untuk memenuhi permintaan. Jika menggunakan EC2 instans Amazon atau ECS kluster Amazon, Anda dapat mengonfigurasi penskalaan otomatis ini agar terjadi berdasarkan metrik penggunaan yang sesuai dengan permintaan beban kerja Anda. Untuk AmazonEC2, CPU pemanfaatan rata-rata, jumlah permintaan penyeimbang beban, atau bandwidth jaringan dapat digunakan untuk skala (atau skala) EC2 instance. Untuk AmazonECS, CPU pemanfaatan rata-rata, jumlah permintaan penyeimbang beban, dan pemanfaatan memori dapat digunakan untuk skala (atau skala) tugas. ECS Dengan menggunakan Target Auto Scaling aktif AWS, autoscaler bertindak seperti termostat rumah tangga, menambah atau menghapus sumber daya untuk mempertahankan nilai target (misalnya, pemanfaatan 70%CPU) yang Anda tentukan.

Amazon EC2 Auto Scaling juga dapat melakukan Predictive Auto Scaling, yang menggunakan pembelajaran mesin untuk menganalisis beban kerja historis setiap sumber daya dan secara teratur memperkirakan beban masa depan.

Hukum Little membantu menghitung berapa banyak contoh komputasi (EC2instance, fungsi Lambda bersamaan, dll.) yang Anda butuhkan.

$L = \lambda W$

L = jumlah instans (atau konkurensi nilai tengah dalam sistem)

λ = rasio rata-rata permintaan yang diterima (permintaan/detik)

W = waktu rata-rata yang diperlukan setiap permintaan di dalam sistem (detik)

Misalnya, dengan laju 100 rps (permintaan per detik), jika setiap permintaan memerlukan 0,5 detik untuk diproses, maka Anda akan memerlukan 50 instans untuk memenuhi permintaan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Tambahkan sumber daya setelah terdeteksi bahwa beban kerja memerlukan lebih banyak sumber daya. Skalakan sumber daya secara proaktif untuk memenuhi permintaan dan menghindari dampak ketersediaan.
  - Hitung banyaknya sumber daya komputasi yang akan Anda butuhkan (konkurensi komputasi) untuk menangani rasio permintaan tertentu.
    - Menceritakan Tentang Little's Law
  - Jika Anda memiliki pola historis untuk penggunaan, siapkan penskalaan terjadwal untuk penskalaan EC2 otomatis Amazon.
    - Penskalaan Terjadwal untuk EC2 Auto Scaling Amazon
  - Gunakan AWS penskalaan prediktif.
    - Penskalaan prediktif untuk Amazon EC2 Auto Scaling

Sumber daya

Dokumen terkait:

- AWS Marketplace: produk yang dapat digunakan dengan penskalaan otomatis
- Mengelola Kapasitas Throughput Secara Otomatis dengan Menggunakan Penskalaan Otomatis DynamoDB
- Penskalaan Prediktif untukEC2, Didukung oleh Machine Learning
- Penskalaan Terjadwal untuk EC2 Auto Scaling Amazon
- Menceritakan Tentang Little's Law
- Apa itu EC2 Auto Scaling Amazon?

REL07-BP04 Beban uji beban kerja Anda

Adopsi metodologi pengujian beban untuk mengukur apakah aktivitas penskalaan memenuhi persyaratan beban kerja.

Pengujian beban yang berkelanjutan penting untuk dilakukan. Tes beban harus menemukan titik puncaknya dan menguji kinerja beban kerja Anda. AWS memudahkan untuk mengatur lingkungan pengujian sementara yang memodelkan skala beban kerja produksi Anda. Di cloud, Anda dapat membuat sebuah lingkungan pengujian berskala produksi sesuai permintaan, menyelesaikan

pengujian, dan kemudian menonaktifkan sumber dayanya. Karena Anda hanya membayar lingkungan pengujian saat sedang berjalan, Anda dapat menyimulasikan lingkungan langsung Anda dengan biaya yang lebih murah daripada pengujian on-premise.

Pengujian beban di lingkungan produksi juga harus dipertimbangkan sebagai bagian dari game day di mana sistem produksi diberikan tekanan, selama jam-jam penggunaan pelanggan yang lebih rendah, dengan semua personel yang siap untuk menerjemahkan hasilnya dan menangani masalah-masalah yang muncul.

Anti-pola umum:

- Melakukan pengujian beban di lingkungan deployment yang tidak memiliki konfigurasi yang sama dengan lingkungan produksi Anda.
- Melakukan pengujian beban hanya pada beban kerja Anda secara terpisah-pisah, bukan pada keseluruhan beban kerja.
- Melakukan pengujian beban dengan subset permintaan, bukan set permintaan riil yang representatif.
- Melakukan pengujian beban ke satu faktor keselamatan kecil di atas beban yang diharapkan.

Manfaat menerapkan praktik terbaik ini: Anda mengetahui komponen apa saja di dalam arsitektur Anda yang gagal saat menerima beban, dan mampu mengidentifikasi metrik apa saja yang perlu diamati sebagai indikator bahwa Anda mendekati beban tersebut tepat waktu untuk mengatasi masalah dan mencegah dampak kegagalan tersebut.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Lakukan pengujian beban untuk mengidentifikasi aspek mana dalam beban kerja Anda yang menunjukkan bahwa Anda harus menambah atau menghapus kapasitas. Pengujian beban harus memiliki lalu lintas representatif yang serupa dengan yang Anda terima di lingkungan produksi. Tingkatkan beban sambil mengamati metrik yang telah Anda instrumentasikan untuk menentukan metrik mana yang menunjukkan kapan Anda harus menambah atau menghapus sumber daya.
  - Pengujian Beban Terdistribusi pada AWS: mensimulasikan ribuan pengguna yang terhubung
    - Identifikasi gabungan permintaan. Anda mungkin memiliki gabungan permintaan yang beragam, sehingga Anda harus melihat berbagai kerangka waktu saat mengidentifikasi gabungan lalu lintas tersebut.

- Implementasikan pendorong beban. Anda dapat menggunakan perangkat lunak kode kustom, sumber terbuka, atau komersial untuk mengimplementasikan pendorong beban.
- Lakukan uji beban di awal dengan menggunakan kapasitas kecil. Anda melihat beberapa efek langsung dengan mendorong beban ke kapasitas yang lebih kecil, kemungkinan seukuran satu instans atau kontainer.
- Uji beban dengan kapasitas yang lebih besar. Efek-efek tersebut akan berbeda di beban yang terdistribusi, sehingga Anda harus melakukan pengujian di lingkungan yang semirip mungkin dengan lingkungan produksi.

Sumber daya

Dokumen terkait:

- Pengujian Beban Terdistribusi pada AWS: mensimulasikan ribuan pengguna yang terhubung
- Aplikasi pengujian beban

Video terkait:

- AWS Summit ANZ 2023: Mempercepat dengan percaya diri melalui Pengujian Beban AWS Terdistribusi

## REL8. Bagaimana cara mengimplementasikan perubahan?

Perubahan terkontrol diperlukan untuk melakukan deployment fungsionalitas baru, dan untuk memverifikasi bahwa beban kerja dan lingkungan operasi menjalankan perangkat lunak yang dikenal dan dapat di-patch atau diganti dengan cara yang dapat diprediksi. Jika perubahan-perubahan ini tidak terkontrol, maka akan sulit untuk memprediksi efek dari perubahan-perubahan tersebut, atau untuk mengatasi masalah yang timbul sebagai akibatnya.

Praktik terbaik

- REL08-BP01 Gunakan runbook untuk aktivitas standar seperti penerapan
- REL08-BP02 Integrasikan pengujian fungsional sebagai bagian dari penerapan Anda
- REL08-BP03 Mengintegrasikan pengujian ketahanan sebagai bagian dari penerapan Anda
- REL08-BP04 Terapkan menggunakan infrastruktur yang tidak dapat diubah
- REL08-BP05 Menyebarkan perubahan dengan otomatisasi

REL08-BP01 Gunakan runbook untuk aktivitas standar seperti penerapan

Runbook adalah prosedur terdokumentasi untuk mencapai hasil-hasil tertentu. Gunakan runbook untuk melakukan aktivitas-aktivitas standar, baik yang dilakukan secara manual maupun otomatis. Contohnya termasuk menerapkan beban kerja, menambal beban kerja, atau membuat modifikasi. DNS

Misalnya, letakkan proses di tempat untuk memastikan keamanan rollback selama deployment. Memastikan bahwa Anda dapat membatalkan deployment tanpa menimbulkan gangguan terhadap pelanggan adalah sesuatu yang penting dalam menciptakan keandalan layanan.

Untuk prosedur runbook, mulailah dengan proses manual efektif yang valid, implementasikan dalam kode, dan lakukan invokasi agar berjalan secara otomatis saat diperlukan.

Bahkan untuk beban kerja canggih yang sangat otomatis, runbook masih berguna untuk menjalankan game day atau memenuhi persyaratan pelaporan dan audit yang ketat.

Perlu diperhatikan bahwa playbook digunakan untuk merespons insiden-insiden tertentu, sedangkan runbook digunakan untuk meraih hasil-hasil tertentu. Sering kali, runbook ditujukan untuk aktivitas rutin, sedangkan playbook digunakan untuk merespons peristiwa-peristiwa non-rutin.

Anti-pola umum:

- Melakukan perubahan tidak terencana pada konfigurasi di lingkungan produksi.
- Melewatkan langkah-langkah yang diuraikan dalam rencana Anda untuk melakukan deployment yang lebih cepat, yang mengakibatkan deployment mengalami kegagalan.
- Membuat perubahan tanpa melakukan uji pembatalan perubahan.

Manfaat menerapkan praktik terbaik ini: Perencanaan perubahan yang efektif meningkatkan kemampuan Anda untuk berhasil menjalankan perubahan karena Anda mengetahui semua sistem yang terpengaruh. Validasi perubahan di lingkungan pengujian meningkatkan kepercayaan diri Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Sediakan respons yang cepat dan konsisten terhadap peristiwa yang dipahami dengan baik dengan cara membuat dokumentasi prosedur-prosedur penanganan peristiwa di dalam runbook.
  - Kerangka Kerja AWS Well-Architected: Konsep: Runbook

- Gunakan prinsip infrastruktur sebagai kode untuk menentukan infrastruktur Anda. Dengan menggunakan AWS CloudFormation (atau pihak ketiga tepercaya) untuk menentukan infrastruktur Anda, Anda dapat menggunakan perangkat lunak kontrol versi untuk versi dan melacak perubahan.
  - Gunakan AWS CloudFormation (atau penyedia pihak ketiga tepercaya) untuk menentukan infrastruktur Anda.
    - Apa itu AWS CloudFormation?
  - Buat templat yang bersifat tunggal dan terpisah-pisah, dengan menggunakan prinsip desain perangkat lunak yang baik.
    - Tentukan izin, templat, dan pihak-pihak yang bertanggung jawab untuk implementasi.
      - Mengontrol akses dengan AWS Identity and Access Management
    - Gunakan kontrol sumber, seperti AWS CodeCommit atau alat pihak ketiga tepercaya, untuk kontrol versi.
      - Apa itu AWS CodeCommit?

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu Anda membuat solusi penerapan otomatis
- AWS Marketplace: produk yang dapat digunakan untuk mengotomatisasi deployment Anda
- AWS Kerangka Well-Architected: Konsep: Runbook
- Apa itu AWS CloudFormation?
- Apa itu AWS CodeCommit?

Contoh terkait:

- Melakukan otomatisasi operasi dengan Playbook dan Runbook

REL08-BP02 Integrasikan pengujian fungsional sebagai bagian dari penerapan Anda

Uji fungsional dijalankan sebagai bagian dari deployment otomatis. Jika kriteria keberhasilan tidak terpenuhi, maka alur akan dihentikan atau dikembalikan (rollback). Pengujian ini dijalankan dalam lingkungan pra-produksi, yang dilaksanakan sebelum perkembangan produksi (pipeline). Idealnya, ini dilakukan sebagai bagian dari sebuah pipeline deployment.

Hasil yang diinginkan: Anda menggunakan otomatisasi untuk melakukan pengujian fungsional, dan data pengujian terkait mengurangi durasi dan biaya pengujian serta meningkatkan akurasi dari hasil pengujian yang didapatkan. Anda mengintegrasikan pengujian fungsional sebagai bagian dari proses deployment Anda, yang membantu Anda untuk mengotomatiskan pipeline rilis agar pembaruan aplikasi dan infrastruktur menjadi cepat dan andal.

Anti-pola umum:

- Anda melakukan pengujian secara manual di luar pipeline deployment.
- Anda melewatkan langkah-langkah pengujian dalam otomatisasi Anda melalui alur kerja darurat manual.
- Anda tidak mengikuti rencana dan proses pengujian yang telah ditetapkan demi mempercepat kronologi (timelines).

Manfaat menerapkan praktik terbaik ini: Tes fungsional akan memvalidasi bahwa sistem beroperasi sesuai dengan persyaratan yang ditentukan. Ini digunakan untuk secara konsisten memverifikasi urutan kerja komponen yang dimaksudkan seperti antarmuka pengguna, databaseAPIs, dan kode sumber. Ketika Anda memeriksa komponen-komponen sistem ini, pengujian fungsional memverifikasi bahwa setiap fitur berperilaku seperti yang diharapkan, yang melindungi harapan pengguna sekaligus integritas perangkat lunak. Integrasikan pengujian fungsional sebagai bagian dari deployment rutin Anda, dan gunakan otomatisasi untuk melakukan deployment semua perubahan, yang akan mengurangi potensi kesalahan manusia.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Integrasikan pengujian fungsional sebagai bagian dari deployment Anda. Uji fungsional dijalankan sebagai bagian dari deployment otomatis. Jika kriteria keberhasilan tidak terpenuhi, pipa dihentikan atau digulung kembali. AWS CodePipeline menyediakan saluran pengiriman berkelanjutan untuk pengujian otomatis, yang memungkinkan penguji untuk mengotomatiskan seluruh proses pengujian dan penerapan. Ini terintegrasi dengan AWS layanan seperti AWS CodeBuild dan AWS CodeDeploy untuk mengotomatiskan fase build, test, dan deployment dari siklus hidup pengembangan perangkat lunak.

Langkah-langkah implementasi

- Konfigurasikan pipeline Anda: Siapkan tahapan sumber, build, uji, dan deploy menggunakan AWS CodePipeline konsol atau AWS Command Line Interface (CLI).

- Tentukan sumber Anda: Dengan AWS CodePipeline, Anda dapat secara otomatis mengambil kode sumber dari sistem kontrol versi seperti GitHub, AWS CodeCommit, atau Bitbucket, yang memverifikasi bahwa kode terbaru selalu digunakan untuk pengujian.
- Mengotomatiskan build dan pengujian: AWS CodeBuild dapat secara otomatis membangun dan menguji kode Anda dan menghasilkan laporan pengujian. Ini mendukung kerangka pengujian populer sepertiJUnit,NUnit, dan TestNG.
- Menerapkan kode Anda: Setelah kode dibuat dan diuji, AWS CodeDeploy dapat menerapkannya ke lingkungan pengujian Anda, termasuk EC2 instance Amazon, AWS Lambda fungsi, atau server lokal.
- Memantau pipeline: AWS CodePipeline dapat melacak kemajuan pipeline Anda dan status dari masing-masing tahap. Anda juga dapat menggunakan pemeriksaan kualitas untuk memblokir pipeline berdasarkan status pelaksanaan pengujian. Anda juga dapat menerima notifikasi untuk setiap kegagalan tahap pipeline atau penyelesaian pipeline.

Sumber daya

Dokumen terkait:

- Gunakan AWS CodePipeline dengan AWS CodeBuild untuk menguji kode dan menjalankan build
- Penebangan dan pemantauan di AWS CodeBuild
- Indikator untuk pengujian fungsional

REL08-BP03 Mengintegrasikan pengujian ketahanan sebagai bagian dari penerapan Anda

Integrasikan pengujian ketahanan dengan memasukkan kegagalan secara sadar ke dalam sistem Anda guna mengukur kemampuannya apabila terjadi skenario yang mengganggu. Pengujian ketahanan berbeda dari pengujian unit dan fungsi yang biasanya terintegrasi dalam siklus deployment, karena pengujian ini berfokus pada identifikasi terhadap kegagalan-kegagalan yang tidak terduga di dalam sistem Anda. Meskipun memulai dengan integrasi pengujian ketahanan dalam tahap praproduksi aman dilakukan, tetapkan tujuan untuk mengimplementasikan pengujian ini dalam produksi sebagai bagian dari game day Anda.

Hasil yang diinginkan: Pengujian ketahanan akan membantu Anda membangun kepercayaan pada kemampuan sistem untuk bertahan dari degradasi dalam produksi. Eksperimen mengidentifikasi titik lemah yang dapat menyebabkan kegagalan, yang akan membantu Anda meningkatkan kualitas

sistem Anda untuk mengurangi terjadinya kegagalan dan penurunan kualitas secara otomatis dan efisien.

Anti-pola umum:

- Kurangnya observabilitas dan pemantauan dalam proses deployment
- Ketergantungan pada manusia untuk mengatasi kegagalan sistem
- Mekanisme analisis kualitas yang buruk
- Fokus pada masalah yang diketahui dalam suatu sistem dan kurangnya eksperimen untuk mengidentifikasi masalah yang belum diketahui
- Identifikasi kegagalan yang tidak mempunyai penyelesaian
- Tidak ada dokumentasi temuan dan runbook

Manfaat menerapkan praktik terbaik: Pengujian ketahanan yang terintegrasi dalam deployment Anda akan membantu Anda dalam mengidentifikasi masalah yang tidak diketahui yang terjadi dalam sistem yang tidak diketahui, yang dapat menyebabkan waktu henti dalam produksi. Melakukan identifikasi terhadap masalah-masalah yang tidak diketahui di dalam sistem akan membantu Anda mendokumentasikan temuan, mengintegrasikan pengujian ke dalam proses CI/CD Anda, dan membangun runbook, yang pada akhirnya akan menyederhanakan mitigasi melalui mekanisme yang efisien dan dapat diulang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Bentuk pengujian ketahanan yang paling umum yang dapat diintegrasikan dalam deployment sistem Anda adalah pemulihan bencana dan rekayasa kekacauan (chaos engineering).

- Sertakan pembaruan pada rencana pemulihan bencana dan prosedur operasi standar (SOPs) dengan penerapan yang signifikan.
- Integrasikan pengujian keandalan ke dalam pipeline deployment otomatis Anda. Layanan-layanan seperti AWS Resilience Hub dapat diintegrasikan ke dalam pipeline CI/CD Anda untuk membuat penilaian ketahanan berkelanjutan yang secara otomatis dievaluasi sebagai bagian dari setiap deployment.
- Tentukan aplikasi Anda di AWS Resilience Hub. Penilaian ketahanan menghasilkan cuplikan kode yang membantu Anda membuat prosedur pemulihan sebagai dokumen AWS Systems Manager

untuk aplikasi Anda dan memberikan daftar monitor dan alarm Amazon yang direkomendasikan. CloudWatch

- Setelah DR Anda merencanakan dan SOPs diperbarui, selesaikan pengujian pemulihan bencana untuk memverifikasi bahwa mereka efektif. Pengujian pemulihan bencana akan membantu Anda untuk menentukan apakah Anda dapat memulihkan sistem setelah terjadinya peristiwa tertentu dan kembali ke beroperasi secara normal. Anda dapat membuat simulasi dari berbagai strategi pemulihan bencana dan mengidentifikasi apakah perencanaan Anda sudah memadai untuk memenuhi persyaratan-persyaratan waktu aktif Anda. Strategi pemulihan bencana yang biasanya dibuat mencakup pencadangan dan pemulihan, pilot light, cold standby, warm standby, hot standby, dan active-active, dan semuanya memerlukan biaya dan memiliki kompleksitas yang berbeda-beda. Sebelum pengujian pemulihan bencana, kami menyarankan Anda menentukan tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO) untuk menyederhanakan pilihan strategi untuk disimulasikan. AWS menawarkan alat pemulihan bencana seperti AWS Elastic Disaster Recoveryuntuk membantu Anda memulai dengan perencanaan dan pengujian Anda.
- Eksperimen chaos engineering memasukkan gangguan (disruptions) ke dalam sistem, seperti pemadaman jaringan dan kegagalan layanan. Dengan melakukan simulasi dengan kegagalan terkontrol, Anda akan dapat menemukan kerentanan sistem Anda sambil mengendalikan dampak-dampak yang ditimbulkan oleh kegagalan yang dimasukkan. Sama seperti strategi lainnya, jalankan simulasi kegagalan terkontrol di lingkungan non-produksi dengan menggunakan layanan-layanan seperti AWS Fault Injection Service untuk mendapatkan kepercayaan diri sebelum melakukan deployment di lingkungan produksi.

Sumber daya

Dokumen terkait:

- Bereksperimen dengan kegagalan menggunakan pengujian ketahanan untuk membangun kesiapan pemulihan
- Terus menilai ketahanan aplikasi dengan dan AWS Resilience HubAWS CodePipeline
- Arsitektur pemulihan bencana (DR) pada AWS, bagian 1: Strategi untuk pemulihan di cloud
- Verifikasi ketahanan beban kerja Anda dengan menggunakan Chaos Engineering
- Prinsip-prinsip Chaos Engineering
- Lokakarya Chaos Engineering

Video terkait:

- AWS re:Invent 2020: Menguji Ketahanan menggunakan Perekayasaan Chaos
- Meningkatkan Ketahanan Aplikasi dengan Layanan Injeksi AWS Kesalahan
- Mempersiapkan & Melindungi Aplikasi Anda Dari Gangguan AWS Resilience Hub

REL08-BP04 Terapkan menggunakan infrastruktur yang tidak dapat diubah

Infrastruktur tetap adalah model yang menuntut bahwa tidak ada pembaruan, patch keamanan, atau perubahan konfigurasi yang terjadi di tempat pada beban kerja produksi. Saat perubahan diperlukan, arsitektur dibangun ke infrastruktur baru dan di-deploy ke dalam lingkungan produksi.

Ikuti strategi penerapan infrastruktur tetap untuk meningkatkan keandalan, konsistensi, dan keterulangan (reproducibility) dalam deployment beban kerja Anda.

Hasil yang diinginkan: Dengan infrastruktur yang tidak dapat diubah, tidak ada modifikasi di tempat yang diizinkan untuk menjalankan sumber daya infrastruktur dalam sebuah beban kerja. Sebaliknya, ketika ada sebuah perubahan yang diperlukan, kumpulan sumber daya infrastruktur baru yang sudah diperbarui, yang berisi semua perubahan yang diperlukan, di-deploy secara paralel dengan sumber daya Anda yang ada. Deployment ini divalidasi secara otomatis, dan jika berhasil, lalu lintas akan dialihkan secara bertahap ke kumpulan sumber daya baru.

Strategi deployment ini berlaku di antaranya untuk pembaruan perangkat lunak, patch keamanan, perubahan infrastruktur, pembaruan konfigurasi, dan pembaruan aplikasi.

Anti-pola umum:

- Menerapkan perubahan di tempat untuk menjalankan sumber daya infrastruktur.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan konsistensi di seluruh lingkungan: Karena tidak ada perbedaan dalam sumber daya infrastruktur di seluruh lingkungan, maka konsistensi ditingkatkan dan pengujian disederhanakan.
- Mengurangi penyimpangan konfigurasi: Dengan mengganti sumber daya infrastruktur dengan konfigurasi yang diketahui dan dikontrol versi, infrastruktur tersebut diatur ke status yang diketahui, diuji, dan tepercaya, menghindari penyimpangan konfigurasi.
- Deployment atom yang andal: Deployment berhasil diselesaikan atau tidak ada yang berubah, meningkatkan konsistensi dan keandalan dalam proses deployment.
- Deployment disederhanakan: Deployment disederhanakan karena tidak memerlukan pembaruan dukungan. Pembaruan hanyalah berupa deployment baru.

- Deployment yang lebih aman dengan proses rollback dan pemulihan yang cepat: Deployment lebih aman karena versi sebelumnya yang digunakan tidak diubah. Anda dapat melakukan rollback jika ada kesalahan yang terdeteksi.
- Postur keamanan yang ditingkatkan: Dengan tidak mengizinkan perubahan infrastruktur, mekanisme akses jarak jauh (sepertiSSH) dapat dinonaktifkan. Hal ini akan mengurangi vektor serangan, sehingga meningkatkan postur keamanan organisasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Otomatisasi

Saat menentukan strategi deployment infrastruktur yang tidak dapat diubah, Anda sebaiknya menggunakan otomatisasi sebanyak mungkin untuk meningkatkan reproduksibilitas dan meminimalkan potensi kesalahan manusia. Untuk detail selengkapnya, lihat REL08-BP05 Menerapkan perubahan dengan otomatisasi dan Mengotomatiskan penerapan hands-off yang aman.

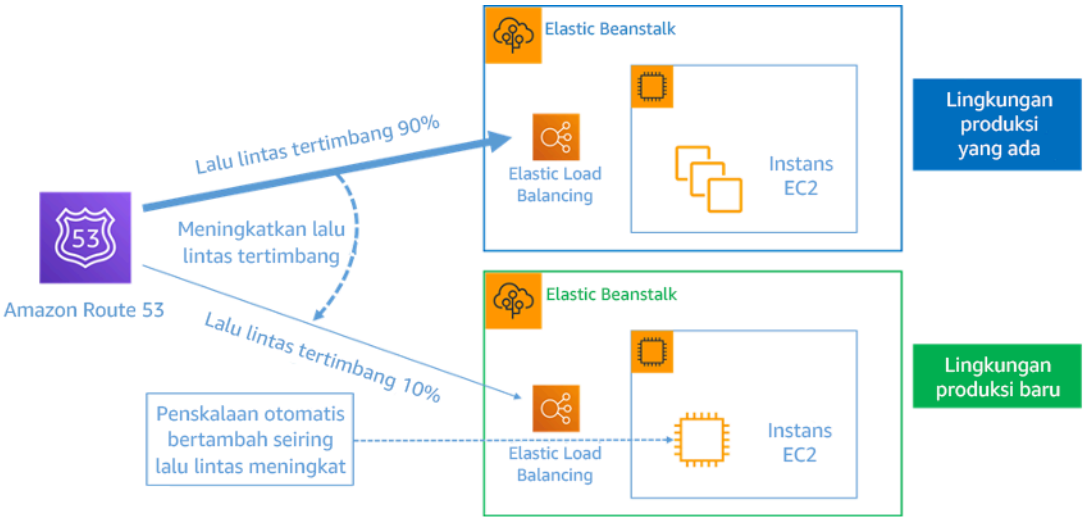
Dengan Infrastruktur sebagai Kode (IaC), langkah-langkah penyediaan infrastruktur, orkestrasi, dan deployment ditentukan dengan cara terprogram, deskriptif, dan deklaratif dan disimpan dalam sistem kontrol sumber. Memanfaatkan infrastruktur sebagai kode akan makin memudahkan otomatisasi deployment infrastruktur dan membantu Anda mewujudkan ketetapan (immutability) infrastruktur.

Pola deployment

Ketika ada perubahan dalam beban kerja yang diperlukan, strategi deployment tetap mengharuskan deployment sumber daya infrastruktur yang baru, termasuk semua perubahan yang diperlukan. Kumpulan sumber daya baru ini harus mengikuti pola rollout yang meminimalkan dampak terhadap pengguna. Ada dua strategi utama untuk deployment ini:

Deployment canary: Praktik ini mengarahkan sejumlah kecil pelanggan kepada versi baru, yang biasanya dijalankan di sebuah instans layanan tunggal (canary). Lalu, Anda meneliti secara mendalam setiap perubahan perilaku atau kesalahan yang dihasilkan. Anda dapat menghapus lalu lintas dari canary jika menemui masalah-masalah kritis dan mengembalikan pengguna ke versi sebelumnya. Jika penerapan berhasil, Anda dapat terus menerapkan pada kecepatan yang Anda inginkan, sambil memantau perubahan untuk kesalahan, hingga Anda sepenuhnya digunakan. AWS CodeDeploy dapat dikonfigurasi dengan konfigurasi penerapan yang memungkinkan penerapan kenari.

Deployment blue/green: Deployment yang bersifat serupa dengan deployment canary, kecuali armada penuh aplikasi di-deploy secara paralel. Anda mengubah deployment Anda di dua tumpukan (blue and green). Sekali lagi, Anda mengirimkan lalu lintas ke versi yang baru, dan kembali ke versi lama jika Anda melihat ada masalah yang terjadi dengan deployment. Umumnya semua lalu lintas diaktifkan sekaligus, namun Anda juga dapat menggunakan sebagian kecil lalu lintas Anda ke setiap versi untuk mengaktifkan adopsi versi baru menggunakan DNS kemampuan perutean tertimbang Amazon Route 53. AWS CodeDeploy dan AWS Elastic Beanstalkdapat dikonfigurasi dengan konfigurasi penerapan yang memungkinkan penerapan biru/hijau.




Gambar 8: Deployment blue/green dengan AWS Elastic Beanstalk dan Amazon Route 53

Deteksi penyimpangan

Penyimpangan didefinisikan sebagai setiap perubahan yang menyebabkan sumber daya infrastruktur memiliki status atau konfigurasi yang berbeda dengan apa yang diharapkan. Setiap jenis perubahan konfigurasi yang tidak terkelola bertentangan dengan gagasan infrastruktur tetap dan tidak bisa diubah (immutable), dan harus dideteksi dan diperbaiki agar infrastruktur tetap berhasil diimplementasikan.

Langkah-langkah implementasi

- Larang modifikasi di tempat pada sumber daya infrastruktur yang sedang berjalan.
  - Anda dapat menggunakan AWS Identity and Access Management (IAM) untuk menentukan siapa atau apa yang dapat mengakses layanan dan sumber daya AWS, mengelola izin berbutir halus secara terpusat, dan menganalisis akses untuk menyempurnakan izin di seluruh. AWS
- Lakukan otomatisasi terhadap deployment sumber daya infrastruktur untuk meningkatkan reproduksibilitas dan meminimalkan terjadinya potensi kesalahan manusia.

- Seperti yang dijelaskan dalam Introduction to DevOps on AWS whitepaper, otomatisasi adalah landasan dengan AWS layanan dan didukung secara internal di semua layanan, fitur, dan penawaran.
- Memanggang Amazon Machine Image (AMI) Anda dapat mempercepat waktu untuk meluncurkannya. EC2Image Builder adalah AWS layanan terkelola penuh yang membantu Anda mengotomatiskan pembuatan, pemeliharaan, validasi, berbagi, dan penyebaran kustom up-to-date Linux atau Windows yang disesuaikan, aman, dan. AMI
- Beberapa layanan yang mendukung otomatisasi adalah:
  - AWS Elastic Beanstalkadalah layanan untuk menyebarkan dan menskalakan aplikasi web dengan cepat yang dikembangkan dengan Java,. NET,PHP, Node.js, Python, Ruby, Go, dan Docker di server yang sudah dikenal seperti Apache,, Passenger, NGINX dan. IIS
  - AWS Protonmembantu tim platform menghubungkan dan mengoordinasikan semua alat berbeda yang dibutuhkan tim pengembangan Anda untuk penyediaan infrastruktur, penerapan kode, pemantauan, dan pembaruan. AWS Proton memungkinkan infrastruktur otomatis sebagai penyediaan kode dan penyebaran aplikasi tanpa server dan berbasis kontainer.
- Memanfaatkan infrastruktur sebagai kode memudahkan untuk mengotomatisasi penyebaran infrastruktur, dan membantu mencapai kekekalan infrastruktur. AWS menyediakan layanan yang memungkinkan pembuatan, penyebaran, dan pemeliharaan infrastruktur dengan cara terprogram, deskriptif, dan deklaratif.
  - AWS CloudFormationmembantu pengembang menciptakan AWS sumber daya secara teratur dan dapat diprediksi. Sumber daya ditulis dalam file teks menggunakan JSON atau YAML format. Templat memerlukan sebuah sintaks dan struktur tertentu sesuai dengan jenis sumber daya yang sedang dibuat dan dikelola. Anda menulis sumber daya Anda di JSON atau YAML dengan editor kode apa pun seperti AWS Cloud9, memeriksanya ke dalam sistem kontrol versi, dan kemudian CloudFormation membangun layanan yang ditentukan dengan cara yang aman dan berulang.
  - AWS Serverless Application Model (AWS SAM) adalah kerangka kerja sumber terbuka yang dapat Anda gunakan untuk membangun aplikasi tanpa server. AWS AWS SAM terintegrasi dengan AWS layanan lain, dan merupakan perpanjangan dari. AWS CloudFormation
  - AWS Cloud Development Kit (AWS CDK) adalah sebuah kerangka kerja pengembangan perangkat lunak sumber terbuka untuk membuat model dan menyediakan sumber daya aplikasi cloud Anda menggunakan bahasa pemrograman yang sudah dikenal. Anda dapat menggunakan AWS CDK untuk memodelkan infrastruktur aplikasi menggunakan TypeScript, Python, Java, dan. NET. AWS CDK digunakan AWS CloudFormation di latar belakang untuk menyediakan sumber daya dengan cara yang aman dan berulang.

    - AWS Cloud Control APImemperkenalkan kumpulan umum Create, Read, Update, Delete, dan List (CRUDL) APIs untuk membantu pengembang mengelola infrastruktur cloud mereka dengan cara yang mudah dan konsisten. Cloud Control API Common APIs memungkinkan pengembang untuk mengelola siklus hidup dan layanan pihak ketiga secara seragam. AWS
- Implementasikan pola deployment yang meminimalkan dampak-dampaknya terhadap pengguna.
    - Deployment canary:
        - Siapkan penerapan rilis kenari API Gateway
        - Buat pipeline dengan penerapan canary untuk Amazon menggunakan ECS AWS App Mesh
    - Penerapan biru/hijau: Penerapan Biru/Hijau pada AWS whitepaper menjelaskan contoh teknik untuk menerapkan strategi penyebaran biru/hijau.
- Deteksi konfigurasi atau penyimpangan status. Untuk informasi selengkapnya, silakan lihat Mendeteksi perubahan konfigurasi yang tidak terkelola pada tumpukan dan sumber daya.

Sumber daya

Praktik-praktik terbaik terkait:

- REL08-BP05 Menyebarkan perubahan dengan otomatisasi

Dokumen terkait:

- Melakukan otomatisasi deployment secara aman dan otonom
- Memanfaatkan AWS CloudFormation untuk menciptakan infrastruktur yang tidak dapat diubah di Nubank
- Infrastruktur sebagai kode
- Menerapkan alarm untuk secara otomatis mendeteksi penyimpangan di tumpukan AWS CloudFormation

Video terkait:

- AWS Re: Invent 2020: Keandalan, konsistensi, dan kepercayaan diri melalui kekekalan

REL08-BP05 Menyebarkan perubahan dengan otomatisasi

Deployment dan patching diotomatisasi untuk menghilangkan dampak-dampak negatif.

Membuat perubahan pada sistem produksi adalah salah satu area risiko terbesar bagi banyak organisasi. Kami menganggap deployment sebagai masalah kelas pertama untuk diatasi bersama dengan masalah-masalah bisnis yang ditangani oleh perangkat lunak. Saat ini, ini artinya penggunaan otomatisasi kapan saja memungkinkan dalam operasi, termasuk untuk menguji dan melakukan deployment perubahan, menambah atau menghapus kapasitas, dan memigrasikan data.

Hasil yang diinginkan: Anda membangun keamanan deployment otomatis ke dalam proses rilis dengan pengujian pra-produksi yang ekstensif, rollback otomatis, dan deployment produksi yang sangat baik. Otomatisasi ini meminimalkan potensi dampak pada produksi yang disebabkan oleh deployment yang gagal, dan developer tidak perlu lagi mengawasi deployment hingga tahapan produksi secara aktif.

Anti-pola umum:

- Anda melakukan perubahan secara manual.
- Anda melewatkan langkah-langkah dalam otomatisasi Anda melalui alur kerja darurat manual.
- Anda tidak mengikuti rencana dan proses yang telah ditetapkan demi mempercepat kronologi (timeline).
- Anda melakukan deployment susulan cepat tanpa menyediakan waktu menanam.

Manfaat menerapkan praktik terbaik ini: Ketika Anda menggunakan otomatisasi untuk melakukan deployment atas semua perubahan, Anda menghapus kemungkinan adanya kesalahan manusia dan memberikan kemampuan untuk melakukan pengujian sebelum Anda mengubahnya ke tahap produksi. Melakukan proses ini sebelum deployment di lingkungan produksi (push) dapat memverifikasi bahwa rencana Anda sudah lengkap. Selain itu, rollback otomatis ke dalam proses rilis Anda dapat mengidentifikasi masalah-masalah produksi dan mengembalikan beban kerja Anda ke keadaan operasional yang diketahui berfungsi sebelumnya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Lakukan otomatisasi terhadap pipeline deployment Anda. Pipeline deployment memungkinkan Anda untuk menginvokasi pengujian dan deteksi anomali secara otomatis, serta memberi Anda pilihan untuk menghentikan pipeline pada langkah tertentu sebelum deployment produksi atau membatalkan perubahan secara otomatis. Bagian integral dari hal ini adalah adopsi budaya integrasi berkelanjutan dan pengiriman/deployment berkelanjutan (CI/CD), di mana pelaksanaan komit atau perubahan kode

melewati berbagai gerbang tahapan otomatis dari tahap pembuatan dan pengujian hingga tahap deployment pada lingkungan produksi.

Meskipun kebijaksanaan konvensional menyarankan Anda untuk melibatkan personel untuk prosedur operasional paling sulit, kami justru menyarankan Anda mengotomatiskan prosedur paling sulit karena alasan tersebut.

Langkah-langkah implementasi

Anda dapat mengotomatiskan deployment untuk menghapus operasi-operasi manual dengan mengikuti langkah-langkah berikut:

- Siapkan repositori kode untuk menyimpan kode Anda dengan aman: Gunakan AWS CodeCommit, untuk membuat sebuah repositori berbasis Git yang aman.
- Konfigurasikan layanan integrasi berkelanjutan untuk mengompilasi kode sumber, menjalankan pengujian, dan membuat artefak penerapan: Untuk menyiapkan proyek build untuk tujuan ini, lihat Memulai AWS CodeBuild menggunakan konsol.
- Siapkan layanan penyebaran yang mengotomatiskan penerapan aplikasi dan menangani kompleksitas pembaruan aplikasi tanpa bergantung pada penerapan manual yang rawan kesalahan: AWS CodeDeploymengotomatiskan penerapan perangkat lunak ke berbagai layanan komputasi, seperti Amazon,, dan server lokal Anda. EC2 AWS FargateAWS Lambda Untuk mengonfigurasi langkah-langkah ini, lihat Memulai dengan CodeDeploy.
- Siapkan layanan pengiriman berkelanjutan yang mengotomatiskan pipeline rilis Anda untuk pembaruan aplikasi dan infrastruktur yang lebih cepat dan lebih andal: Pertimbangkan menggunakan AWS CodePipeline yang akan membantu Anda mengotomatiskan pipeline rilis. Untuk detail lebih lanjut, lihat CodePipelinetutorial.

Sumber daya

Praktik-praktik terbaik terkait:

- OPS05-BP04 Gunakan sistem manajemen build dan deployment
- OPS05- BP1 0 Mengotomatiskan integrasi dan penyebaran sepenuhnya
- OPS06-BP02 Uji penerapan
- OPS06-BP04 Mengotomatiskan pengujian dan rollback

Dokumen terkait:

- Pengiriman Berkelanjutan dari AWS CloudFormation Tumpukan Bersarang Menggunakan AWS CodePipeline
- CI/CD lengkap dengan AWS CodeCommit,, AWS CodeBuild, dan AWS CodeDeployAWS CodePipeline
- APNMitra: mitra yang dapat membantu Anda membuat solusi penerapan otomatis
- AWS Marketplace: produk yang dapat digunakan untuk mengotomatisasi deployment Anda
- Otomatiskan pesan obrolan dengan webhooks.
- Amazon Builders' Library: Memastikan keamanan rollback selama deployment
- Amazon Builders' Library: Melaju lebih cepat dengan pengiriman berkelanjutan
- Apa itu AWS CodePipeline?
- Apa itu CodeDeploy?
- AWS Manajer Patch Systems Manager
- Apa itu AmazonSES?
- Apa itu Amazon Simple Notification Service?

Video terkait:

- AWS KTT 2019: CI/CD aktif AWS

## Manajemen kegagalan

Pertanyaan

- REL9. Bagaimana cara mencadangkan data?
- REL10. Bagaimana cara menggunakan isolasi kesalahan untuk melindungi beban kerja?
- REL11. Bagaimana Anda mendesain beban kerja agar dapat bertahan jika terjadi kegagalan komponen?
- REL12. Bagaimana cara menguji keandalan?
- REL13. Bagaimana cara Anda mempersiapkan pemulihan bencana (DR)?

### REL9. Bagaimana cara mencadangkan data?

Cadangkan data, aplikasi, dan konfigurasi untuk memenuhi kebutuhan Anda untuk tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO).

Praktik terbaik

- REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber
- REL09-BP02 Mengamankan dan mengenkripsi cadangan
- REL09-BP03 Lakukan pencadangan data secara otomatis
- REL09-BP04 Lakukan pemulihan data secara berkala untuk memverifikasi integritas dan proses cadangan

REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber

Pahami dan gunakan kemampuan-kemampuan pencadangan sumber daya dan layanan data yang digunakan oleh beban kerja. Sebagian besar layanan menyediakan kemampuan untuk mencadangkan data beban kerja.

Hasil yang diinginkan: Sumber data telah diidentifikasi dan diklasifikasikan berdasarkan tingkat kekritisan. Kemudian, buat strategi untuk pemulihan data berdasarkanRPO. Strategi ini melibatkan pencadangan sumber-sumber data, atau memiliki kemampuan untuk memproduksi ulang data dari sumber yang lain. Dalam kasus kehilangan data, strategi yang diterapkan memungkinkan pemulihan atau reproduksi data dalam yang ditentukan RPO danRTO.

Fase kematangan cloud: Dasar

Anti-pola umum:

- Tidak mengetahui semua sumber data untuk beban kerja serta tingkat kekritisannya.
- Tidak melakukan pencadangan sumber data kritis.
- Melakukan pencadangan hanya beberapa sumber data tanpa menggunakan tingkat kekritisan sebagai kriteria.
- Tidak ditentukanRPO, atau frekuensi cadangan tidak dapat memenuhiRPO.
- Tidak mengevaluasi apakah cadangan diperlukan atau apakah data dapat diproduksi ulang dari sumber yang lain.

Manfaat menerapkan praktik terbaik ini: Mengidentifikasi tempat-tempat yang memerlukan pencadangan dan mengimplementasikan mekanisme untuk membuat cadangan, atau mampu

memproduksi ulang data dari sumber eksternal, semuanya dapat meningkatkan kemampuan untuk memulihkan dan mengembalikan data selama pemadaman.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Semua penyimpanan AWS data menawarkan kemampuan cadangan. Layanan seperti Amazon RDS dan Amazon DynamoDB juga mendukung pencadangan otomatis yang point-in-time memungkinkan pemulihan PITR (), yang memungkinkan Anda memulihkan cadangan kapan saja hingga lima menit atau kurang sebelum waktu saat ini. Banyak AWS layanan menawarkan kemampuan untuk menyalin cadangan ke yang lain. Wilayah AWS AWS Backup adalah alat yang memberi Anda kemampuan untuk memusatkan dan mengotomatiskan perlindungan data di seluruh AWS layanan. AWS Elastic Disaster Recoverymemungkinkan Anda menyalin beban kerja server penuh dan mempertahankan perlindungan data berkelanjutan dari on-premise, cross-AZ atau Cross-region, dengan Recovery Point Objective () yang diukur dalam hitungan detik. RPO

Amazon S3 dapat digunakan sebagai tujuan cadangan untuk sumber data yang dikelola sendiri dan AWS dikelola. AWS layanan seperti AmazonEBS, AmazonRDS, dan Amazon DynamoDB telah membangun kemampuan untuk membuat cadangan. Perangkat lunak pencadangan pihak ketiga juga dapat digunakan.

Data lokal dapat dicadangkan ke AWS Cloud penggunaan AWS Storage Gatewayatau AWS DataSync. Bucket Amazon S3 dapat digunakan untuk menyimpan data ini di AWS. Amazon S3 menawarkan beberapa tingkatan penyimpanan seperti Amazon S3 Glacier atau S3 Glacier Deep Archive untuk mengurangi biaya penyimpanan data.

Anda mungkin dapat memenuhi kebutuhan pemulihan data Anda dengan memproduksi ulang data dari sumber yang lain. Misalnya, node ElastiCache replika Amazon atau replika RDS baca Amazon dapat digunakan untuk mereproduksi data jika primer hilang. Dalam kasus di mana sumber seperti ini dapat digunakan untuk memenuhi Tujuan Titik Pemulihan (RPO) dan Tujuan Waktu Pemulihan (RTO), Anda mungkin tidak memerlukan cadangan. Contoh lain, jika bekerja dengan AmazonEMR, mungkin tidak perlu membuat cadangan penyimpanan HDFS data Anda, selama Anda dapat mereproduksi data ke Amazon EMR dari Amazon S3.

Ketika memilih strategi pencadangan, pertimbangkan waktu yang diperlukan untuk melakukan pemulihan data. Waktu yang diperlukan untuk melakukan pemulihan data tergantung pada tipe cadangan (untuk kasus strategi pencadangan), atau kompleksitas mekanisme produksi ulang data. Kali ini harus termasuk dalam RTO beban kerja.

Langkah-langkah implementasi

1. Mengidentifikasi semua sumber daya untuk beban kerja. Data dapat disimpan pada sejumlah sumber daya seperti basis data, volume, filesystem, sistem pencatatan log, dan penyimpanan objek. Lihat bagian Sumber Daya untuk menemukan dokumen terkait pada berbagai AWS layanan tempat data disimpan, dan kemampuan cadangan yang disediakan layanan ini.
2. Klasifikasikan sumber data berdasarkan tingkat kekritisan. Set data yang berbeda akan memiliki tingkat kekritisan yang berbeda untuk suatu beban kerja, sehingga memiliki persyaratan ketahanan yang berbeda pula. Misalnya, beberapa data mungkin kritis dan memerlukan RPO mendekati nol, sementara data lain mungkin kurang kritis dan dapat mentolerir kehilangan data yang lebih tinggi RPO dan beberapa. Demikian pula, kumpulan data yang berbeda mungkin memiliki RTO persyaratan yang berbeda juga.
3. Gunakan AWS atau layanan pihak ketiga untuk membuat cadangan data. AWS Backupadalah layanan terkelola yang memungkinkan pembuatan cadangan berbagai sumber data di. AWSAWS Elastic Disaster Recoverymenangani replikasi data sub-detik otomatis ke file. Wilayah AWS Sebagian besar AWS layanan juga memiliki kemampuan asli untuk membuat cadangan. Mereka AWS Marketplace memiliki banyak solusi yang menyediakan kemampuan ini juga. Lihat Sumber Daya yang disebutkan di bawah ini untuk mendapatkan informasi tentang cara membuat cadangan data dari berbagai layanan AWS .
4. Untuk data yang tidak dicadangkan, bangun mekanisme produksi ulang data. Anda mungkin memilih untuk tidak mencadangkan data yang dapat diproduksi ulang dari sumber yang lain karena berbagai alasan. Mungkin terdapat situasi di mana produksi ulang data dari sumber yang lain saat diperlukan lebih murah daripada membuat cadangan, karena mungkin ada biaya-biaya yang timbul terkait penyimpanan cadangan. Contoh lain adalah di mana memulihkan dari cadangan membutuhkan waktu lebih lama daripada mereproduksi data dari sumber, yang mengakibatkan pelanggaran. RTO Pada situasi-situasi demikian, pertimbangkan semua kompromi dan bangun sebuah proses yang ditetapkan dengan baik terkait bagaimana data dapat diproduksi ulang dari sumber-sumber ini saat pemulihan data diperlukan. Misalnya, jika Anda telah memuat data dari Amazon S3 ke gudang data (seperti Amazon Redshift), MapReduce atau cluster (seperti EMR Amazon) untuk melakukan analisis pada data tersebut, ini mungkin merupakan contoh data yang dapat direproduksi dari sumber lain. Selama hasil analisis ini disimpan di suatu tempat atau dapat direproduksi, Anda tidak akan mengalami kehilangan data karena kegagalan di gudang data atau cluster. MapReduce Contoh lain yang dapat direproduksi dari sumber termasuk cache (seperti Amazon ElastiCache) atau replika RDS baca.
5. Buat jadwal pencadangan data. Membuat cadangan sumber data adalah proses periodik dan frekuensinya harus bergantung pada. RPO

Tingkat upaya untuk Rencana Implementasi: Sedang

Sumber daya

Praktik-Praktik Terbaik Terkait:

REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data

REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan

Dokumen terkait:

- Apa itu AWS Backup?
- Apa itu AWS DataSync?
- Apa itu Volume Gateway?
- APNMitra: mitra yang dapat membantu dengan cadangan
- AWS Marketplace: produk yang dapat digunakan untuk pencadangan
- EBSCuplikan Amazon
- Mencadangkan Amazon EFS
- Mencadangkan Amazon FSx untuk Windows File Server
- Backup dan Restore ElastiCache untuk Redis
- Membuat Snapshot Klaster DB di Neptune
- Membuat Snapshot DB
- Membuat EventBridge Aturan yang Memicu Jadwal
- Replikasi Lintas Wilayah dengan Amazon S3
- EFS-untuk- EFS AWS Backup
- Mengekspor Data Log ke Amazon S3
- Manajemen siklus aktif objek
- Pencadangan Sesuai Permintaan dan Pemulihan untuk DynamoDB
- oint-in-timePemulihan P untuk DynamoDB
- Bekerja dengan Snapshot Indeks OpenSearch Layanan Amazon
- Apa itu AWS Elastic Disaster Recovery?

Video terkait:

- AWS re:invent 2021 - Backup, pemulihan bencana, dan perlindungan ransomware dengan AWS
- AWS Backup Demo: Cadangan Lintas Akun dan Lintas Wilayah
- AWS Re: invent 2019: Menyelam dalam, ft. AWS Backup Rackspace () STG341

Contoh terkait:

- Well-Architected Lab - Menerapkan Replikasi Lintas Wilayah Bi-Directional () untuk Amazon S3 CRR
- Lab Well-Architected - Menguji Cadangan dan Penyimpanan Kembali Data
- Lab Well-Architected: Pencadangan dan Pemulihan dengan Failback untuk Beban Kerja Analitik
- Lab Well-Architected: Pemulihan Bencana - Pencadangan dan Pemulihan

REL09-BP02 Mengamankan dan mengenkripsi cadangan

Kontrol dan deteksi akses ke cadangan menggunakan autentikasi dan otorisasi. Gunakan enkripsi untuk mencegah dan mendeteksi jika integritas data cadangan terancam.

Anti-pola umum:

- Buatlah akses yang sama ke cadangan dan otomatisasi pemulihan seperti yang dilakukan pada data.
- Tidak mengenkripsi cadangan.

Manfaat menerapkan praktik terbaik ini: Mengamankan data Anda akan mencegah adanya gangguan terhadap data, dan enkripsi data mencegah akses ke data tersebut jika tidak sengaja terekspos.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kontrol dan deteksi akses ke cadangan menggunakan otentikasi dan otorisasi, seperti (). AWS Identity and Access Management IAM Gunakan enkripsi untuk mencegah dan mendeteksi jika integritas data cadangan terancam.

Amazon S3 mendukung beberapa metode enkripsi data diam. Dengan menggunakan enkripsi di sisi server, Amazon S3 dapat menerima objek sebagai data yang tidak terenkripsi dan mengenkripsinya

saat disimpan. Dengan menggunakan enkripsi di sisi klien, aplikasi beban kerja bertanggung jawab untuk mengenkripsi data sebelum mengirimkannya ke Amazon S3. Kedua metode memungkinkan Anda menggunakan AWS Key Management Service (AWS KMS) untuk membuat dan menyimpan kunci data, atau Anda dapat memberikan kunci Anda sendiri, yang kemudian menjadi tanggung jawab Anda. Dengan menggunakan AWS KMS, Anda dapat menetapkan kebijakan menggunakan IAM siapa yang dapat dan tidak dapat mengakses kunci data dan data yang didekripsi.

Untuk AmazonRDS, jika Anda telah memilih untuk mengenkripsi database Anda, maka cadangan Anda juga dienkripsi. Pencadangan DynamoDB selalu dienkripsi. Saat menggunakan AWS Elastic Disaster Recovery, semua data dalam perjalanan dan saat istirahat dienkripsi. Dengan Elastic Disaster Recovery, data saat istirahat dapat dienkripsi menggunakan Kunci EBS Enkripsi Volume enkripsi Amazon default atau kunci yang dikelola pelanggan khusus.

Langkah-langkah implementasi

1. Gunakan enkripsi untuk setiap penyimpanan data. Jika sumber data terenkripsi, maka cadangannya juga akan terenkripsi.
   - Gunakan enkripsi di AmazonRDS. . Anda dapat mengonfigurasi enkripsi saat istirahat menggunakan AWS Key Management Service saat Anda membuat RDS instance.
   - Gunakan enkripsi pada EBS volume Amazon. . Anda dapat mengonfigurasi enkripsi default atau menentukan sebuah kunci unik saat pembuatan volume.
   - Gunakan enkripsi Amazon DynamoDB yang diperlukan. DynamoDB mengenkripsi semua data diam. Anda dapat menggunakan kunci yang AWS dimiliki atau AWS KMS kunci AWS terkelolaKMS, menentukan kunci yang disimpan di akun Anda.
   - Enkripsi data Anda yang disimpan di Amazon EFS. Konfigurasikan enkripsi saat Anda membuat sistem file.
   - Konfigurasikan enkripsi di Wilayah sumber dan tujuan. Anda dapat mengonfigurasi enkripsi saat istirahat di Amazon S3 menggunakan kunci yang disimpanKMS, tetapi kuncinya khusus Wilayah. Anda dapat menentukan kunci tujuan saat Anda mengonfigurasikan replikasi.
   - Pilih apakah akan menggunakan EBSenkripsi Amazon default atau kustom untuk Elastic Disaster Recovery. Opsi ini akan mengenkripsi data diam yang direplikasi di diska Subnet Area Staging dan diska yang direplikasi.
2. Implementasikan izin hak akses paling rendah untuk mengakses cadangan Anda. Ikuti praktik-praktik terbaik untuk membatasi akses ke cadangan, snapshot, dan replika sesuai dengan praktik terbaik untuk keamanan.

Sumber daya

Dokumen terkait:

- AWS Marketplace: produk yang dapat digunakan untuk pencadangan
- EBSEnkripsi Amazon
- Amazon S3: Melindungi Data Menggunakan Enkripsi
- CRRKonfigurasi Tambahan: Mereplikasi Objek yang Dibuat dengan Enkripsi Sisi Server (SSE) Menggunakan Kunci Enkripsi yang disimpan di AWS KMS
- Enkripsi DynamoDB Saat Diam
- Mengenkripsi Sumber Daya Amazon RDS
- Mengenkripsi Data dan Metadata di Amazon EFS
- Enkripsi untuk Backup di AWS
- Mengelola Tabel yang Dienkripsi
- Pilar Keamanan - Kerangka AWS Well-Architected
- Apa itu AWS Elastic Disaster Recovery?

Contoh terkait:

- Well-Architected Lab - Menerapkan Replikasi Lintas Wilayah Bi-Directional () untuk Amazon S3 CRR

REL09-BP03 Lakukan pencadangan data secara otomatis

Konfigurasikan backup yang akan diambil secara otomatis berdasarkan jadwal periodik yang diinformasikan oleh Recovery Point Objective (RPO), atau oleh perubahan dalam dataset. Set data kritis dengan persyaratan data hilang yang rendah perlu dicadangkan otomatis secara rutin, sedangkan data yang tidak terlalu kritis yang apabila hilang masih dapat dimaklumi dapat dicadangkan tidak terlalu sering.

Hasil yang Diinginkan: Sebuah proses otomatis yang membuat cadangan sumber data dengan jadwal yang ditetapkan.

Anti-pola umum:

- Melakukan pencadangan secara manual.

- Menggunakan sumber daya yang memiliki kemampuan pencadangan, tetapi tidak termasuk pencadangan dalam otomatisasi Anda.

Manfaat membangun praktik terbaik ini: Mengotomatiskan cadangan memverifikasi bahwa mereka diambil secara teratur berdasarkan AndaRPO, dan memberi tahu Anda jika tidak diambil.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS Backup dapat digunakan untuk membuat backup data otomatis dari berbagai sumber AWS data. RDSInstans Amazon dapat dicadangkan hampir terus menerus setiap lima menit dan objek Amazon S3 dapat dicadangkan hampir terus menerus setiap lima belas menit, menyediakan point-in-time pemulihan PITR () ke titik waktu tertentu dalam riwayat cadangan. Untuk sumber AWS data lain, seperti EBS volume Amazon, tabel Amazon DynamoDB, atau sistem file FSx Amazon AWS Backup , dapat menjalankan pencadangan otomatis sesering setiap jam. Layanan ini juga menawarkan kemampuan cadangan asli. AWS layanan yang menawarkan pencadangan otomatis dengan point-in-time pemulihan termasuk Amazon DynamoDB, RDS Amazon, dan Amazon Keyspaces (untuk Apache Cassandra) - ini dapat dikembalikan ke titik waktu tertentu dalam riwayat pencadangan. Sebagian besar layanan penyimpanan data AWS lainnya menawarkan kemampuan untuk menjadwalkan pencadangan secara berkala, dengan frekuensi setiap satu jam.

Amazon RDS dan Amazon DynamoDB menawarkan pencadangan berkelanjutan dengan pemulihan. point-in-time Penentuan versi Amazon S3, setelah dihidupkan, akan berjalan otomatis. Amazon Data Lifecycle Manager dapat digunakan untuk mengotomatiskan pembuatan, penyalinan, dan penghapusan snapshot Amazon. EBS Ini juga dapat mengotomatiskan pembuatan, penyalinan, penghentian, dan deregistrasi Amazon Machine Images () yang EBS didukung Amazon dan snapshot Amazon yang mendasarinya. AMIs EBS

AWS Elastic Disaster Recovery menyediakan replikasi tingkat blok berkelanjutan dari lingkungan sumber (lokal atau AWS) ke wilayah pemulihan target. Point-in-timeEBSSnapshot Amazon secara otomatis dibuat dan dikelola oleh layanan.

Untuk tampilan terpusat dari otomatisasi dan riwayat pencadangan Anda, AWS Backup sediakan solusi pencadangan berbasis kebijakan yang dikelola sepenuhnya. Layanan ini memusatkan dan mengotomatiskan pencadangan data di beberapa layanan AWS di cloud serta on-premise dengan menggunakan AWS Storage Gateway.

Selain penentuan versi, Amazon S3 dilengkapi dengan fitur replikasi. Seluruh bucket S3 dapat direplikasi secara otomatis ke bucket lain yang ada di Wilayah AWS yang sama atau berbeda.

Langkah-langkah implementasi

1. Identifikasi sumber data yang sedang dicadangkan secara manual. Untuk detail selengkapnya, lihat REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber.
2. Tentukan RPO untuk beban kerja. Untuk detail selengkapnya, lihat REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data.
3. Gunakan solusi pencadangan otomatis atau layanan terkelola. AWS Backup adalah layanan yang dikelola sepenuhnya yang memudahkan untuk memusatkan dan mengotomatiskan perlindungan data di seluruh AWS layanan, di cloud, dan lokal. Dengan menggunakan rencana cadangan di AWS Backup, buatlah aturan yang menetapkan sumber daya yang akan dicadangkan, dan frekuensi pembuatan cadangan ini. Frekuensi ini harus diinformasikan oleh yang RPO ditetapkan pada Langkah 2. Untuk panduan langsung tentang cara membuat backup otomatis menggunakan, AWS Backup lihat Menguji Backup and Restore of Data. Kemampuan cadangan asli ditawarkan oleh sebagian besar AWS layanan yang menyimpan data. Misalnya, RDS dapat dimanfaatkan untuk backup otomatis dengan point-in-time recovery (). PITR
4. Untuk sumber data yang tidak didukung oleh solusi pencadangan otomatis atau layanan terkelola seperti sumber data on-premise atau antrean pesan, pertimbangkan untuk menggunakan solusi pihak ketiga tepercaya untuk membuat cadangan secara otomatis. Atau, Anda dapat membuat otomatisasi untuk melakukan ini menggunakan AWS CLI atauSDKs. Anda dapat menggunakan AWS Lambda Fungsi atau AWS Step Functions untuk menentukan logika yang terlibat dalam membuat cadangan data, dan menggunakan Amazon EventBridge untuk memanggilnya pada frekuensi berdasarkan pada AndaRPO.

Tingkat upaya untuk Rencana Implementasi: Rendah

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu dengan cadangan
- AWS Marketplace: produk yang dapat digunakan untuk pencadangan
- Membuat EventBridge Aturan yang Memicu Jadwal
- Apa itu AWS Backup?

- Apa itu AWS Step Functions?
- Apa itu AWS Elastic Disaster Recovery?

Video terkait:

- AWS Re: invent 2019: Menyelam dalam, ft. AWS Backup Rackspace () STG341

Contoh terkait:

- Lab Well-Architected - Menguji Cadangan dan Penyimpanan Kembali Data

REL09-BP04 Lakukan pemulihan data secara berkala untuk memverifikasi integritas dan proses cadangan

Validasi bahwa implementasi proses pencadangan Anda memenuhi Tujuan Waktu Pemulihan (RTO) dan Tujuan Titik Pemulihan (RPO) Anda dengan melakukan tes pemulihan.

Hasil yang diinginkan: Data dari cadangan dipulihkan secara berkala menggunakan mekanisme yang terdefinisi dengan baik untuk memverifikasi bahwa pemulihan dimungkinkan dalam tujuan waktu pemulihan yang ditetapkan (RTO) untuk beban kerja. Verifikasi bahwa restorasi dari cadangan menghasilkan sumber daya yang berisi data asli tanpa ada yang rusak atau tidak dapat diakses, dan dengan kehilangan data dalam tujuan titik pemulihan ()RPO.

Anti-pola umum:

- Memulihkan cadangan, tetapi tidak mengambil data atau membuat kueri data apa pun untuk memastikan bahwa data hasil pemulihan dapat digunakan.
- Dengan anggapan bahwa cadangan sudah ada.
- Dengan anggapan bahwa cadangan sistem dapat dioperasikan sepenuhnya dan data dapat dipulihkan dari sistem.
- Dengan asumsi bahwa waktu untuk memulihkan atau memulihkan data dari cadangan termasuk dalam RTO beban kerja.
- Dengan asumsi bahwa data yang terkandung pada cadangan termasuk dalam RPO beban kerja
- Melakukan pemulihan apabila diperlukan, tanpa menggunakan runbook, atau di luar prosedur otomatis yang ditetapkan.

Manfaat membangun praktik terbaik ini: Menguji pemulihan cadangan memverifikasi bahwa data dapat dipulihkan bila diperlukan tanpa khawatir bahwa data mungkin hilang atau rusak, bahwa pemulihan dan pemulihan dimungkinkan dalam RTO untuk beban kerja, dan kehilangan data apa pun termasuk dalam beban kerja. RPO

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Pengujian kemampuan pencadangan dan pemulihan akan meningkatkan keyakinan Anda pada kemampuan untuk menjalankan tindakan ini selama terjadi pemadaman (outage). Pulihkan cadangan ke lokasi baru secara berkala dan lakukan pengujian untuk memastikan integritas data. Beberapa tes umum yang harus dilakukan adalah memeriksa apakah semua data tersedia, tidak rusak, dapat diakses, dan bahwa setiap kehilangan data termasuk dalam RPO beban kerja. Tes semacam itu juga dapat membantu memastikan apakah mekanisme pemulihan cukup cepat untuk mengakomodasi beban kerja. RTO

Dengan menggunakan AWS, Anda dapat berdiri di lingkungan pengujian dan memulihkan cadangan Anda untuk menilai RTO dan RPO kemampuan, dan menjalankan pengujian pada konten dan integritas data.

Selain itu, Amazon RDS dan Amazon DynamoDB point-in-time memungkinkan pemulihan (). PITR Dengan menggunakan pencadangan yang berkelanjutan, Anda dapat memulihkan set data ke statusnya sesuai dengan waktu dan tanggal yang ditentukan.

Jika semua data tersedia, tidak rusak, dapat diakses, dan kehilangan data apa pun termasuk dalam RPO beban kerja. Tes semacam itu juga dapat membantu memastikan apakah mekanisme pemulihan cukup cepat untuk mengakomodasi beban kerja. RTO

AWS Elastic Disaster Recovery menawarkan snapshot point-in-time pemulihan berkelanjutan dari volume AmazonEBS. Saat server sumber direplikasi, point-in-time status dicatat dari waktu ke waktu berdasarkan kebijakan yang dikonfigurasi. Pemulihan Bencana Elastis dapat membantu Anda untuk memverifikasi integritas snapshot ini dengan meluncurkan instans untuk tujuan pengujian dan latihan tanpa mengarahkan lalu lintas.

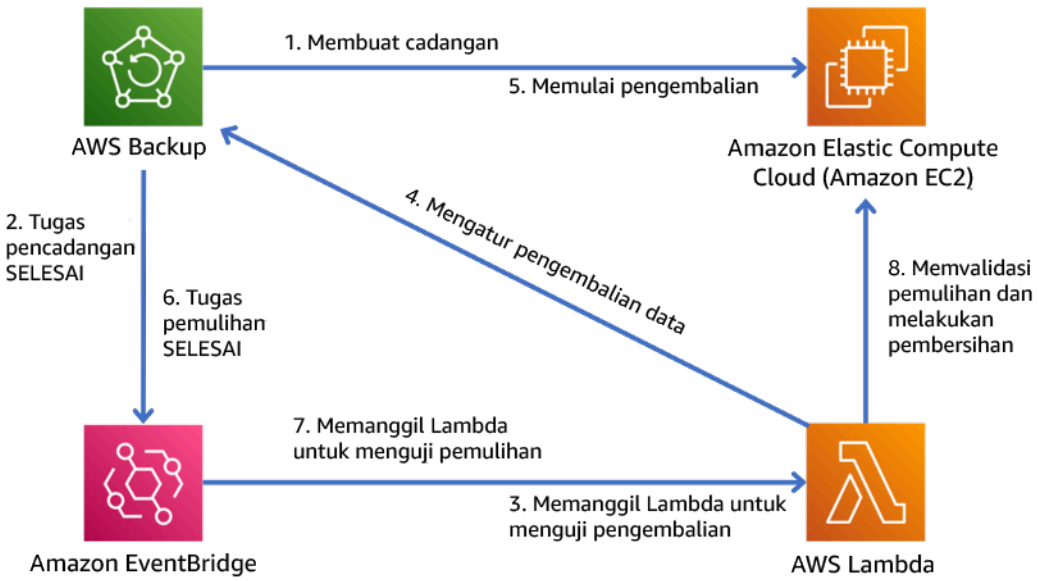
Langkah-langkah implementasi

1. Identifikasi sumber data yang dicadangkan saat ini dan lokasi penyimpanan cadangan tersebut. Untuk panduan implementasi, lihat REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber.

2. Menetapkan kriteria untuk validasi data untuk masing-masing sumber data. Jenis data yang berbeda akan memiliki properti data yang berbeda, yang dapat memerlukan mekanisme validasi yang berbeda. Pertimbangkan bagaimana data ini dapat divalidasi sebelum Anda yakin untuk menggunakannya dalam produksi. Beberapa cara umum untuk memvalidasi data adalah dengan menggunakan data dan properti pencadangan seperti jenis data, format, checksum, ukuran, atau gabungan darinya dengan logika validasi kustom. Misalnya, hal ini dapat dilakukan dengan perbandingan nilai checksum antara sumber daya yang dipulihkan dan sumber data pada waktu cadangan dibuat.
3. Menetapkan RTO dan RPO memulihkan data berdasarkan kekritisan data. Untuk panduan implementasi, lihat REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data.
4. Menilai kemampuan pemulihan Anda. Tinjau strategi pencadangan dan pemulihan Anda untuk memahami apakah itu dapat memenuhi Anda RTO danRPO, dan sesuaikan strategi yang diperlukan. Dengan menggunakan AWS Resilience Hub, Anda dapat menjalankan penilaian terhadap beban kerja Anda. Penilaian mengevaluasi konfigurasi aplikasi Anda terhadap kebijakan ketahanan dan laporan jika Anda RTO dan RPO target dapat dipenuhi.
5. Lakukan penyimpanan kembali pengujian dengan menggunakan proses yang ditetapkan saat ini yang digunakan dalam produksi untuk pemulihan data. Proses ini bergantung pada cara sumber data asli dicadangkan, format dan lokasi penyimpanan cadangan tersebut, atau apakah data diproduksi ulang dari sumber-sumber lainnya. Misalnya, jika Anda menggunakan sebuah layanan terkelola seperti AWS Backup, hal ini mungkin sesederhana memulihkan cadangan ke sumber daya baru. Jika Anda menggunakan AWS Elastic Disaster Recovery , maka Anda dapat meluncurkan latihan pemulihan.
6. Validasi pemulihan data dari sumber daya yang dipulihkan berdasarkan kriteria validasi data yang sebelumnya Anda buat. Apakah data yang direstorasi dan dipulihkan memiliki sebagian besar catatan atau item terbaru pada waktu pencadangan? Apakah data ini termasuk dalam RPO beban kerja?
7. Ukur waktu yang diperlukan untuk pemulihan dan pemulihan dan bandingkan dengan yang Anda mapanRTO. Apakah proses ini termasuk dalam RTO beban kerja? Misalnya, bandingkan stempel waktu dari kapan proses restorasi dimulai dan kapan validasi pemulihan selesai untuk menghitung waktu yang diperlukan proses ini. Semua AWS API panggilan diberi stempel waktu dan informasi ini tersedia di. AWS CloudTrail Meskipun informasi ini dapat menyediakan detail waktu kapan proses pemulihan dimulai, namun stempel waktu akhir yang menunjukkan kapan validasi diselesaikan harus dicatat berdasarkan logika validasi Anda. Jika Anda menggunakan proses otomatis, maka layanan-layanan seperti Amazon DynamoDB dapat digunakan untuk

menyimpan informasi ini. Selain itu, banyak AWS layanan menyediakan riwayat peristiwa yang memberikan informasi stempel waktu ketika tindakan tertentu terjadi. Di dalam AWS Backup, tindakan pencadangan dan pemulihan disebut sebagai pekerjaan, dan pekerjaan ini berisi informasi stempel waktu sebagai bagian dari metadatanya yang dapat digunakan untuk mengukur waktu yang diperlukan untuk pemulihan dan pemulihan.

8. Beri tahu pemangku kepentingan jika validasi data gagal, atau jika waktu yang diperlukan untuk pemulihan dan pemulihan melebihi yang ditetapkan RTO untuk beban kerja. Saat menerapkan otomatisasi untuk melakukan ini, seperti di lab ini, layanan seperti Amazon Simple Notification Service (AmazonSNS) dapat digunakan untuk mengirim pemberitahuan push seperti email atau SMS kepada pemangku kepentingan. Pesan-pesan ini juga dapat dipublikasikan ke aplikasi perpesanan seperti Amazon Chime, Slack, atau Microsoft Teams atau digunakan untuk membuat tugas seperti menggunakan Systems OpsItems Manager AWS. OpsCenter
9. Otomatiskan proses ini untuk menjalankannya secara berkala. Misalnya, layanan seperti AWS Lambda atau State Machine in AWS Step Functions dapat digunakan untuk mengotomatiskan proses pemulihan dan pemulihan, dan Amazon EventBridge dapat digunakan untuk menjalankan alur kerja otomatisasi ini secara berkala seperti yang ditunjukkan pada diagram arsitektur di bawah ini. Pelajari cara Mengotomatiskan validasi pemulihan data dengan. AWS Backup Selain itu, lab Well-Architected ini memberikan pengalaman langsung tentang satu cara untuk melakukan otomatisasi untuk beberapa langkah yang diuraikan di sini.




Gambar 9. Proses pencadangan dan pemulihan otomatis

Tingkat upaya untuk Rencana Implementasi: Sedang hingga tinggi tergantung pada kompleksitas kriteria validasinya.

Sumber daya

Dokumen terkait:

- Otomatiskan validasi pemulihan data dengan AWS Backup
- APNMitra: mitra yang dapat membantu dengan cadangan
- AWS Marketplace: produk yang dapat digunakan untuk pencadangan
- Membuat EventBridge Aturan yang Memicu Jadwal
- Pencadangan sesuai permintaan dan pemulihan untuk DynamoDB
- Apa itu AWS Backup?
- Apa itu AWS Step Functions?
- Apa itu AWS Elastic Disaster Recovery
- AWS Elastic Disaster Recovery

Contoh terkait:

- Lab Well-Architected: Menguji Cadangan dan Penyimpanan Kembali Data

## REL10. Bagaimana cara menggunakan isolasi kesalahan untuk melindungi beban kerja?

Batas isolasi kesalahan membatasi efek dari sebuah kegagalan di dalam beban kerja hingga jumlah komponen yang terbatas. Komponen-komponen yang ada di luar batasan-batasan ini tidak terpengaruh oleh kegagalan tersebut. Menggunakan beberapa batasan isolasi kesalahan, Anda dapat membatasi dampak pada beban kerja Anda.

Praktik terbaik

- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL10-BP02 Pilih lokasi yang sesuai untuk penyebaran multi-lokasi Anda
- REL10-BP03 Mengotomatiskan pemulihan untuk komponen yang dibatasi ke satu lokasi
- REL10-BP04 Gunakan arsitektur sekat untuk membatasi ruang lingkup dampak

REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi

Distribusikan sumber daya dan data beban kerja ke beberapa Zona Ketersediaan atau, jika diperlukan, ke beberapa Wilayah AWS. Lokasi tersebut dapat beragam sesuai kebutuhan.

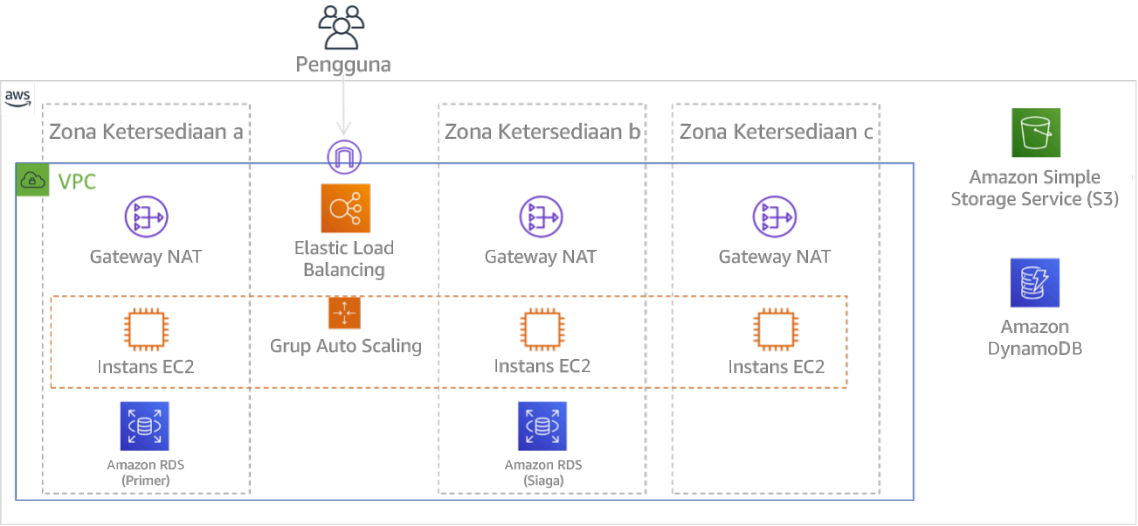
Salah satu prinsip dasar untuk desain layanan AWS adalah menghindari titik tunggal kegagalan dalam infrastruktur fisik yang mendasarinya. Hal ini memotivasi kami untuk membangun sistem dan perangkat lunak yang menggunakan beberapa Zona Ketersediaan dan tahan terhadap kegagalan yang terjadi di satu zona. Dengan cara yang serupa, sistem dibangun agar tahan terhadap kegagalan yang terjadi di satu simpul komputasi, satu volume penyimpanan, atau satu instans basis data. Saat membangun sistem yang bergantung pada komponen yang berlebihan, penting untuk memastikan bahwa komponen beroperasi secara independen, dan dalam kasus, secara mandiri. Wilayah AWS Manfaat yang diperoleh dengan melakukan kalkulasi ketersediaan teoretis dengan komponen redundan hanya valid jika dapat dibuktikan kebenarannya.

Zona Ketersediaan (AZs)

Wilayah AWS terdiri dari beberapa Availability Zone yang dirancang untuk independen satu sama lain. Setiap Zona Ketersediaan dipisahkan oleh jarak fisik yang cukup dari zona lain untuk menghindari skenario kegagalan terkait yang disebabkan oleh bahaya-bahaya lingkungan seperti kebakaran, banjir, dan tornado. Setiap Zona Ketersediaan juga memiliki infrastruktur fisik yang independen: koneksi khusus ke daya utilitas, sumber daya cadangan mandiri, layanan mekanis independen, dan konektivitas jaringan independen di dalam dan di luar Zona Ketersediaan. Desain ini membatasi kesalahan yang terjadi dalam salah satu sistem agar hanya berdampak pada satu AZ saja. Meskipun terpisah secara geografis, Zona Ketersediaan berada di wilayah yang sama yang akan memungkinkan jaringan mempunyai latensi rendah dan throughput tinggi. Seluruh Wilayah AWS (di semua Availability Zone, yang terdiri dari beberapa pusat data yang independen secara fisik) dapat diperlakukan sebagai target penerapan logis tunggal untuk beban kerja Anda, termasuk kemampuan untuk mereplikasi data secara sinkron (misalnya, antar database). Hal ini memungkinkan Anda untuk menggunakan Zona Ketersediaan dalam konfigurasi aktif/aktif atau aktif/siaga.

Zona Ketersediaan bersifat independen, dan oleh karena itu ketersediaan beban kerja akan meningkat saat beban kerja dirancang untuk menggunakan beberapa zona. Beberapa AWS layanan (termasuk pesawat data EC2 instans Amazon) digunakan sebagai layanan zona ketat di mana mereka telah berbagi nasib dengan Availability Zone tempat mereka berada. EC2Contoh Amazon di sisi lain tidak AZs akan terpengaruh dan terus berfungsi. Dengan cara yang serupa, jika terjadi kesalahan di Zona Ketersediaan yang menyebabkan basis data Amazon Aurora gagal, instans Aurora replika baca di AZ yang tidak terdampak dapat dipindahkan ke AZ utama secara otomatis.

Sebaliknya, layanan AWS regional seperti Amazon DynamoDB secara internal menggunakan beberapa Zona Ketersediaan dalam konfigurasi aktif/aktif guna mencapai tujuan desain ketersediaan untuk layanan tersebut, tanpa perlu mengonfigurasi penempatan AZ.




Gambar 9: Arsitektur multitingkat di-deploy di tiga Zona Ketersediaan. Perhatikan bahwa Amazon S3 dan Amazon DynamoDB selalu Multi-AZ secara otomatis. ELBJuga dikerahkan ke ketiga zona.

Meskipun pesawat AWS kontrol biasanya menyediakan kemampuan untuk mengelola sumber daya di seluruh Wilayah (beberapa Availability Zone), pesawat kontrol tertentu (termasuk Amazon EC2 dan AmazonEBS) memiliki kemampuan untuk memfilter hasil ke satu Availability Zone. Saat hal ini sudah dilakukan, permintaan hanya diproses di Zona Ketersediaan tertentu, mengurangi eksposur gangguan di Zona Ketersediaan lainnya. AWS CLI Contoh ini menggambarkan mendapatkan informasi EC2 instans Amazon hanya dari Zona Ketersediaan us-east-2c:

```
AWS ec2 describe-instances --filters Name=availability-zone,Values=us-east-2c
```

### AWS Local Zones

AWS Local Zones bertindak mirip dengan Availability Zone Wilayah AWS dalam masing-masing sehingga mereka dapat dipilih sebagai lokasi penempatan untuk AWS sumber daya zona seperti subnet dan EC2 instance. Apa yang membuat mereka istimewa adalah bahwa mereka tidak berada di tempat yang terkait Wilayah AWS, tetapi dekat dengan populasi besar, industri, dan pusat TI di mana tidak Wilayah AWS ada saat ini. Namun, zona-zona ini tetap mampu mempertahankan bandwidth yang tinggi, mengamankan koneksi yang ada di antara beban kerja di zona lokal dan yang dijalankan di Wilayah AWS. Anda harus menggunakan Zona Lokal AWS untuk melakukan deployment beban kerja secara lebih dekat dengan pengguna untuk persyaratan latensi rendah.

Amazon Global Edge Network

Amazon Global Edge Network terdiri dari lokasi edge yang ada di kota-kota di seluruh dunia. Amazon CloudFront menggunakan jaringan ini untuk mengirimkan konten kepada pengguna akhir dengan latensi lebih rendah. AWS Global Accelerator memungkinkan Anda membuat titik akhir beban kerja di lokasi tepi ini untuk memberikan orientasi ke jaringan AWS global yang dekat dengan pengguna Anda. Amazon API Gateway memungkinkan API titik akhir yang dioptimalkan tepi menggunakan CloudFront distribusi untuk memfasilitasi akses klien melalui lokasi tepi terdekat.

Wilayah AWS

Wilayah AWS dirancang untuk menjadi otonom, oleh karena itu, untuk menggunakan pendekatan Multi-wilayah Anda akan menyebarkan salinan layanan khusus untuk setiap Wilayah.

Pendekatan banyak Wilayah adalah umum untuk strategi pemulihan bencana guna memenuhi tujuan-tujuan pemulihan ketika peristiwa skala besar satu kali terjadi. Lihat Rencana Pemulihan Bencana (DR) untuk informasi selengkapnya tentang strategi ini. Namun di sini, kami berfokus pada ketersediaan, yang berupaya memberikan tujuan waktu aktif rata-rata (mean uptime) dari waktu ke waktu. Untuk tujuan ketersediaan tinggi, arsitektur multi-wilayah umumnya akan dirancang menjadi aktif/aktif, dengan setiap salinan layanan (di wilayah masing-masing) yang aktif (melayani permintaan).

> **Rekomendasi**
>
> Tujuan-tujuan ketersediaan untuk sebagian besar beban kerja dapat dipenuhi dengan menggunakan strategi Multi-AZ dalam satu Wilayah AWS. Pertimbangkan untuk menggunakan arsitektur multi-Wilayah hanya saat beban kerja memiliki persyaratan ketersediaan yang sangat tinggi, atau tujuan bisnis lain yang memerlukan arsitektur multi-Wilayah.

AWS memberi Anda kemampuan untuk mengoperasikan layanan lintas wilayah. Misalnya, AWS menyediakan replikasi data asinkron berkelanjutan dari data menggunakan Amazon Simple Storage Service (Amazon S3) Replikasi Simple Storage Service (Amazon S3), Amazon Read Replicas (termasuk Aurora Read Replicas), dan RDS Amazon DynamoDB Global Tables. Dengan replikasi berkelanjutan, versi data tersedia untuk bisa langsung digunakan di setiap Wilayah aktif.

Dengan menggunakan AWS CloudFormation, Anda dapat menentukan infrastruktur Anda dan menerapkannya secara konsisten di seluruh Akun AWS dan di seluruh Wilayah AWS. Dan

AWS CloudFormation StackSets memperluas fungsi ini dengan memungkinkan Anda membuat, memperbarui, atau menghapus AWS CloudFormation tumpukan di beberapa akun dan wilayah dengan satu operasi. Untuk penerapan EC2 instans Amazon, (AMIAmazon Machine Image) digunakan untuk menyediakan informasi seperti konfigurasi perangkat keras dan perangkat lunak yang diinstal. Anda dapat menerapkan pipeline Amazon EC2 Image Builder yang membuat kebutuhan AMIs Anda dan menyalinnya ke wilayah aktif Anda. Ini memastikan bahwa Golden ini AMIs memiliki semua yang Anda butuhkan untuk menerapkan dan meningkatkan beban kerja Anda di setiap wilayah baru.

Untuk merutekan lalu lintas, Amazon Route 53 dan AWS Global Accelerator mengizinkan definisi kebijakan yang menentukan pengguna mana yang pergi ke titik akhir regional aktif mana. Dengan Akselerator Global, Anda dapat mengatur panggilan lalu lintas untuk mengontrol persentase lalu lintas yang diarahkan ke setiap titik akhir aplikasi. Route 53 mendukung pendekatan persentase ini, dan juga beberapa kebijakan lain yang tersedia termasuk kebijakan berbasis kedekatan geografis (geo-proximity) dan latensi. Global Accelerator secara otomatis memanfaatkan jaringan server AWS edge yang luas, untuk mengarahkan lalu lintas ke tulang punggung AWS jaringan sesegera mungkin, menghasilkan latensi permintaan yang lebih rendah.

Semua kemampuan ini dioperasikan untuk menjaga otonomi dari masing-masing Wilayah. Ada sangat sedikit pengecualian untuk pendekatan ini, termasuk layanan kami yang menyediakan pengiriman edge global (seperti Amazon CloudFront dan Amazon Route 53), bersama dengan pesawat kontrol untuk layanan AWS Identity and Access Management (IAM). Sebagian besar layanan dioperasikan sepenuhnya dalam satu Wilayah.

Pusat data on-premise

Untuk beban kerja yang berjalan di pusat data lokal, buat pengalaman hybrid jika memungkinkan. AWS Direct Connect menyediakan koneksi jaringan khusus dari tempat Anda untuk AWS memungkinkan Anda menjalankan keduanya.

Pilihan lain adalah menjalankan AWS infrastruktur dan layanan di tempat menggunakan AWS Outposts. AWS Outposts adalah layanan yang dikelola sepenuhnya yang memperluas AWS infrastruktur, AWS layananAPIs, dan alat ke pusat data Anda. Infrastruktur perangkat keras yang sama yang AWS Cloud digunakan di diinstal di pusat data Anda. AWS Outposts kemudian terhubung ke yang terdekat Wilayah AWS. Anda kemudian dapat menggunakan AWS Outposts untuk mendukung beban kerja Anda yang memiliki latensi rendah atau persyaratan pemrosesan data lokal.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Gunakan beberapa Availability Zone dan Wilayah AWS. Distribusikan sumber daya dan data beban kerja ke beberapa Zona Ketersediaan atau, jika diperlukan, ke beberapa Wilayah AWS. Lokasi tersebut dapat beragam sesuai kebutuhan.
  - Layanan-layanan regional di-deploy secara permanen di seluruh Zona Ketersediaan.
    - Ini termasuk Amazon S3, Amazon DynamoDB, AWS Lambda dan (bila tidak terhubung ke a) VPC
  - Deploy kontainer Anda, instans, dan beban kerja berbasis fungsi ke beberapa Zona Ketersediaan. Gunakan penyimpanan data multi-zona, termasuk cache. Gunakan fitur Amazon EC2 Auto Scaling, penempatan ECS tugas Amazon, konfigurasi AWS Lambda fungsi saat berjalan di AndaVPC, dan ElastiCache cluster.
    - Gunakan subnet yang berada di Zona Ketersediaan terpisah saat Anda melakukan deployment grup Auto Scaling.
      - Misalnya: Mendistribusikan instans di seluruh Zona Ketersediaan
      - Memilih Wilayah dan Zona Ketersediaan
    - Gunakan parameter penempatan ECS tugas, menentukan grup subnet DB.
      - Strategi penempatan ECS tugas Amazon
    - Gunakan subnet di beberapa Availability Zone saat Anda mengonfigurasi fungsi untuk dijalankan di. VPC
      - Mengonfigurasi AWS Lambda fungsi untuk mengakses sumber daya di Amazon VPC
    - Gunakan beberapa Availability Zone dengan ElastiCache cluster.
      - Memilih Wilayah dan Zona Ketersediaan
- Jika beban kerja harus di-deploy di beberapa Wilayah, pilih strategi multi-Wilayah. Sebagian besar kebutuhan keandalan dapat dipenuhi dalam satu Wilayah AWS menggunakan strategi Multi-Availability Zone. Gunakan strategi multi-Wilayah untuk memenuhi kebutuhan bisnis, jika diperlukan.
  - AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2
    - Backup ke yang lain Wilayah AWS dapat menambahkan lapisan jaminan lain bahwa data akan tersedia saat dibutuhkan.
    - Beberapa beban kerja memiliki persyaratan berdasarkan regulasi yang mengharuskan penggunaan strategi multi-Wilayah.

- AWS Outposts Evaluasi beban kerja Anda. Jika beban kerja memerlukan latensi rendah ke pusat data on-premise Anda atau memiliki persyaratan pemrosesan data lokal. Kemudian jalankan AWS infrastruktur dan layanan di tempat menggunakan AWS Outposts
  - Apa itu AWS Outposts?
- Tentukan apakah AWS Local Zones membantu Anda memberikan layanan kepada pengguna Anda. Jika Anda memiliki persyaratan latensi rendah, lihat apakah AWS Local Zones terletak di dekat pengguna Anda. Jika iya, manfaatkan hal tersebut untuk melakukan deployment beban kerja dengan lebih dekat ke para pengguna itu.
  - Zona Lokal AWS FAQ

Sumber daya

Dokumen terkait:

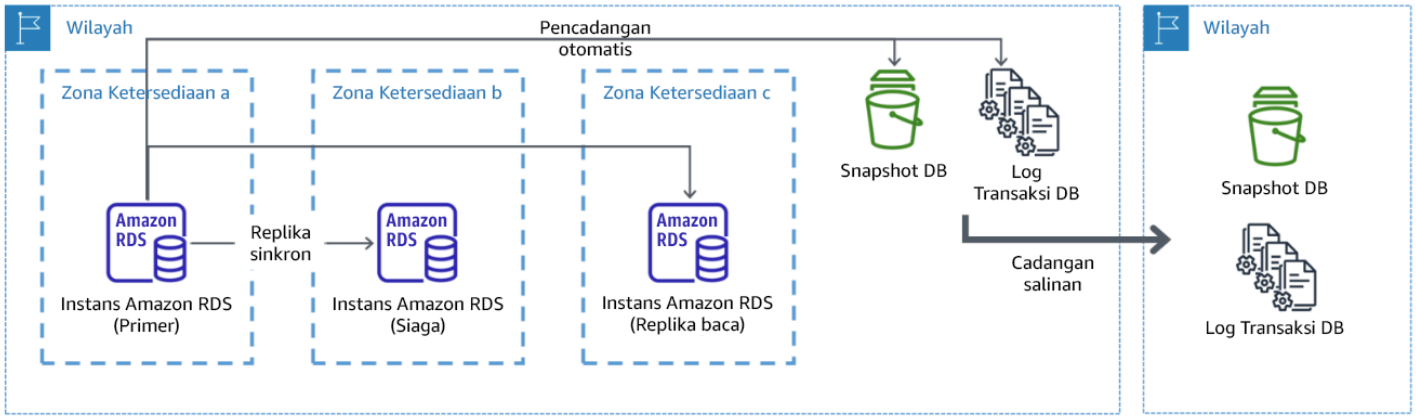
- Infrastruktur Global AWS
- Zona Lokal AWS FAQ
- Strategi penempatan ECS tugas Amazon
- Memilih Wilayah dan Zona Ketersediaan
- Misalnya: Mendistribusikan instans di seluruh Zona Ketersediaan
- Tabel Global: Replikasi Multi-Wilayah dengan DynamoDB
- Menggunakan basis data global Amazon Aurora
- Membuat Aplikasi Multi-Region dengan AWS Services seri blog
- Apa itu AWS Outposts?

Video terkait:

- AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2
- AWS re:invent 2019: Inovasi dan pengoperasian infrastruktur jaringan AWS global () NET339

REL10-BP02 Pilih lokasi yang sesuai untuk penyebaran multi-lokasi Anda

Hasil yang diinginkan: Untuk ketersediaan tinggi, selalu (bila memungkinkan) terapkan komponen beban kerja Anda ke beberapa Availability Zone ()AZs. Untuk beban kerja dengan persyaratan ketahanan yang sangat tinggi, lakukan evaluasi dengan cermat terhadap opsi-opsi untuk arsitektur multi-Wilayah.




Deployment basis data multi-AZ yang tangguh dengan pencadangan ke Wilayah AWS lainnya

Anti-pola umum:

- Memilih untuk merancang sebuah arsitektur multi-Wilayah saat arsitektur multi-AZ dapat memenuhi persyaratan.
- Tidak memperhitungkan dependensi antara komponen-komponen aplikasi jika ketahanan dan persyaratan multi-lokasi antara komponen-komponen tersebut berbeda.

Manfaat menerapkan praktik terbaik ini: Untuk ketahanan, Anda harus menggunakan pendekatan yang membangun berlapis-lapis pertahanan. Satu lapisan melindungi terhadap gangguan yang lebih kecil, lebih umum, dengan membangun arsitektur yang sangat tersedia menggunakan beberapaAZs. Lapisan pertahanan lainnya ditujukan untuk memberikan proteksi dari peristiwa-peristiwa langka seperti bencana alam yang meluas dan gangguan tingkat Wilayah. Lapisan kedua ini melibatkan arsitektur aplikasi Anda untuk menjangkau beberapa Wilayah AWS.

- Perbedaan antara ketersediaan 99,5% dan ketersediaan 99,99% adalah lebih dari 3,5 jam per bulan. Ketersediaan beban kerja yang diharapkan hanya dapat mencapai “empat sembilan” jika dalam beberapa. AZs
- Dengan menjalankan beban kerja Anda dalam beberapa kaliAZs, Anda dapat mengisolasi kesalahan dalam daya, pendinginan, dan jaringan, dan sebagian besar bencana alam seperti kebakaran dan banjir.
- Mengimplementasikan strategi multi-Wilayah untuk beban kerja dapat membantu Anda melindunginya dari bencana alam yang menjangkau dan memengaruhi wilayah geografis yang luas di suatu negara, atau kegagalan-kegagalan teknis yang mencakup seluruh Wilayah.

Perhatikan bahwa mengimplementasikan arsitektur multi-Wilayah dapat menjadi suatu hal yang sangat kompleks, dan biasanya tidak diperlukan untuk sebagian besar beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk peristiwa bencana berdasarkan gangguan atau hilangnya sebagian dari satu Availability Zone, menerapkan beban kerja yang sangat tersedia di beberapa Availability Zone dalam satu zona Wilayah AWS membantu mengurangi bencana alam dan teknis. Setiap Wilayah AWS terdiri atas beberapa Zona Ketersediaan, masing-masing zona terisolasi dari kesalahan yang terjadi di zona lain dan dipisahkan oleh jarak yang memadai. Namun demikian, untuk peristiwa bencana yang menimbulkan risiko hilangnya beberapa komponen Zona Ketersediaan, yang jaraknya cukup jauh satu sama lain, Anda harus mengimplementasikan opsi-opsi pemulihan bencana untuk memitigasi kegagalan yang dampaknya mencakup satu Wilayah secara keseluruhan. Untuk beban kerja yang memerlukan ketahanan sangat tinggi (infrastruktur yang sangat penting, aplikasi-aplikasi terkait kesehatan, infrastruktur sistem keuangan, dll.), strategi multi-Wilayah mungkin harus digunakan.

Langkah-langkah Implementasi

1. Evaluasi beban kerja Anda dan tentukan apakah kebutuhan ketahanan dapat dipenuhi dengan pendekatan Multi-AZ (tunggal Wilayah AWS), atau jika mereka memerlukan pendekatan Multi-wilayah. Mengimplementasikan sebuah arsitektur multi-Wilayah untuk memenuhi persyaratan tersebut akan menimbulkan kompleksitas tambahan, karena itu Anda harus mepertimbangkan kasus penggunaan Anda dan persyaratan-persyaratannya dengan cermat. Persyaratan ketahanan hampir selalu dapat dipenuhi dengan menggunakan satu Wilayah AWS. Pertimbangkan persyaratan-persyaratan yang mungkin diperlukan berikut ini saat menentukan apakah Anda perlu menggunakan beberapa Wilayah:
   a. Pemulihan bencana (DR): Untuk peristiwa bencana berdasarkan gangguan atau hilangnya sebagian dari satu Availability Zone, menerapkan beban kerja yang sangat tersedia di beberapa Availability Zone dalam satu Wilayah AWS membantu mengurangi bencana alam dan teknis. Untuk sebuah peristiwa bencana yang menimbulkan risiko kehilangan beberapa komponen Zona Ketersediaan, yang jaraknya cukup jauh satu sama lain, Anda harus mengimplementasikan langkah-langkah pemulihan bencana di seluruh Wilayah untuk memitigasi dampak bencana alam tersebut atau kegagalan-kegagalan teknis mencakup satu Wilayah secara keseluruhan.

b. Ketersediaan tinggi (HA): Arsitektur multi-wilayah (menggunakan beberapa AZs di setiap Wilayah) dapat digunakan untuk mencapai ketersediaan yang lebih besar dari empat 9 (> 99,99%).

c. Lokalisasi tumpukan: Saat menerapkan beban kerja ke audiens global, Anda dapat menerapkan tumpukan yang dilokalkan di berbagai tempat Wilayah AWS untuk melayani audiens di Wilayah tersebut. Pelokalan dapat mencakup bahasa, mata uang, dan jenis data yang disimpan.

d. Kedekatan dengan pengguna: Saat menerapkan beban kerja ke audiens global, Anda dapat mengurangi latensi dengan menerapkan tumpukan di Wilayah AWS dekat tempat pengguna akhir berada.

e. Residensi data: Beberapa beban kerja bergantung pada persyaratan residensi data, ketika data dari pengguna tertentu harus tetap berada dalam batasan negara tertentu. Berdasarkan peraturan yang dimaksud, Anda dapat memilih untuk menyebarkan seluruh tumpukan, atau hanya data, ke dalam Wilayah AWS batas-batas tersebut.

2. Berikut ini adalah beberapa contoh fungsionalitas multi-AZ yang disediakan oleh layanan AWS :

a. Untuk melindungi beban kerja menggunakan EC2 atauECS, gunakan Elastic Load Balancer di depan sumber daya komputasi. Kemudian, Elastic Load Balancing menyediakan solusi untuk mendeteksi instans yang ada di zona yang tidak sehat, dan kemudian merutekan lalu lintas ke zona yang sehat.

   i. Memulai Penyeimbang Beban Aplikasi

   ii. Memulai Penyeimbang Beban Jaringan

b. Dalam kasus EC2 contoh yang menjalankan off-the-shelf perangkat lunak komersial yang tidak mendukung penyeimbangan beban, Anda dapat mencapai bentuk toleransi kesalahan dengan menerapkan metodologi pemulihan bencana multi-AZ.

   i. the section called “REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan”

c. Untuk ECS tugas Amazon, gunakan layanan Anda secara merata di tiga AZs untuk mencapai keseimbangan ketersediaan dan biaya.

   i. Praktik terbaik ECS ketersediaan Amazon | Wadah

d. Untuk Amazon non-AuroraRDS, Anda dapat memilih Multi-AZ sebagai opsi konfigurasi. Setelah kegagalan instance database utama, Amazon RDS secara otomatis mempromosikan database siaga untuk menerima lalu lintas di zona ketersediaan lain. Replika baca multi-Wilayah juga dapat dibuat untuk meningkatkan ketahanan.

   i. Penerapan Amazon RDS Multi AZ

   ii. Membuat replika baca di tempat yang berbeda Wilayah AWS

3. Berikut adalah beberapa contoh fungsionalitas Multi-wilayah yang disediakan oleh AWS layanan:

   a. Untuk beban kerja Amazon S3, ketika ketersediaan multi-AZ disediakan secara otomatis oleh layanan, pertimbangkan Titik Akses Multi-Wilayah jika deployment multi-Wilayah diperlukan.

      i. Titik Akses Multi-Wilayah di Amazon S3

   b. Untuk tabel DynamoDB, ketika ketersediaan multi-AZ disediakan secara otomatis oleh layanan, Anda dapat mengonversi tabel yang ada ke tabel global dengan mudah untuk memperoleh manfaat dari beberapa wilayah.

      i. Konversi Tabel Amazon DynamoDB Wilayah Tunggal menjadi Tabel Global

   c. Jika beban kerja Anda dihadapkan oleh Application Load Balancers atau Network Load Balancers, gunakan AWS Global Accelerator untuk meningkatkan ketersediaan aplikasi Anda dengan mengarahkan lalu lintas ke beberapa wilayah yang berisi titik akhir yang sehat.

      i. Titik akhir untuk akselerator standar di AWS Global Accelerator - AWS Global Accelerator (amazon.com)

   d. Untuk aplikasi yang memanfaatkan AWS EventBridge, pertimbangkan bus lintas wilayah untuk meneruskan acara ke Wilayah lain yang Anda pilih.

      i. Mengirim dan menerima EventBridge acara Amazon antara Wilayah AWS

   e. Untuk basis data Amazon Aurora, pertimbangkan basis data global Aurora, yang menjangkau beberapa wilayah AWS . Klaster yang sudah ada juga dapat diubah untuk menambahkan Wilayah baru.

      i. Memulai basis data global Amazon Aurora

   f. Jika beban kerja Anda menyertakan AWS Key Management Service (AWS KMS) kunci enkripsi, pertimbangkan apakah kunci Multi-region sesuai untuk aplikasi Anda.

      i. Kunci Multi-Region di AWS KMS

   g. Untuk fitur AWS layanan lainnya, lihat seri blog ini tentang Membuat Aplikasi Multi-Region dengan AWS Services series

Tingkat upaya untuk Rencana Implementasi: Sedang cenderung Tinggi

Sumber daya

Dokumen terkait:

- Membuat Aplikasi Multi-Region dengan AWS Services series

- Arsitektur Pemulihan Bencana (DR) pada AWS, Bagian IV: Multi-situs Aktif/Aktif
- Infrastruktur Global AWS
- Zona Lokal AWS FAQ
- Arsitektur Disaster Recovery (DR) pada AWS, Bagian I: Strategi untuk Pemulihan di Cloud
- Pemulihan bencana di cloud tidak sama dengan biasanya
- Tabel Global: Replikasi Multi-Wilayah dengan DynamoDB

Video terkait:

- AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2
- Auth0: Arsitektur Ketersediaan Tinggi Multi-Wilayah yang Menskalakan hingga 1,5B+ Login Sebulan dengan failover otomatis

Contoh terkait:

- Arsitektur Disaster Recovery (DR) pada AWS, Bagian I: Strategi untuk Pemulihan di Cloud
- DTCCmencapai ketahanan jauh melampaui apa yang dapat mereka lakukan di tempat
- Expedia Group menggunakan arsitektur Multi-region, Multi-availability Zone dengan DNS layanan eksklusif untuk menambah ketahanan pada aplikasi
- Uber: Pemulihan Bencana untuk Kafka Multi-Wilayah
- Netflix: Aktif-Aktif untuk Ketahanan Multi-Wilayah
- Cara kami membangun Residensi Data untuk Atlassian Cloud
- Intuit TurboTax berjalan di dua Wilayah

REL10-BP03 Mengotomatiskan pemulihan untuk komponen yang dibatasi ke satu lokasi

Jika komponen beban kerja hanya dapat dijalankan di satu Zona Ketersediaan atau di pusat data on-premise, implementasikan kemampuan untuk membangun kembali beban kerja sepenuhnya dalam lingkup tujuan pemulihan yang telah ditetapkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jika praktik terbaik untuk melakukan deployment beban kerja ke beberapa lokasi tidak mungkin dilakukan karena adanya kendala teknologi, Anda harus mengimplementasikan jalur alternatif untuk

mewujudkan ketahanan. Anda harus melakukan otomatisasi terhadap kemampuan untuk membuat ulang infrastruktur yang dibutuhkan, melakukan deployment ulang aplikasi, dan membuat ulang data yang diperlukan untuk kasus ini.

Misalnya, Amazon EMR meluncurkan semua node untuk klaster tertentu di Availability Zone yang sama karena menjalankan klaster di zona yang sama meningkatkan kinerja alur pekerjaan karena memberikan tingkat akses data yang lebih tinggi. Jika komponen ini tidak dibutuhkan untuk ketahanan beban kerja, maka Anda harus mencari cara lain untuk melakukan deployment ulang klaster dan datanya. Juga untuk AmazonEMR, Anda harus menyediakan redundansi dengan cara selain menggunakan Multi-AZ. Anda dapat menyediakan beberapa simpul. Menggunakan EMRFile System (EMRFS), data dalam EMR dapat disimpan di Amazon S3, yang pada gilirannya dapat direplikasi di beberapa Availability Zone atau. Wilayah AWS

Demikian pula, untuk Amazon Redshift, secara default ia menyediakan klaster Anda di Availability Zone yang dipilih secara acak dalam Wilayah AWS yang Anda pilih. Semua simpul klaster disediakan dalam zona yang sama.

Untuk beban kerja berbasis server stateful yang diterapkan ke pusat data lokal, Anda dapat menggunakannya untuk melindungi beban kerja Anda. AWS Elastic Disaster Recovery AWS Jika Anda sudah di-host AWS, Anda dapat menggunakan Elastic Disaster Recovery untuk melindungi beban kerja Anda ke Availability Zone atau Region alternatif. Pemulihan Bencana Elastis menggunakan replikasi tingkat blok berkelanjutan ke area pentahapan yang ringan untuk menyediakan pemulihan aplikasi berbasis cloud dan on-premise yang cepat dan andal.

Langkah-langkah implementasi

1. Implementasikan pemulihan mandiri. Deploy instans dan kontainer Anda dengan menggunakan penskalaan otomatis jika memungkinkan. Jika Anda tidak dapat menggunakan penskalaan otomatis, gunakan pemulihan otomatis untuk EC2 instans atau terapkan otomatisasi penyembuhan mandiri berdasarkan Amazon EC2 atau peristiwa siklus hidup ECS kontainer.
   - Gunakan grup EC2 Auto Scaling Amazon untuk instans dan beban kerja container yang tidak memiliki persyaratan untuk satu alamat IP instans, alamat IP pribadi, alamat IP Elastis, dan metadata instans.
     - Data pengguna templat peluncuran dapat Anda gunakan untuk mengimplementasikan otomatisasi yang dapat memulihkan sebagian besar beban kerja secara mandiri.
   - Gunakan pemulihan otomatis EC2 instans Amazon untuk beban kerja yang memerlukan satu alamat ID instans, alamat IP pribadi, alamat IP elastis, dan metadata instans.

    - Pemulihan Otomatis akan mengirimkan peringatan status pemulihan ke SNS topik saat kegagalan instans terdeteksi.
- Gunakan peristiwa siklus hidup EC2 instans Amazon atau ECS peristiwa Amazon untuk mengotomatiskan penyembuhan mandiri jika penskalaan atau EC2 pemulihan otomatis tidak dapat digunakan.
    - Gunakan peristiwa untuk menginvokasi otomatisasi yang akan memulihkan komponen Anda berdasarkan proses logika yang diperlukan.
- Berikan proteksi terhadap beban kerja stateful yang terbatas pada satu lokasi dengan menggunakan AWS Elastic Disaster Recovery.

Sumber daya

Dokumen terkait:

- ECSAcara Amazon
- Kait EC2 siklus hidup Auto Scaling Amazon
- Pulihkan instans Anda.
- Penskalaan otomatis layanan
- Apa itu EC2 Auto Scaling Amazon?
- AWS Elastic Disaster Recovery

REL10-BP04 Gunakan arsitektur sekat untuk membatasi ruang lingkup dampak

Implementasikan arsitektur bulkhead (juga disebut sebagai arsitektur berbasis sel) untuk membatasi efek kegagalan dalam beban kerja hingga jumlah komponen yang terbatas.

Hasil yang diinginkan: Arsitektur berbasis sel menggunakan beberapa instans beban kerja yang terisolasi, di mana setiap instans dikenal sebagai sel. Setiap sel bersifat mandiri, tidak berbagi status dengan sel yang lain, dan menangani subset permintaan beban kerja secara keseluruhan. Hal ini dapat mengurangi potensi dampak yang mungkin ditimbulkan oleh sebuah kegagalan, seperti pembaruan perangkat lunak yang buruk, sehingga hanya berdampak pada satu sel individu dan permintaan yang diprosesnya. Jika beban kerja menggunakan 10 sel untuk melayani 100 permintaan, ketika ada satu kegagalan yang terjadi, 90% dari seluruh permintaan tidak akan terpengaruh oleh kegagalan tersebut.

Anti-pola umum:

- Membiarkan sel bertumbuh tanpa batas.
- Menerapkan pembaruan kode atau deployment ke semua sel pada waktu yang sama.
- Berbagi status atau komponen antara sel (dengan pengecualian lapisan router).
- Menambahkan bisnis yang kompleks atau mengarahkan rute logika ke lapisan router.
- Tidak meminimalkan interaksi lintas sel.

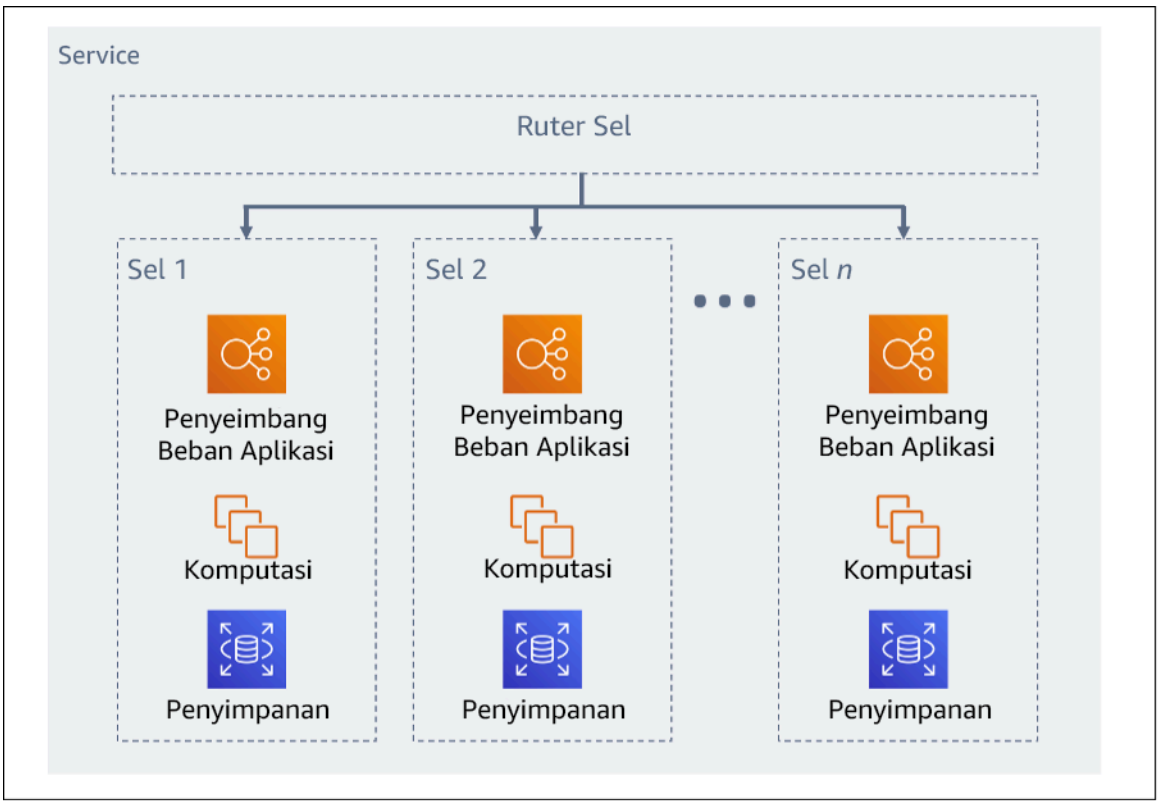
Manfaat menerapkan praktik terbaik ini: Dengan arsitektur berbasis sel, akan ada banyak jenis kegagalan umum yang dapat terkandung di dalam sel itu sendiri, yang memberikan isolasi kesalahan tambahan. Batasan-batasan kesalahan ini dapat memberikan ketahanan terhadap berbagai tipe kegagalan yang sulit ditampung, seperti deployment kode yang tidak berhasil atau permintaan yang rusak atau menginvokasi mode kegagalan tertentu (juga dikenal sebagai permintaan pil racun).

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Di sebuah kapal, bulkhead memastikan kebocoran lambung kapal hanya dibatasi dalam satu bagian lambung kapal saja. Di sistem yang kompleks, pola ini sering kali direplikasi untuk memungkinkan Anda melakukan isolasi kesalahan. Batasan-batasan isolasi kesalahan ini akan membatasi efek yang ditimbulkan kegagalan di dalam sebuah beban kerja sehingga hanya berdampak pada sejumlah komponen yang terbatas saja. Komponen-komponen yang ada di luar batasan-batasan ini tidak terpengaruh oleh kegagalan tersebut. Menggunakan beberapa batasan isolasi kesalahan, Anda dapat membatasi dampak pada beban kerja Anda. Pada AWS, pelanggan dapat menggunakan beberapa Availability Zone dan Regions untuk memberikan isolasi kesalahan, tetapi konsep isolasi kesalahan dapat diperluas ke arsitektur beban kerja Anda juga.

Beban kerja secara keseluruhan adalah sel-sel yang dipartisi oleh sebuah kunci partisi. Kunci ini harus selaras dengan grain layanan, atau cara alami beban kerja layanan dapat dibagi lebih lanjut dengan interaksi lintas sel yang minim. Contoh kunci partisi adalah ID pelanggan, ID sumber daya, atau parameter lain yang mudah diakses di sebagian besar API panggilan. Lapisan perutean sel mendistribusikan permintaan ke masing-masing sel berdasarkan kunci partisi tersebut dan memberikan satu titik akhir ke klien.




Gambar 11: Arsitektur berbasis sel

Langkah-langkah implementasi

Ketika mendesain sebuah arsitektur berbasis sel, ada beberapa pertimbangan desain yang harus Anda pikirkan:

1. Kunci partisi: Pertimbangan khusus harus diambil saat memilih kunci partisi.
   - Kunci ini harus sesuai dengan grain layanan, atau cara alami beban kerja dari sebuah layanan dapat dibagi lebih lanjut dengan interaksi lintas sel yang minimum. Contohnya adalah `customer ID` atau `resource ID`.
   - Kunci partisi harus tersedia dalam semua permintaan, baik secara langsung atau dengan cara yang dapat disimpulkan dengan pasti dan mudah berdasarkan parameter-parameter lain.
2. Pemetaan sel persisten: Layanan-layanan hulu seharusnya hanya berinteraksi dengan satu sel untuk siklus hidup sumber daya mereka.
   - Bergantung pada beban kerjanya, strategi migrasi sel mungkin perlu digunakan untuk memigrasikan data dari satu sel ke yang lain. Kemungkinan skenario ketika migrasi sel mungkin diperlukan adalah jika sumber daya atau pengguna tertentu yang ada di dalam beban kerja Anda menjadi terlalu besar dan memerlukan sebuah sel khusus.

- Sel tidak boleh berbagi status atau komponen antara sel.
- Oleh karena itu, interaksi lintas sel harus dihindari atau dijaga agar tetap minimum, karena interaksi tersebut akan menimbulkan dependensi antara sel, dan karena itu akan mengurangi peningkatan isolasi kesalahan.

3. Lapisan router: Lapisan router adalah sebuah komponen bersama antar sel, dan karena itu tidak dapat mengikuti strategi kompartementalisasi yang sama seperti sel.
   - Sebaiknya lapisan router mendistribusikan permintaan ke masing-masing sel dengan menggunakan algoritme pemetaan partisi dengan cara yang efisien secara komputasional, seperti menggabungkan fungsi hash kriptografi dan aritmetika modul untuk memetakan kunci partisi ke sel.
   - Untuk menghindari dampak yang timbul terhadap banyak sel, lapisan perutean harus dibuat sesederhana mungkin dan dapat diskalakan sehorizontal mungkin, sehingga Anda harus menghindari logika bisnis kompleks di dalam lapisan ini. Hal ini memiliki manfaat tambahan yakni mempermudah pemahaman terhadap harapan perilakunya di setiap waktu, yang memberikan kemampuan untuk diuji secara menyeluruh. Seperti yang dijelaskan oleh Colm MacCárthaigh dalam Keandalan, kerja konstan, dan secangkir kopi yang enak, desain sederhana dan pola kerja yang konstan akan menghasilkan sistem yang andal dan mengurangi anti-kerapuhan.
4. Ukuran sel: Sel harus memiliki ukuran maksimum dan tidak boleh dibiarkan tumbuh melampaui ukuran itu.
   - Ukuran maksimum harus diidentifikasi dengan melakukan pengujian menyeluruh, sampai titik rusak tercapai dan margin pengoperasian yang aman ditetapkan. Untuk detail selengkapnya tentang cara mengimplementasikan praktik pengujian, lihat REL07-BP04 Beban uji beban kerja Anda
   - Beban kerja secara keseluruhan harus bertumbuh dengan menambahkan sel-sel tambahan, sehingga beban kerja dapat diskalakan seiring dengan peningkatan permintaan.
5. Strategi Multi-AZ atau Multi-Wilayah: Beberapa lapisan ketahanan harus dimanfaatkan untuk memberikan perlindungan dari domain kegagalan yang berbeda.
   - Untuk ketahanan, Anda harus menggunakan sebuah pendekatan yang membangun berlapis-lapis pertahanan. Satu lapisan melindungi terhadap gangguan yang lebih kecil dan lebih umum dengan membangun arsitektur yang sangat tersedia menggunakan beberapaAZs. Lapisan pertahanan lainnya ditujukan untuk memberikan proteksi dari peristiwa-peristiwa langka seperti bencana alam yang meluas dan gangguan tingkat Wilayah. Lapisan kedua ini melibatkan arsitektur aplikasi Anda untuk menjangkau beberapa Wilayah AWS. Mengimplementasikan

strategi multi-Wilayah untuk beban kerja dapat membantu Anda melindunginya dari bencana alam yang menjangkau dan memengaruhi wilayah geografis yang luas di suatu negara, atau kegagalan-kegagalan teknis yang mencakup seluruh Wilayah. Perhatikan bahwa mengimplementasikan arsitektur multi-Wilayah dapat menjadi suatu hal yang sangat kompleks, dan biasanya tidak diperlukan untuk sebagian besar beban kerja. Untuk detail selengkapnya, lihat REL10-BP02 Pilih lokasi yang sesuai untuk penyebaran multi-lokasi Anda.

6. Kode deployment: Strategi deployment kode yang bertahap seharusnya lebih disarankan untuk digunakan daripada menerapkan deployment perubahan kode ke semua sel secara bersamaan.
   - Hal ini akan membantu Anda dalam meminimalkan potensi kegagalan pada beberapa sel karena deployment yang buruk atau kesalahan manusia. Untuk detail selengkapnya, silakan lihat Melakukan otomatisasi deployment secara aman dan otonom.

Sumber daya

Praktik-praktik terbaik terkait:

- REL07-BP04 Beban uji beban kerja Anda
- REL10-BP02 Pilih lokasi yang sesuai untuk penyebaran multi-lokasi Anda

Dokumen terkait:

- Keandalan, kerja konstan, dan secangkir kopi yang enak
- AWS dan Kompartementalisasi
- Isolasi beban kerja dengan menggunakan shuffle-sharding
- Melakukan otomatisasi deployment secara aman dan otonom

Video terkait:

- AWS RE: Invent 2018: Tutup Loop dan Membuka Pikiran: Cara Mengendalikan Sistem, Besar dan Kecil
- AWS Re:invent 2018: Bagaimana AWS Meminimalkan Radius Kegagalan Ledakan () ARC338
- Shuffle-sharding: AWS re:Invent 2019: Memperkenalkan Perpustakaan Pembangun Amazon () DOP328
- AWS Summit ANZ 2021 - Semuanya gagal, sepanjang waktu: Merancang untuk ketahanan

Contoh terkait:

- Lab Well-Architected: Isolasi kesalahan dengan shuffle sharding

## REL11. Bagaimana Anda mendesain beban kerja agar dapat bertahan jika terjadi kegagalan komponen?

Beban kerja dengan persyaratan ketersediaan tinggi dan waktu rata-rata yang rendah untuk pemulihan (MTTR) harus dirancang untuk ketahanan.

Praktik terbaik

- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP02 Gagal ke sumber daya yang sehat
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL11-BP04 Mengandalkan bidang data dan bukan bidang kontrol selama pemulihan
- REL11-BP05 Gunakan stabilitas statis untuk mencegah perilaku bimodal
- REL11-BP06 Kirim pemberitahuan saat acara memengaruhi ketersediaan
- REL11-BP07 Arsitek produk Anda untuk memenuhi target ketersediaan dan perjanjian tingkat layanan uptime () SLAs

### REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan

Lakukan pemantauan terus-menerus terhadap kondisi beban kerja agar Anda dan sistem otomatis Anda langsung mengetahui adanya penurunan kualitas atau kegagalan ketika muncul. Memantau indikator kinerja utama (KPIs) berdasarkan nilai bisnis.

Semua mekanisme pemulihan dan penyembuhan harus dimulai dengan kemampuan untuk mendeteksi adanya masalah secara cepat. Kegagalan teknis harus dideteksi terlebih dahulu sehingga dapat diselesaikan. Namun, ketersediaan didasarkan pada kemampuan beban kerja Anda untuk memberikan nilai bisnis, sehingga indikator kinerja utama (KPIs) yang mengukur ini perlu menjadi bagian dari strategi deteksi dan remediasi Anda.

Hasil yang diinginkan: Komponen penting dari suatu beban kerja dipantau secara independen untuk mendeteksi dan memperingatkan adanya kegagalan pada saat dan di bagian mana kegagalan tersebut terjadi.

Anti-pola umum:

- Tidak ada alarm yang dikonfigurasi, sehingga pemadaman (outage) terjadi tanpa notifikasi.
- Alarm tersedia, tetapi pada ambang batas yang tidak menyediakan waktu yang cukup untuk bereaksi.
- Metrik tidak dikumpulkan cukup sering untuk memenuhi tujuan waktu pemulihan (RTO).
- Hanya antarmuka beban kerja yang terlihat oleh pelanggan yang secara aktif dipantau.
- Hanya mengumpulkan metrik-metrik teknis, dan mengabaikan metrik-metrik fungsi bisnis.
- Tidak ada metrik yang mengukur pengalaman pengguna beban kerja.
- Terlalu banyak pemantau yang dibuat.

Manfaat menerapkan praktik terbaik ini: Pemantauan yang sesuai di semua lapisan akan memungkinkan Anda untuk menghemat waktu pemulihan karena berkurangnya waktu deteksi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Identifikasi semua beban kerja yang akan ditinjau untuk pemantauan. Setelah Anda mengidentifikasi semua komponen beban kerja yang perlu dipantau, selanjutnya Anda harus menentukan rentang waktu pemantauan. Rentang waktu pemantauan akan berdampak langsung pada seberapa cepat pemulihan dapat dimulai berdasarkan waktu yang diperlukan untuk mendeteksi sebuah kegagalan. Waktu rata-rata untuk deteksi (MTTD) adalah jumlah waktu antara kegagalan yang terjadi dan ketika operasi perbaikan dimulai. Daftar layanan harus luas dan lengkap.

Pemantauan harus mencakup semua lapisan tumpukan aplikasi termasuk aplikasi, platform, infrastruktur, dan jaringan.

Strategi pemantauan Anda harus mempertimbangkan dampak kegagalan abu-abu. Untuk detail lebih lanjut tentang kegagalan abu-abu (samar), lihat Kegagalan abu-abu di laporan resmi Pola Ketahanan Multi-AZ Tingkat Lanjut.

Langkah-langkah implementasi

- Rentang waktu pemantauan Anda bergantung pada seberapa cepat waktu pemulihan yang Anda perlukan. Waktu pemulihan Anda didorong oleh waktu yang diperlukan untuk pulih, jadi Anda harus menentukan frekuensi pengumpulan dengan menghitung waktu ini dan tujuan waktu pemulihan Anda (RTO).
- Konfigurasikan pemantauan mendetail untuk komponen dan layanan terkelola.

  - Tentukan apakah pemantauan terperinci untuk EC2 instans dan Auto Scaling diperlukan. Pemantauan mendetail menyediakan metrik rentang waktu satu menit, sedangkan pemantauan default menyediakan metrik rentang waktu lima menit.
  - Tentukan apakah pemantauan yang ditingkatkan RDS diperlukan. Pemantauan yang disempurnakan menggunakan agen pada RDS instance untuk mendapatkan informasi berguna tentang berbagai proses atau utas.
  - Tentukan persyaratan pemantauan komponen tanpa server penting untuk Lambda API, Gateway, Amazon, EKSAmazon ECS, dan semua jenis penyeimbang beban.
  - Tentukan persyaratan pemantauan komponen penyimpanan untuk Amazon S3, Amazon, FSxAmazonEFS, dan Amazon. EBS
- Buat metrik khusus untuk mengukur indikator kinerja kunci bisnis (KPIs). Beban kerja menerapkan fungsi bisnis utama, yang harus digunakan untuk membantu mengidentifikasi kapan masalah tidak langsung terjadi. KPIs
- Pantau pengalaman pengguna untuk mendeteksi terjadinya kegagalan menggunakan canary pengguna. Pengujian transaksi sintetis (juga disebut pengujian canary, tetapi bedakan dengan deployment canary) yang dapat menjalankan dan menyimulasikan perilaku pelanggan adalah salah satu proses pengujian yang paling penting. Jalankan pengujian ini secara konstan terhadap titik akhir beban kerja Anda dari beragam lokasi jarak jauh.
- Buat metrik kustom yang melacak pengalaman pengguna. Jika Anda dapat menginstrumentasikan pengalaman pelanggan, Anda dapat menentukan kapan pengalaman pelanggan mengalami degradasi.
- Atur alarm untuk mendeteksi saat ada bagian dari beban kerja Anda yang tidak berfungsi dengan baik, dan untuk menunjukkan kapan harus menerapkan penskalaan secara otomatis pada sumber daya. Alarm dapat ditampilkan secara visual di dasbor, mengirim peringatan melalui Amazon SNS atau email, dan bekerja dengan Auto Scaling untuk meningkatkan atau menurunkan sumber daya beban kerja.
- Buat dasbor untuk memvisualisasikan metrik Anda. Dasbor dapat digunakan untuk melihat tren, penyimpangan, dan indikator masalah lain yang mungkin muncul, atau menyediakan penanda untuk masalah yang ingin Anda selidiki.
- Buat pemantauan penelusuran terdistribusi untuk layanan-layanan Anda. Dengan pemantauan terdistribusi, Anda dapat memahami cara kerja aplikasi Anda dan layanan-layanan yang mendasarinya dalam melakukan identifikasi dan memecahkan akar penyebab masalah dan kesalahan kinerja.

- Buat dasbor sistem pemantauan (menggunakan CloudWatchatau X-Ray) dan pengumpulan data di Wilayah dan akun terpisah.
- Buat integrasi untuk pemantauan Amazon Health Aware untuk memungkinkan pemantauan visibilitas ke AWS sumber daya yang mungkin mengalami degradasi. Untuk beban kerja penting bisnis, solusi ini menyediakan akses ke peringatan proaktif dan real-time untuk layanan. AWS

Sumber daya

Praktik-praktik terbaik terkait:

- Definisi Ketersediaan
- REL11-BP06 Kirim Pemberitahuan saat acara memengaruhi ketersediaan

Dokumen terkait:

- Amazon CloudWatch Synthetics memungkinkan Anda membuat kenari pengguna
- Aktifkan atau Nonaktifkan Pemantauan Mendetail untuk Instans Anda
- Pemantauan yang Ditingkatkan
- Memantau Grup dan Instans Auto Scaling Anda Menggunakan Amazon CloudWatch
- Memublikasikan Metrik Kustom
- Menggunakan CloudWatch Alarm Amazon
- Menggunakan CloudWatch Dasbor
- Menggunakan CloudWatch Dasbor Lintas Akun Lintas Wilayah
- Menggunakan Penelusuran X-Ray Lintas Akun dan Lintas Wilayah
- Memahami ketersediaan
- Menerapkan Amazon Health Aware (AHA)

Video terkait:

- Memitigasi kegagalan abu-abu

Contoh terkait:

- Lab Well-Architected: Level 300: Mengimplementasikan Pemeriksaan Kondisi dan Mengelola Dependensi untuk Meningkatkan Keandalan

- Lokakarya Satu Observabilitas: Menjelajahi X-Ray

Alat terkait:

- CloudWatch
- CloudWatchX-Ray

REL11-BP02 Gagal ke sumber daya yang sehat

Jika terjadi kegagalan sumber daya, sumber daya yang sehat harus terus melayani permintaan. Untuk gangguan lokasi (seperti Availability Zone atau Wilayah AWS), pastikan bahwa Anda memiliki sistem di tempat untuk gagal ke sumber daya yang sehat di lokasi yang tidak terganggu.

Saat merancang sebuah layanan, distribusikan beban di seluruh sumber daya, Zona Ketersediaan, atau Wilayah. Oleh karena itu, kegagalan atau gangguan yang terjadi pada satu sumber daya individu dapat dimitigasi dengan mengalihkan lalu lintas ke sumber daya sehat yang masih ada. Pertimbangkan bagaimana layanan ditemukan dan dirutekan jika ada suatu kegagalan yang terjadi.

Rancang layanan Anda dengan mempertimbangkan pemulihan kesalahan. Di AWS, kami merancang layanan untuk meminimalkan waktu pemulihan dari kegagalan dan dampak pada data. Layanan-layanan kami utamanya menggunakan penyimpanan data yang mengenali permintaan hanya setelah disimpan dalam waktu lama di beberapa replika di dalam suatu Wilayah. Layanan dan sumber daya ini dibangun untuk menggunakan isolasi berbasis sel dan menggunakan isolasi kesalahan yang disediakan oleh Zona Ketersediaan. Kami banyak menggunakan otomatisasi di dalam prosedur-prosedur operasional kami. Kami juga mengoptimalkan replace-and-restart fungsionalitas kami untuk pulih dengan cepat dari gangguan.

Pola dan desain yang memungkinkan failover berbeda-beda untuk setiap layanan platform AWS . Banyak layanan terkelola AWS asli adalah beberapa Availability Zone (seperti Lambda API atau Gateway). AWS Layanan lain (seperti EC2 danEKS) memerlukan desain praktik terbaik khusus untuk mendukung failover sumber daya atau penyimpanan data secara keseluruhanAZs.

Pemantauan harus disiapkan untuk memeriksa apakah sumber daya failover sehat, melacak kemajuan sumber daya yang melakukan failover, dan memantau pemulihan proses bisnis.

Hasil yang diinginkan: Sistem mampu secara otomatis atau manual menggunakan sumber daya baru untuk pulih dari degradasi.

Anti-pola umum:

- Perencanaan kegagalan bukan bagian dari fase perencanaan dan desain.
- RTOdan RPO tidak didirikan.
- Pemantauan yang tidak memadai untuk mendeteksi sumber daya yang gagal.
- Isolasi domain kegagalan yang layak.
- Kegagalan Multi-Wilayah tidak dipertimbangkan.
- Deteksi kegagalan terlalu sensitif atau agresif saat memutuskan untuk melakukan failover.
- Tidak menguji atau memvalidasi desain failover.
- Melakukan otomatisasi pemulihan otomatis, tetapi tidak memberikan notifikasi bahwa pemulihan diperlukan.
- Kurangnya periode peredaman untuk menghindari kegagalan berulang yang terlalu cepat.

Manfaat menerapkan praktik terbaik ini: Anda dapat membangun sistem-sistem yang lebih tangguh yang mempertahankan keandalan saat mengalami kegagalan dengan melakukan degradasi secara mulus dan pulih dengan cepat.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS layanan, seperti Elastic Load Balancing dan Amazon Auto EC2 Scaling, membantu mendistribusikan beban di seluruh sumber daya dan Availability Zone. Oleh karena itu, kegagalan sumber daya individu (seperti EC2 contoh) atau penurunan Zona Ketersediaan dapat dikurangi dengan mengalihkan lalu lintas ke sumber daya yang sehat yang tersisa.

Untuk beban kerja multi-Wilayah, desainnya lebih rumit. Misalnya, replika baca lintas wilayah memungkinkan Anda menyebarkan data ke beberapa. Wilayah AWS Namun, failover masih diperlukan untuk mempromosikan replika baca ke primer kemudian mengarahkan lalu lintas Anda ke titik akhir baru. Amazon Route 53, Amazon Application Recovery Controller (ARC) CloudFront, Amazon, dan AWS Global Accelerator dapat membantu merutekan lalu lintas Wilayah AWS.

AWS layanan, seperti Amazon S3, LambdaAPI, Gateway, Amazon, Amazon, Amazon, SQS Amazon SES Pinpoint, SNS Amazon,,,,, AWS Certificate Manager atau EventBridge Amazon DynamoDBECR, secara otomatis digunakan ke beberapa Availability Zone oleh. AWS Jika terjadi kegagalan, AWS layanan ini secara otomatis mengarahkan lalu lintas ke lokasi yang sehat. Data disimpan secara redundan di beberapa Zona Ketersediaan dan tetap tersedia.

Untuk AmazonRDS, Amazon Aurora, Amazon Redshift, Amazon, atau EKS ECS Amazon, Multi-AZ adalah opsi konfigurasi. AWS dapat mengarahkan lalu lintas ke instance sehat jika failover dimulai. Tindakan failover ini dapat diambil oleh AWS atau sebagaimana disyaratkan oleh pelanggan

Untuk EC2 instans Amazon, Amazon Redshift, tugas ECS Amazon, atau pod EKS Amazon, Anda memilih Availability Zone yang akan digunakan. Untuk beberapa desain, Elastic Load Balancing menyediakan solusi untuk mendeteksi instans yang ada di zona yang tidak sehat, dan kemudian merutekan lalu lintas ke zona yang sehat. Penyeimbang Beban Elastis dapat merutekan lalu lintas ke komponen di pusat data on-premise Anda.

Untuk failover lalu lintas Multi-Wilayah, pengalihan rute dapat memanfaatkan Amazon Route 53, Amazon Application Recovery Controller, AWS Global Accelerator, Route 53 Private DNS untukVPCs, atau CloudFront untuk menyediakan cara untuk menentukan domain internet dan menetapkan kebijakan perutean, termasuk pemeriksaan kesehatan, untuk merutekan lalu lintas ke Wilayah yang sehat. AWS Global Accelerator menyediakan alamat IP statis yang bertindak sebagai titik masuk tetap ke aplikasi Anda, lalu rute ke titik akhir yang Anda pilih, menggunakan jaringan AWS global alih-alih internet untuk kinerja dan keandalan yang lebih baik. Wilayah AWS

Langkah-langkah implementasi

- Buat desain failover untuk semua aplikasi dan layanan yang sesuai. Isolasi setiap komponen arsitektur dan buat rapat desain failover RTO dan RPO untuk setiap komponen.
- Konfigurasikan lingkungan yang lebih rendah (seperti pengembangan atau pengujian) dengan semua layanan yang diharuskan memiliki rencana failover. Lakukan deployment solusi dengan menggunakan infrastruktur sebagai kode (IaC) untuk memastikan kemampuan pengulangan.
- Konfigurasikan lokasi pemulihan, misalnya Wilayah kedua, untuk mengimplementasikan dan menguji desain failover. Jika perlu, sumber daya untuk pengujian dapat dikonfigurasi secara sementara untuk membatasi biaya-biaya tambahan.
- Tentukan rencana failover mana yang diotomatisasi oleh AWS, yang dapat diotomatisasi oleh suatu DevOps proses, dan mana yang mungkin manual. Dokumentasikan dan ukur setiap layanan RTO danRPO.
- Buatlah sebuah playbook failover dan sertakan semua langkah untuk melakukan failover pada setiap sumber daya, aplikasi, dan layanan.
- Buatlah sebuah playbook failback dan sertakan semua langkah untuk melakukan failback (dengan pengaturan waktu) pada setiap sumber daya, aplikasi, dan layanan
- Buatlah sebuah rencana untuk memulai dan melatih playbook. Gunakan simulasi dan pengujian kekacauan untuk menguji langkah-langkah dan otomatisasi playbook.

- Untuk kerusakan lokasi (seperti Availability Zone atau Wilayah AWS), pastikan Anda memiliki sistem di tempat untuk gagal ke sumber daya yang sehat di lokasi yang tidak terganggu. Periksa kuota, tingkat penskalaan otomatis, dan sumber daya yang berjalan sebelum pengujian failover.

Sumber daya

Praktik terbaik Well-Architected terkait:

- REL13- Rencana untuk DR
- REL10 - Gunakan isolasi kesalahan untuk melindungi beban kerja Anda

Dokumen terkait:

- Pengaturan RTO dan RPO Target
- Failover menggunakan perutean tertimbang Route 53
- Pemulihan Bencana dengan Pengontrol Pemulihan Aplikasi Amazon
- EC2dengan autoscaling
- EC2Penerapan - Multi-AZ
- ECSPenerapan - Multi-AZ
- Beralih lalu lintas menggunakan Amazon Application Recovery Controller
- Lambda dengan Penyeimbang Beban Aplikasi dan Failover
- ACMReplikasi dan Failover
- Replikasi Penyimpanan Parameter dan Failover
- ECRreplikasi lintas wilayah dan Failover
- Konfigurasi replikasi lintas wilayah manajer rahasia
- Aktifkan replikasi lintas wilayah untuk EFS dan Failover
- EFSReplikasi Lintas Wilayah dan Failover
- Failover Jaringan
- Failover titik akhir S3 menggunakan MRAP
- Membuat replikasi lintas wilayah untuk S3
- Panduan untuk Failover Lintas Wilayah dan Kegagalan Anggun di AWS
- Failover menggunakan akselerator global multi-wilayah
- Failover dengan DRS

- Membuat Mekanisme Pemulihan Bencana Menggunakan Amazon Route 53

Contoh terkait:

- Pemulihan Bencana di AWS
- Pemulihan Bencana Elastis pada AWS

REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan

Setelah kegagalan dideteksi, gunakan kemampuan otomatis untuk melakukan tindakan perbaikan. Degradasi dapat dipulihkan secara otomatis melalui mekanisme servis internal atau memerlukan sumber daya untuk dimulai ulang atau dihapus melalui tindakan remediasi.

Untuk aplikasi yang dikelola secara mandiri dan perbaikan lintas-Wilayah, desain pemulihan dan proses perbaikan otomatis dapat ditarik dari praktik terbaik yang ada sekarang.

Kemampuan untuk memulai ulang atau menghapus sumber daya adalah alat yang penting untuk melakukan remediasi kegagalan. Salah satu praktik terbaiknya adalah membuat layanan menjadi stateless, jika memungkinkan. Praktik ini mencegah hilangnya data atau ketersediaan pada saat sumber daya dimulai ulang. Di cloud, Anda dapat (dan umumnya harus) mengganti seluruh sumber daya (misalnya, instans komputasi atau fungsi nirserver) sebagai bagian dari proses mulai ulang. Tindakan memulai ulang itu sendiri adalah cara yang mudah dan andal untuk pulih dari kegagalan. Ada berbagai jenis kegagalan yang terjadi di dalam beban kerja. Kegagalan dapat terjadi di perangkat keras, perangkat lunak, komunikasi, dan operasi.

Memulai ulang atau mencoba ulang juga berlaku untuk permintaan-permintaan jaringan. Terapkan pendekatan pemulihan yang sama terhadap peristiwa waktu habis jaringan maupun kegagalan dependensi di mana dependensi menunjukkan kesalahan. Kedua peristiwa tersebut memiliki efek yang serupa terhadap sistem, sehingga alih-alih berupaya untuk menjadikan masing-masing sebagai kasus spesial, terapkan strategi serupa berupa coba ulang terbatas dengan penundaan eksponensial dan jitter. Kemampuan untuk memulai ulang sebuah adalah mekanisme pemulihan yang disertakan dalam komputasi berorientasi pemulihan dan arsitektur klaster ketersediaan tinggi.

Hasil yang diinginkan: Tindakan otomatis dilakukan untuk meremediasi deteksi kegagalan.

Anti-pola umum:

- Menyediakan sumber daya tanpa penskalaan otomatis.
- Melakukan deployment aplikasi di instans atau kontainer secara terpisah.

- Melakukan deployment aplikasi yang tidak dapat dilakukan ke beberapa lokasi tanpa menggunakan pemulihan otomatis.
- Memulihkan secara manual aplikasi yang gagal dipulihkan oleh penskalaan otomatis dan pemulihan otomatis.
- Tidak ada otomatisasi untuk failover basis data.
- Tidak ada metode otomatis untuk mengalihkan rute lalu lintas ke titik akhir baru.
- Tidak ada replikasi penyimpanan.

Manfaat menerapkan praktik terbaik ini: Pemulihan otomatis dapat mengurangi waktu rata-rata pemulihan dan meningkatkan ketersediaan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Desain untuk Amazon EKS atau layanan Kubernetes lainnya harus mencakup replika minimum dan maksimum atau set stateful serta ukuran cluster dan grup node minimum. Mekanisme ini menyediakan jumlah minimum sumber daya pemrosesan yang tersedia secara terus-menerus sambil secara otomatis memulihkan kegagalan apa pun menggunakan bidang kontrol Kubernetes.

Pola desain yang diakses melalui penyeimbang beban menggunakan klaster komputasi harus memanfaatkan grup Auto Scaling (penskalaan otomatis). Elastic Load Balancing (ELB) secara otomatis mendistribusikan lalu lintas aplikasi yang masuk ke beberapa target dan peralatan virtual dalam satu atau beberapa Availability Zones (). AZs

Desain berbasis komputasi terklaster yang tidak menggunakan penyeimbangan beban harus dirancang ukurannya untuk kehilangan setidaknya satu simpul. Dengan begitu, layanan dapat terus berjalan dalam kapasitas yang kemungkinan lebih rendah saat layanan memulihkan simpul baru. Contoh layanan adalah Mongo, DynamoDB Accelerator, Amazon Redshift, Amazon, Cassandra, KafkaMSK, EMR -, Couchbase,, dan Amazon Service. EC2 ELK OpenSearch Banyak dari layanan-layanan ini dapat dirancang dengan fitur pemulihan otomatis tambahan. Beberapa teknologi klaster harus menghasilkan peringatan atas hilangnya sebuah simpul yang memicu alur kerja otomatis atau manual untuk membuat ulang simpul baru. Alur kerja ini dapat diotomatisasi menggunakan AWS Systems Manager untuk memulihkan masalah dengan cepat.

Amazon EventBridge dapat digunakan untuk memantau dan memfilter peristiwa seperti CloudWatch alarm atau perubahan status di AWS layanan lain. Berdasarkan informasi peristiwa, kemudian dapat memanggil, Systems Manager Automation AWS Lambda, atau target lain untuk menjalankan logika

remediasi khusus pada beban kerja Anda. EC2Auto Scaling Amazon dapat dikonfigurasi untuk memeriksa kesehatan EC2 misalnya. Jika instance dalam status apa pun selain berjalan, atau jika status sistem terganggu, Amazon EC2 Auto Scaling menganggap instance tersebut tidak sehat dan meluncurkan instance pengganti. Untuk penggantian skala besar (seperti hilangnya seluruh Zona Ketersediaan), sebaiknya gunakan stabilitas statis karena ketersediaannya yang tinggi.

Langkah-langkah implementasi

- Gunakan grup Auto Scaling (penskalaan otomatis) untuk melakukan deployment tingkatan di sebuah beban kerja. Penskalaan otomatis dapat melakukan pemulihan mandiri pada aplikasi tanpa status, dan menambahkan atau menghapus kapasitas.
- Untuk instans komputasi yang disebutkan sebelumnya, gunakan penyeimbangan beban dan pilih jenis penyeimbang beban yang sesuai.
- Pertimbangkan penyembuhan untuk AmazonRDS. Dengan instans siaga, konfigurasikan untuk failover otomatis ke instans siaga tersebut. Untuk Amazon RDS Read Replica, alur kerja otomatis diperlukan untuk membuat replika baca utama.
- Menerapkan pemulihan otomatis pada EC2 instance yang memiliki aplikasi yang digunakan yang tidak dapat digunakan di beberapa lokasi, dan dapat mentolerir reboot setelah kegagalan. Pemulihan otomatis dapat digunakan untuk mengganti perangkat keras yang mengalami kegagalan dan memulai ulang instans ketika aplikasi tidak dapat diterapkan di beberapa lokasi. Metadata instans dan alamat IP terkait disimpan, serta EBSvolume dan titik mount ke Amazon Elastic File System atau FileSystems untuk Lustre dan Windows. Dengan menggunakan AWS OpsWorks, Anda dapat mengonfigurasi penyembuhan otomatis EC2 instance di tingkat lapisan.
- Implementasikan pemulihan otomatis dengan menggunakan AWS Step Functions dan AWS Lambda ketika Anda tidak dapat menggunakan penskalaan otomatis atau pemulihan otomatis, atau ketika pemulihan otomatis gagal. Ketika Anda tidak dapat menggunakan penskalaan otomatis, dan tidak dapat menggunakan pemulihan otomatis atau pemulihan otomatis gagal, Anda dapat mengotomatiskan penyembuhan menggunakan AWS Step Functions dan. AWS Lambda
- Amazon EventBridge dapat digunakan untuk memantau dan memfilter peristiwa seperti CloudWatchalarm atau perubahan status di AWS layanan lain. Berdasarkan informasi peristiwa, layanan ini kemudian dapat menginvokasi AWS Lambda (atau target-target lainnya) untuk menjalankan logika remediasi kustom pada beban kerja Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- Definisi Ketersediaan
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan

Dokumen terkait:

- Cara Kerja AWS Auto Scaling
- Pemulihan EC2 Otomatis Amazon
- Toko Blok Elastis Amazon (AmazonEBS)
- Amazon Elastic File System (AmazonEFS)
- Apa itu Amazon FSx untuk Lustre?
- Apa itu Amazon FSx untuk Windows File Server?
- AWS OpsWorks: Menggunakan Auto Healing untuk Mengganti Instans yang Gagal
- Apa itu AWS Step Functions?
- Apa itu AWS Lambda?
- Apa itu Amazon EventBridge?
- Menggunakan CloudWatch Alarm Amazon
- RDSKegagalan Amazon
- SSM- Otomatisasi Systems Manager
- Praktik Terbaik Arsitektur Tangguh

Video terkait:

- Penyediaan dan Penskalaan OpenSearch Layanan Secara Otomatis
- Amazon RDS Failover Secara Otomatis

Contoh terkait:

- Lokakarya tentang Auto Scaling (penskalaan otomatis)
- Lokakarya Amazon RDS Failover

Alat terkait:

- CloudWatch

- CloudWatchX-Ray

REL11-BP04 Mengandalkan bidang data dan bukan bidang kontrol selama pemulihan

Bidang kontrol menyediakan administrasi yang APIs digunakan untuk membuat, membaca, dan mendeskripsikan, memperbarui, menghapus, dan mencantumkan (CRUDL) sumber daya, sementara pesawat data menangani lalu lintas day-to-day layanan. Saat mengimplementasikan respons pemulihan atau mitigasi terhadap peristiwa yang berpotensi berdampak pada ketahanan, fokuslah pada penggunaan operasi bidang kontrol dalam jumlah minim untuk memulihkan, mengubah skala, mengembalikan, memperbaiki, atau melakukan failover layanan. Tindakan bidang data harus menggantikan aktivitas apa pun selama terjadi peristiwa degradasi ini.

Misalnya, berikut ini adalah semua tindakan bidang kontrol: meluncurkan instans komputasi baru, membuat penyimpanan blok, dan mendeskripsikan layanan antrean. Saat Anda meluncurkan instans komputasi, bidang kontrol harus melakukan beberapa tugas seperti menemukan temuan host fisik dengan kapasitas, mengalokasikan antarmuka jaringan, menyiapkan volume penyimpanan blok lokal, menghasilkan kredensial, dan menambahkan aturan keamanan. Orkestrasi bidang kontrol cenderung rumit.

Hasil yang diinginkan: Ketika sumber daya memasuki keadaan terganggu, sistem mampu pulih secara otomatis atau manual dengan mengalihkan lalu lintas dari sumber daya yang terganggu ke sumber daya yang sehat.

Anti-pola umum:

- Ketergantungan pada perubahan DNS catatan untuk merutekan kembali lalu lintas.
- Ketergantungan pada operasi penskalaan bidang kontrol untuk menggantikan komponen yang terganggu karena sumber daya yang disediakan tidak memadai.
- Mengandalkan tindakan pesawat API multi-kontrol yang luas, multi layanan, untuk memulihkan kategori gangguan apa pun.

Manfaat menerapkan praktik terbaik ini: Peningkatan tingkat keberhasilan untuk remediasi otomatis dapat mengurangi waktu rata-rata pemulihan Anda dan meningkatkan ketersediaan beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang: Untuk jenis degradasi layanan tertentu, bidang kontrol terpengaruh. Ketergantungan pada penggunaan ekstensif bidang kontrol untuk remediasi dapat meningkatkan waktu pemulihan (RTO) dan waktu rata-rata untuk pemulihan (). MTTR

Panduan implementasi

Untuk membatasi tindakan bidang data, lakukan penilaian atas setiap layanan untuk menemukan tindakan-tindakan yang diperlukan untuk memulihkan layanan.

Manfaatkan Amazon Application Recovery Controller untuk menggeser DNS lalu lintas. Fitur-fitur ini terus memantau kemampuan aplikasi Anda untuk pulih dari kegagalan dan memungkinkan Anda untuk mengontrol pemulihan aplikasi Anda di beberapa Wilayah AWS, Availability Zone, dan di lokasi.

Kebijakan perutean Route 53 menggunakan bidang kontrol, jadi jangan mengandalkannya untuk pemulihan. Pesawat data Route 53 menjawab DNS pertanyaan dan melakukan serta mengevaluasi pemeriksaan kesehatan. Mereka didistribusikan secara global dan dirancang untuk perjanjian tingkat layanan ketersediaan 100% (SLA).

Manajemen APIs dan konsol Route 53 tempat Anda membuat, memperbarui, dan menghapus sumber daya Route 53 berjalan pada bidang kontrol yang dirancang untuk memprioritaskan konsistensi dan daya tahan yang kuat yang Anda butuhkan saat mengelola. DNS Untuk mencapai hal ini, bidang kontrol ditempatkan di satu Wilayah: AS Timur (Virginia Utara). Sementara kedua sistem dibangun agar sangat andal, pesawat kontrol tidak termasuk dalamSLA. Mungkin ada peristiwa langka di mana desain tangguh bidang data memungkinkannya untuk mempertahankan ketersediaan sedangkan bidang kontrol tidak. Untuk mekanisme failover dan pemulihan bencana, Anda harus menggunakan fungsi bidang data untuk memberikan keandalan yang sebaik mungkin.

Rancang infrastruktur komputasi Anda agar stabil secara statis agar tidak menggunakan bidang kontrol selama insiden. Misalnya, jika Anda menggunakan EC2 instans Amazon, hindari penyediaan instans baru secara manual atau instruksikan Grup Auto Scaling untuk menambahkan instance sebagai respons. Untuk tingkat ketahanan tertinggi, berikan kapasitas yang cukup di klaster yang digunakan untuk failover. Jika ambang kapasitas ini harus dibatasi, tetapkan throttle pada keseluruhan end-to-end sistem untuk membatasi lalu lintas total yang mencapai set sumber daya terbatas.

Untuk layanan seperti Amazon DynamoDB, API Amazon Gateway, load balancer, dan tanpa server AWS Lambda , menggunakan layanan tersebut memanfaatkan bidang data. Namun, membuat fungsi baru, penyeimbang beban, API gateway, atau tabel DynamoDB adalah tindakan bidang kontrol dan harus diselesaikan sebelum degradasi sebagai persiapan untuk acara dan latihan tindakan failover. Untuk AmazonRDS, tindakan bidang data memungkinkan akses ke data.

Untuk informasi selengkapnya tentang bidang data, pesawat kontrol, dan cara AWS membangun layanan untuk memenuhi target ketersediaan tinggi, lihat Stabilitas statis menggunakan Availability Zones.

Pahami operasi mana yang ada di bidang data dan mana yang ada di bidang kontrol.

Langkah-langkah implementasi

Untuk setiap beban kerja yang perlu dipulihkan setelah terjadinya peristiwa degradasi, lakukan evaluasi terhadap runbook failover, desain ketersediaan tinggi, desain perbaikan otomatis, atau rencana pemulihan sumber daya HA. Identifikasi setiap tindakan yang mungkin dianggap sebagai tindakan bidang kontrol.

Pertimbangkan mengubah tindakan kontrol ke tindakan bidang data:

- Auto Scaling (bidang kontrol) ke sumber daya EC2 Amazon yang telah diskalakan sebelumnya (bidang data)
- Penskalaan EC2 instans Amazon (bidang kontrol) ke AWS Lambda penskalaan (bidang data)
- Nilai desain apa pun menggunakan Kubernetes dan sifat tindakan bidang kontrol. Menambahkan pod adalah tindakan bidang data di Kubernetes. Tindakan harus dibatasi ke penambahan pod dan bukan ke penambahan simpul. Menggunakan simpul yang disediakan secara berlebihan adalah metode yang lebih disukai untuk membatasi tindakan bidang kontrol

Pertimbangkan pendekatan alternatif yang memungkinkan tindakan-tindakan bidang data untuk memengaruhi remediasi yang sama.

- Rute 53 Rekam perubahan (bidang kontrol) atau Pengontrol Pemulihan Aplikasi Amazon (bidang data)
- Route 53 Pemeriksaan kondisi untuk pembaruan yang lebih otomatis

Pertimbangkan beberapa layanan di Wilayah sekunder, jika layanan sangat penting, untuk memungkinkan lebih banyak tindakan bidang kontrol dan bidang data di Wilayah yang tidak terdampak.

- Amazon EC2 Auto Scaling atau Amazon EKS di Wilayah utama dibandingkan dengan Amazon Auto EC2 Scaling atau EKS Amazon di Wilayah sekunder dan merutekan lalu lintas ke Wilayah sekunder (aksi bidang kontrol)
- Membuat replika baca di primer sekunder atau mencoba tindakan yang sama di Wilayah primer (tindakan bidang kontrol)

Sumber daya

Praktik-praktik terbaik terkait:

- Definisi Ketersediaan
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan

Dokumen terkait:

- APNMitra: mitra yang dapat membantu otomatisasi toleransi kesalahan Anda
- AWS Marketplace: produk yang dapat digunakan untuk toleransi kesalahan
- Amazon Builders' Library: Menghindari kelebihan beban dalam sistem terdistribusi dengan mengontrol layanan yang lebih kecil
- Amazon API DynamoDB (bidang kontrol dan bidang data)
- AWS Lambda Eksekusi (dibagi menjadi bidang kontrol dan bidang data)
- AWS Elemental MediaStore Pesawat Data
- Membangun aplikasi yang sangat tangguh menggunakan Amazon Application Recovery Controller, Bagian 1: Tumpukan Wilayah Tunggal
- Membangun aplikasi yang sangat tangguh menggunakan Amazon Application Recovery Controller, Bagian 2: Tumpukan Multi-Wilayah
- Membuat Mekanisme Pemulihan Bencana Menggunakan Amazon Route 53
- Apa itu Pengontrol Pemulihan Aplikasi Amazon
- Bidang kontrol dan bidang data Kubernetes

Video terkait:

- Kembali ke Dasar - Menggunakan Stabilitas Statis
- Membangun beban kerja multi-lokasi yang tangguh menggunakan layanan global AWS

Contoh terkait:

- Memperkenalkan Pengontrol Pemulihan Aplikasi Amazon
- Amazon Builders' Library: Menghindari kelebihan beban dalam sistem terdistribusi dengan mengontrol layanan yang lebih kecil

- Membangun aplikasi yang sangat tangguh menggunakan Amazon Application Recovery Controller, Bagian 1: Tumpukan Wilayah Tunggal
- Membangun aplikasi yang sangat tangguh menggunakan Amazon Application Recovery Controller, Bagian 2: Tumpukan Multi-Wilayah
- Stabilitas statis menggunakan Zona Ketersediaan

Alat terkait:

- Amazon CloudWatch
- AWS X-Ray

REL11-BP05 Gunakan stabilitas statis untuk mencegah perilaku bimodal

Beban kerja harus stabil secara statis dan hanya beroperasi dalam mode normal tunggal. Perilaku bimodal adalah ketika beban kerja Anda menunjukkan perilaku yang berbeda dalam mode normal dan mode kegagalan.

Misalnya, Anda mungkin mencoba untuk melakukan pemulihan dari kegagalan Zona Ketersediaan dengan meluncurkan instans baru di Zona Ketersediaan yang berbeda. Hal ini dapat menyebabkan respons bimodal saat berada dalam mode kegagalan. Sebagai gantinya, Anda harus membangun beban kerja yang stabil secara statis dan beroperasi dalam satu mode saja. Dalam contoh ini, instans-instans tersebut seharusnya telah disediakan di Zona Ketersediaan kedua sebelum terjadinya kegagalan. Desain stabilitas statis ini memastikan beban kerja hanya beroperasi dalam satu mode tunggal.

Hasil yang diinginkan: Beban kerja tidak menunjukkan perilaku bimodal selama mode normal dan mode kegagalan.

Anti-pola umum:

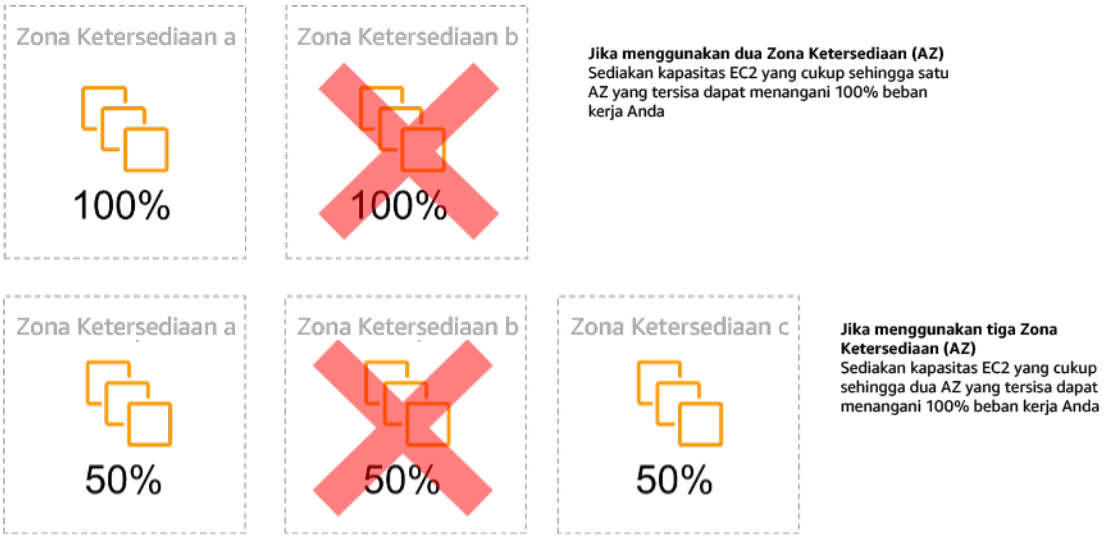
- Asumsikan bahwa sumber daya selalu dapat disediakan terlepas dari ruang lingkup kegagalannya.
- Mencoba memperoleh sumber daya secara dinamis selama terjadi kegagalan.
- Tidak menyediakan sumber daya yang memadai di seluruh zona atau Wilayah sampai terjadi kegagalan.
- Mempertimbangkan desain stabil statis hanya untuk sumber daya komputasi.

Manfaat menerapkan praktik terbaik ini: Beban kerja yang berjalan dengan desain yang stabil secara statis mampu memiliki hasil-hasil yang dapat diprediksi selama terjadi peristiwa normal dan kegagalan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Perilaku bimodal terjadi ketika beban kerja Anda menunjukkan perilaku yang berbeda dalam mode normal dan mode gagal, (misalnya, mengandalkan peluncuran instans baru jika Zona Ketersediaan gagal). Contoh perilaku bimodal adalah ketika EC2 desain Amazon yang stabil menyediakan cukup instance di setiap Availability Zone untuk menangani beban beban kerja jika satu AZ dihapus. Penyeimbang Beban Elastis atau kesehatan Amazon Route 53 akan memeriksa untuk mengalihkan beban dari instans yang terganggu. Setelah lalu lintas bergeser, gunakan AWS Auto Scaling untuk mengganti instance secara asinkron dari zona gagal dan meluncurkannya di zona sehat. Stabilitas statis untuk penerapan komputasi (seperti EC2 instance atau kontainer) menghasilkan keandalan tertinggi.




Stabilitas statis EC2 instance di seluruh Availability Zone

Ini harus ditimbang berdasarkan biaya untuk model ini serta nilai bisnis untuk memelihara beban kerja berdasarkan semua kasus ketahanan. Menyediakan kapasitas komputasi yang lebih sedikit dan mengandalkan peluncuran instans baru apabila terjadi kegagalan memang lebih murah, tetapi untuk kegagalan-kegagalan berskala besar (seperti gangguan pada Zona Ketersediaan atau Regional), pendekatan ini kurang efektif karena bergantung pada bidang operasional serta sumber daya yang tersedia di zona atau Wilayah yang tidak terdampak.

Solusi Anda harus mengukur keandalan berdasarkan kebutuhan biaya untuk beban kerja Anda. Arsitektur stabilitas statis berlaku untuk berbagai arsitektur termasuk instans komputasi yang tersebar di Availability Zone, desain replika baca database, desain cluster Kubernetes (AmazonEKS), dan arsitektur failover Multi-region.

Anda juga dapat menerapkan desain yang lebih stabil secara statis dengan menggunakan lebih banyak sumber daya di setiap zona. Dengan menggunakan lebih banyak zona, Anda mengurangi jumlah komputasi tambahan yang Anda perlukan untuk stabilitas statis.

Contoh perilaku bimodal adalah batas waktu jaringan yang dapat menyebabkan sebuah sistem untuk mencoba melakukan refresh status konfigurasi seluruh sistem. Ini akan menambahkan beban yang tak terduga ke komponen lain dan dapat menyebabkan komponen tersebut mengalami kegagalan, sehingga menimbulkan konsekuensi-konsekuensi lain yang tak diharapkan. Putaran umpan balik negatif ini memengaruhi ketersediaan beban kerja Anda. Sebagai gantinya, Anda dapat membangun sistem yang stabil secara statis dan beroperasi dalam satu mode saja. Desain yang stabil secara statis akan melakukan tugas yang konstan, dan selalu menyegarkan status konfigurasi dengan irama yang teratur. Ketika sebuah panggilan gagal, beban kerja akan menggunakan nilai yang sebelumnya di-cache, dan memulai alarm.

Contoh perilaku bimodal lainnya adalah memperbolehkan klien untuk melewati cache beban kerja Anda ketika kegagalan terjadi. Ini mungkin terlihat seperti solusi yang mengakomodasi kebutuhan klien, tetapi hal ini secara signifikan dapat mengubah permintaan di beban kerja Anda dan kemungkinan akan mengakibatkan kegagalan.

Lakukan penilaian beban kerja kritis untuk menentukan beban kerja apa yang memerlukan jenis desain ketahanan ini. Untuk beban kerja yang dianggap kritis, setiap komponen aplikasi harus ditinjau. Contoh jenis layanan yang memerlukan evaluasi stabilitas statis adalah:

- Hitung: AmazonEC2, EKS -EC2, ECS -EC2, EMR - EC2
- Basis Data: Amazon Redshift, Amazon, RDS Amazon Aurora
- Penyimpanan: Amazon S3 (Zona Tunggal), Amazon (tunggangan), Amazon EFS FSx (tunggangan)
- Penyeimbang beban: Menggunakan desain tertentu

Langkah-langkah implementasi

- Bangun sistem yang stabil secara statis dan hanya beroperasi dalam satu mode saja. Dalam hal ini, sediakan cukup instans di setiap Zona Ketersediaan atau Wilayah untuk menangani kapasitas

beban kerja jika satu Zona Ketersediaan atau Wilayah dihapus. Berbagai layanan dapat digunakan untuk perutean ke sumber daya yang sehat, seperti:

- DNSPerutean Lintas Wilayah
- MRAPPerutean Amazon S3 MultiRegion
- AWS Global Accelerator
- Pengontrol Pemulihan Aplikasi Amazon

- Konfigurasikan replika baca basis data untuk memperhitungkan hilangnya satu instans utama atau replika baca. Jika lalu lintas dilayani oleh replika baca, maka kuantitas di setiap Zona Ketersediaan dan setiap Wilayah harus sama dengan kebutuhan keseluruhan jika terjadi kegagalan zona atau Wilayah.
- Konfigurasikan data kritis di dalam penyimpanan Amazon S3 yang dirancang agar stabil secara statis untuk data yang disimpan jika terjadi kegagalan Zona Ketersediaan. Jika kelas penyimpanan Amazon S3 One Zone-IA digunakan, ini tidak boleh dianggap stabil secara statis, karena hilangnya zona tersebut akan meminimalkan akses ke data yang disimpan.
- Penyeimbang beban terkadang dikonfigurasi secara salah atau secara bawaan untuk melayani satu Zona Ketersediaan tertentu. Dalam hal ini, desain yang stabil secara statis mungkin untuk menyebarkan beban kerja di beberapa AZs dalam desain yang lebih kompleks. Desain asli dapat digunakan untuk mengurangi lalu lintas antar zona karena alasan keamanan, latensi, atau biaya.

Sumber daya

Praktik terbaik Well-Architected terkait:

- Definisi Ketersediaan
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP04 Mengandalkan bidang data dan bukan bidang kontrol selama pemulihan

Dokumen terkait:

- Meminimalkan Ketergantungan dalam Rencana Pemulihan Bencana
- Amazon Builders' Library: Stabilitas statis menggunakan Zona Ketersediaan
- Batas Isolasi Kesalahan
- Stabilitas statis menggunakan Zona Ketersediaan
- Multi-Zona RDS

- Meminimalkan Ketergantungan dalam Rencana Pemulihan Bencana
- DNSPerutean Lintas Wilayah
- MRAPPerutean Amazon S3 MultiRegion
- AWS Global Accelerator
- Pengontrol Pemulihan Aplikasi Amazon
- Amazon S3 Zona Tunggal
- Penyeimbangan Beban Lintas Zona

Video terkait:

- Stabilitas statis di AWS: AWS re:Invent 2019: Memperkenalkan Amazon Builders' Library () DOP328

REL11-BP06 Kirim pemberitahuan saat acara memengaruhi ketersediaan

Notifikasi dikirimkan setelah pelanggaran ambang batas terdeteksi, bahkan apabila peristiwa yang menyebabkan masalah tersebut sudah diatasi secara otomatis.

Pemulihan otomatis menjadikan beban kerja Anda andal. Namun demikian, kemampuan ini juga menyembunyikan masalah dasar yang perlu diatasi. Implementasikan pemantauan peristiwa yang baik agar Anda dapat mendeteksi setiap pola masalah, termasuk masalah-masalah yang ditangani oleh pemulihan otomatis, sehingga Anda dapat mengatasi akar penyebab masalahnya.

Sistem yang tangguh dirancang sedemikian rupa sehingga setiap terjadi peristiwa degradasi langsung dikomunikasikan kepada tim yang tepat. Notifikasi ini harus dikirim melalui satu atau banyak saluran komunikasi.

Hasil yang diinginkan: Peringatan segera dikirim ke tim operasi ketika ambang batas dilanggar, seperti tingkat kesalahan, latensi, atau metrik indikator kinerja kunci penting lainnya (KPI), sehingga masalah ini diselesaikan sesegera mungkin dan dampak pengguna dihindari atau diminimalkan.

Anti-pola umum:

- Mengirimkan terlalu banyak alarm.
- Mengirimkan alarm yang tidak dapat ditindaklanjuti.
- Mengatur ambang alarm terlalu tinggi (terlalu sensitif) atau terlalu rendah (kurang sensitif).
- Tidak mengirimkan alarm untuk dependensi eksternal.

- Tidak mempertimbangkan kegagalan abu-abu saat merancang pemantauan dan alarm.
- Melakukan otomatisasi pemulihan, tetapi tidak memberikan notifikasi kepada tim yang tepat bahwa pemulihan diperlukan.

Manfaat membangun praktik terbaik ini: Pemberitahuan pemulihan membuat tim operasional dan bisnis sadar akan degradasi layanan sehingga mereka dapat segera bereaksi untuk meminimalkan waktu rata-rata untuk mendeteksi (MTTD) dan waktu rata-rata untuk memperbaiki (MTTR). Notifikasi peristiwa pemulihan juga menjamin bahwa Anda tidak mengabaikan masalah yang jarang terjadi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang. Kegagalan mengimplementasikan mekanisme pemantauan dan notifikasi peristiwa secara tepat dapat mengakibatkan terjadinya kegagalan dalam mendeteksi pola masalah, termasuk masalah yang ditangani oleh pemulihan otomatis. Sebuah tim hanya akan menyadari adanya degradasi sistem ketika pengguna menghubungi layanan pelanggan atau secara kebetulan.

Panduan implementasi

Saat menetapkan strategi pemantauan, alarm yang dipicu adalah sebuah peristiwa umum. Peristiwa ini kemungkinan berisi pengidentifikasi untuk alarm, status alarm (seperti `IN ALARM` dan `OK`), dan detail tentang apa yang memicunya. Dalam banyak kasus, sebuah peristiwa alarm seharusnya dideteksi dan email notifikasi dikirimkan. Ini adalah contoh tindakan pada alarm. Notifikasi alarm sangat penting dalam hal observabilitas karena notifikasi ini memberi tahu orang yang tepat bahwa ada masalah. Namun demikian, ketika tindakan terhadap peristiwa sudah matang di dalam solusi observabilitas Anda, tindakan tersebut dapat secara otomatis memperbaiki masalah tanpa memerlukan campur tangan manusia.

Setelah alarm KPI pemantauan ditetapkan, peringatan harus dikirim ke tim yang sesuai ketika ambang batas terlampaui. Peringatan tersebut juga dapat digunakan untuk memicu proses otomatis yang akan mencoba memperbaiki degradasi.

Untuk pemantauan ambang batas yang lebih kompleks, alarm gabungan harus dipertimbangkan. Alarm komposit menggunakan sejumlah alarm KPI pemantauan untuk membuat peringatan berdasarkan logika bisnis operasional. CloudWatchAlarm dapat dikonfigurasi untuk mengirim email, atau untuk mencatat insiden di sistem pelacakan insiden pihak ketiga menggunakan SNS integrasi Amazon atau Amazon. EventBridge

Langkah-langkah implementasi

Buat berbagai jenis alarm berdasarkan cara yang digunakan untuk memantau beban kerja, seperti:

- Alarm aplikasi digunakan untuk mendeteksi ketika ada bagian dari beban kerja Anda yang tidak berfungsi dengan baik.
- Alarm infrastruktur menunjukkan kapan Anda harus menskalakan sumber daya. Alarm dapat ditampilkan secara visual di dasbor, mengirim peringatan melalui Amazon SNS atau email, dan bekerja dengan Auto Scaling untuk menskalakan sumber daya beban kerja masuk atau keluar.
- Alarm statis dapat dibuat untuk memantau ketika sebuah metrik melanggar ambang batas statis selama periode evaluasi tertentu.
- Alarm gabungan dapat memperhitungkan alarm-alarm kompleks dari berbagai sumber.
- Setelah alarm dibuat, buatlas peristiwa-peristiwa notifikasi yang sesuai. Anda dapat langsung memanggil Amazon SNS API untuk mengirim pemberitahuan dan menautkan otomatisasi apa pun untuk remediasi atau komunikasi.
- Integrasikan pemantauan Amazon Health Aware untuk memungkinkan pemantauan visibilitas ke AWS sumber daya yang mungkin mengalami degradasi. Untuk beban kerja penting bisnis, solusi ini menyediakan akses ke peringatan proaktif dan real-time untuk layanan. AWS

Sumber daya

Praktik terbaik Well-Architected terkait:

- Definisi Ketersediaan

Dokumen terkait:

- Membuat CloudWatch Alarm Berdasarkan Ambang Statis
- Apa itu Amazon EventBridge?
- Apa itu Amazon Simple Notification Service?
- Memublikasikan Metrik Kustom
- Menggunakan CloudWatch Alarm Amazon
- Sadar Kesehatan Amazon (AHA)
- Pengaturan CloudWatch Alarm komposit
- Apa yang baru di AWS Observability at re:Invent 2022

Alat terkait:

- CloudWatch

- CloudWatchX-Ray

REL11-BP07 Arsitek produk Anda untuk memenuhi target ketersediaan dan perjanjian tingkat layanan uptime () SLAs

Arsitek produk Anda untuk memenuhi target ketersediaan dan perjanjian tingkat layanan uptime (SLAs). Jika Anda mempublikasikan atau secara pribadi menyetujui target ketersediaan atau waktu aktifSLAs, verifikasi bahwa arsitektur dan proses operasional Anda dirancang untuk mendukungnya.

Hasil yang diinginkan: Setiap aplikasi memiliki target yang ditentukan untuk ketersediaan dan SLA untuk metrik kinerja, yang dapat dipantau dan dipelihara untuk memenuhi hasil bisnis.

Anti-pola umum:

- Merancang dan menerapkan beban kerja tanpa menyetel apa pun. SLAs
- SLAmetrik ditetapkan terlalu tinggi tanpa alasan atau persyaratan bisnis.
- Pengaturan SLAs tanpa memperhitungkan dependensi dan dasarnya. SLA
- Desain aplikasi dibuat tanpa mempertimbangkan Model Tanggung Jawab Bersama untuk Ketangguhan.

Manfaat menerapkan praktik terbaik ini: Merancang aplikasi berdasarkan target ketahanan utama akan membantu Anda dalam memenuhi tujuan bisnis dan harapan pelanggan. Tujuan-tujuan ini membantu mendorong proses desain aplikasi yang mengevaluasi berbagai macam teknologi dan mempertimbangkan beragam kompromi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Desain aplikasi harus memperhitungkan berbagai rangkaian persyaratan yang didapatkan dari tujuan bisnis, operasional, dan finansial. Dalam persyaratan operasional, beban kerja harus memiliki target-target metrik ketangguhan tertentu sehingga dapat dipantau dan dukung dengan sesuai. Metrik-metrik ketangguhan tidak boleh ditetapkan atau didapatkan setelah melakukan deployment beban kerja. Metrik-metrik ketangguhan tersebut harus ditetapkan selama fase desain dan membantu memandu berbagai keputusan dan kompromi.

- Setiap beban kerja harus memiliki rangkaian metrik-metrik ketangguhannya sendiri. Metrik-metrik tersebut mungkin berbeda dari aplikasi bisnis yang lain.

- Mengurangi dependensi dapat memiliki dampak positif pada ketersediaan. Setiap beban kerja harus mempertimbangkan dependensi dan mereka. SLAs Secara umum, pilihlah dependensi dengan target ketersediaan yang setara dengan atau lebih besar dari target beban kerja Anda.
- Pertimbangkan desain yang mengunakan penggabungan longgar sehingga beban kerja Anda dapat beroperasi dengan benar meskipun ada gangguan dependensi, apabila mungkin.
- Kurangi dependensi bidang kontrol, terutama selama pemulihan atau degradasi. Evaluasi desain yang secara statis stabil untuk beban kerja yang penting bagi misi. Gunakan penghematan sumber daya untuk meningkatkan ketersediaan dependensi tersebut di sebuah beban kerja.
- Observabilitas dan instrumentasi sangat penting untuk dicapai SLAs dengan mengurangi Mean Time to Detection (MTTD) dan Mean Time to Repair (MTTR).
- Kegagalan yang lebih jarang (lebih lamaMTBF), waktu deteksi kegagalan yang lebih pendek (lebih pendekMTTD), dan waktu perbaikan yang lebih pendek (lebih pendekMTTR) adalah tiga faktor yang digunakan untuk meningkatkan ketersediaan dalam sistem terdistribusi.
- Menetapkan dan memenuhi metrik-metrik ketangguhan untuk beban kerja merupakan fondasi dari desain yang efektif. Desain tersebut harus memperhitungkan berbagai kompromi terkait kompleksitas desain, dependensi layanan, performa, penskalaan, dan biaya.

Langkah-langkah implementasi

- Tinjau dan dokumentasikan desain beban kerja sambil mempertimbangkan pertanyaan-pertanyaan berikut:
  - Di mana bidang kontrol digunakan di beban kerja?
  - Bagaimana beban kerja mengimplementasikan toleransi kesalahan?
  - Apa saja pola desain untuk penskalaan, penskalaan otomatis, redundansi, dan komponen dengan ketersediaan tinggi?
  - Apa saja persyaratan untuk ketersediaan dan konsistensi data?
  - Apakah ada pertimbangan untuk penghematan sumber daya atau stabilitas statis sumber daya?
  - Apa saja dependensi layanan?
- Tentukan SLA metrik berdasarkan arsitektur beban kerja saat bekerja dengan pemangku kepentingan. Pertimbangkan semua dependensi yang digunakan oleh beban kerja. SLAs
- Setelah SLA target ditetapkan, optimalkan arsitektur untuk memenuhiSLA.
- Setelah desain diatur yang akan memenuhiSLA, menerapkan perubahan operasional, otomatisasi proses, dan runbook yang juga akan fokus pada pengurangan MTTD danMTTR.

- Setelah digunakan, pantau dan laporkan. SLA

Sumber daya

Praktik-praktik terbaik terkait:

- REL03-BP01 Pilih cara mengelompokkan beban kerja Anda
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL11-BP01 Memantau semua komponen beban kerja untuk mendeteksi kegagalan
- REL11-BP03 Mengotomatiskan penyembuhan pada semua lapisan
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan
- REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data
- Memahami kesehatan beban kerja

Dokumen terkait:

- Ketersediaan dengan redundansi
- Pilar keandalan - Ketersediaan
- Mengukur ketersediaan
- AWS Batas Isolasi Kesalahan
- Model Tanggung Jawab Bersama untuk Ketangguhan
- Stabilitas statis menggunakan Zona Ketersediaan
- AWS Perjanjian Tingkat Layanan (SLAs)
- Panduan untuk Arsitektur Berbasis Sel pada AWS
- AWS infrastruktur
- Laporan resmi Pola Ketangguhan Multi-AZ Lanjutan

Layanan terkait:

- Amazon CloudWatch
- AWS Config
- AWS Trusted Advisor

## REL12. Bagaimana cara menguji keandalan?

Setelah Anda mendesain beban kerja Anda agar tangguh terhadap tekanan produksi, pengujian adalah satu-satunya cara untuk memverifikasi bahwa beban kerja akan beroperasi sesuai desain, dan memberikan ketangguhan yang Anda harapkan.

Praktik terbaik

- REL12-BP01 Gunakan pedoman untuk menyelidiki kegagalan
- REL12-BP02 Lakukan analisis pasca-insiden
- REL12-BP03 Uji persyaratan fungsional
- REL12-BP04 Uji penskalaan dan persyaratan kinerja
- REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan
- REL12-BP06 Lakukan hari permainan secara teratur

### REL12-BP01 Gunakan pedoman untuk menyelidiki kegagalan

Dokumentasikan proses penyelidikan di dalam playbook agar dapat memberikan respons yang cepat dan konsisten terhadap skenario kegagalan yang tidak benar-benar dipahami. Playbook adalah langkah-langkah yang telah ditetapkan di awal untuk mengidentifikasi faktor yang menyebabkan skenario kegagalan. Hasil dari setiap langkah proses digunakan untuk menentukan langkah berikutnya yang harus diambil sampai masalah teridentifikasi atau dieskalasi.

Palybook ini adalah perencanaan proaktif yang harus Anda lakukan, agar Anda dapat mengambil tindakan reaktif secara efektif. Ketika skenario kegagalan yang tidak tercakup dalam playbook dialami di lingkungan produksi, tangani masalah terlebih dahulu (put our the fire). Lalu lihat kembali langkah-langkah yang telah Anda ambil untuk mengatasi masalah tersebut dan gunakan itu semua untuk menambahkan entri baru dalam playbook.

Perhatikan, playbook digunakan untuk merespons insiden tertentu, sedangkan runbook digunakan untuk mencapai hasil tertentu. Sering kali, runbook digunakan untuk untuk aktivitas rutin, dan playbook digunakan untuk merespons peristiwa non-rutin.

Anti-pola umum:

- Berencana untuk melakukan deployment beban kerja tanpa mengetahui proses untuk mendiagnosis masalah atau merespons insiden.
- Keputusan yang tidak direncanakan tentang sistem mana saja yang dikumpulkan log dan metriknya saat menyelidiki peristiwa.

- Tidak mempertahankan metrik dan peristiwa cukup lama agar dapat mengambil data.

Manfaat menerapkan praktik terbaik ini: Merekam playbook akan memastikan bahwa proses dapat diikuti secara konsisten. Melakukan kodifikasi pada playbook dapat membatasi munculnya kesalahan dari aktivitas manual. Melakukan otomatisasi pada playbook dapat menghemat waktu respons peristiwa dengan menghilangkan keharusan campur tangan anggota tim atau memberikan informasi tambahan ketika campur tangan mereka dimulai.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Gunakan playbook untuk mengidentifikasi masalah. Playbook adalah proses-proses yang didokumentasikan untuk menyelidiki masalah. Dokumentasikan proses penyelidikan di playbook agar dapat memberikan respons yang cepat dan konsisten terhadap skenario kegagalan. Buku pedoman harus memuat informasi dan panduan yang dapat digunakan oleh orang yang cukup terampil untuk mengumpulkan informasi, mengidentifikasi potensi sumber kegagalan, mengisolasi kesalahan, dan menentukan faktor penyebabnya (lakukan analisis pasca-insiden).
  - Implementasikan playbook sebagai kode. Jalankan operasi sebagai kode dengan membuat skrip playbook Anda untuk memastikan konsistensi dan mengurangi kesalahan yang disebabkan proses manual. Playbook dapat terdiri dari beberapa skrip sesuai dengan banyaknya langkah yang diperlukan untuk mengidentifikasi faktor-faktor penyebab masalah. Aktivitas runbook dapat diinvokasi atau dijalankan sebagai bagian dari aktivitas buku pedoman, atau mempercepat eksekusi buku pedoman untuk merespons peristiwa yang teridentifikasi.
    - Otomatiskan pedoman operasional Anda dengan Systems Manager AWS
    - AWS Systems Manager Jalankan Perintah
    - AWS Systems Manager Automation
    - Apa itu AWS Lambda?
    - Apa itu Amazon EventBridge?
    - Menggunakan CloudWatch Alarm Amazon

Sumber daya

Dokumen terkait:

- AWS Otomatisasi Systems Manager

- AWS Systems Manager Jalankan Perintah
- Otomatiskan pedoman operasional Anda dengan Systems Manager AWS
- Menggunakan CloudWatch Alarm Amazon
- Menggunakan Canaries (Amazon CloudWatch Synthetics)
- Apa itu Amazon EventBridge?
- Apa itu AWS Lambda?

Contoh terkait:

- Melakukan otomatisasi operasi dengan Playbook dan Runbook

REL12-BP02 Lakukan analisis pasca-insiden

Lakukan peninjauan terhadap peristiwa-peristiwa yang memengaruhi pelanggan, dan identifikasi faktor yang berkontribusi serta tindakan pencegahannya. Gunakan informasi ini untuk mengembangkan langkah-langkah mitigasi untuk meminimalkan atau mencegah kemungkinan terjadi lagi. Kembangkan prosedur untuk respons efektif dan cepat. Komunikasikan faktor-faktor yang berkontribusi dan tindakan-tindakan korektif yang diperlukan, yang disesuaikan dengan audiens target. Buatlah sebuah metode untuk mengomunikasikan penyebab ini ke pihak-pihak lain sesuai keperluan.

Lakukan penilaian untuk mengidentifikasi alasan mengapa pengujian yang ada tidak dapat menemukan masalahnya. Menambahkan pengujian untuk kasus ini jika pengujian belum ada.

Hasil yang diinginkan: Tim Anda memiliki pendekatan yang konsisten dan disepakati untuk menangani analisis pasca-insiden. Salah satu mekanismenya adalah koreksi proses error (COE). COEProses ini membantu tim Anda mengidentifikasi, memahami, dan mengatasi akar penyebab insiden, sementara juga membangun mekanisme dan pagar pembatas untuk membatasi kemungkinan insiden yang sama terjadi lagi.

Anti-pola umum:

- Menemukan temuan tentang faktor-faktor yang berkontribusi, tetapi tidak terus-menerus mencari lebih dalam berusaha mencari masalah potensial dan pendekatan lainnya untuk memitigasi.
- Hanya mengidentifikasi penyebab kesalahan manusia, dan tidak memberikan pelatihan atau otomatisasi apa pun yang dapat mencegah kesalahan manusia.

- Fokus menyalahkan, bukan memahami akar penyebabnya, sehingga tercipta budaya ketakutan dan menghambat komunikasi yang terbuka
- Tidak berbagi wawasan, yang membuat temuan-temuan analisis insiden hanya diketahui kelompok kecil saja, sehingga orang lain tidak dapat belajar dari pengalaman tersebut
- Tidak ada mekanisme yang digunakan untuk mencatat pengetahuan institusional, sehingga wawasan yang berharga hilang karena pelajaran yang didapat tidak diabadikan dalam bentuk praktik terbaik yang diperbarui secara berkelanjutan dan mengakibatkan insiden berulang dengan akar penyebab yang sama atau serupa

Manfaat menerapkan praktik terbaik ini: Dengan melakukan analisis setelah insiden dan membagikan hasilnya, beban kerja lain akan dapat memitigasi risiko jika beban kerja sudah mengimplementasikan faktor yang berkontribusi yang sama, sehingga mitigasi atau pemulihan otomatis dapat diimplementasikan sebelum insiden terjadi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Analisis setelah insiden yang baik memberikan peluang untuk mengusulkan solusi-solusi umum terhadap masalah dengan pola arsitektur yang digunakan di tempat lainnya dalam sistem.

Landasan dari COE proses ini adalah mendokumentasikan dan menangani masalah. Sebaiknya tentukan cara standar untuk mendokumentasikan akar penyebab masalah kritis, dan memastikan penyebab tersebut ditinjau dan ditangani. Tetapkan kepemilikan yang jelas untuk proses analisis setelah insiden. Tunjuk individu atau tim penanggung jawab yang akan mengawasi penyelidikan dan tindak lanjut insiden.

Dorong budaya yang berfokus pada pembelajaran dan peningkatan, bukan menyalahkan. Tekankan bahwa tujuannya adalah untuk mencegah insiden di kemudian hari, bukan untuk menghukum individu.

Kembangkan prosedur yang jelas untuk melakukan analisis setelah insiden. Prosedur ini harus menguraikan langkah-langkah yang harus diambil, informasi yang akan dikumpulkan, dan pertanyaan-pertanyaan penting yang harus dicari jawabannya saat melakukan analisis. Selidiki insiden secara menyeluruh, tidak hanya pada penyebab langsung guna mengidentifikasi akar penyebab masalah dan faktor penyumbangnya. Gunakan teknik-teknik seperti lima alasan untuk memahami lebih dalam masalah-masalah mendasar.

Buatlah repositori pelajaran yang didapat dari analisis insiden. Pengetahuan institusional ini dapat digunakan sebagai referensi untuk insiden dan upaya pencegahan ke depannya. Bagikan temuan dan wawasan dari analisis yang dilakukan setelah insiden, dan pertimbangkan untuk mengadakan pertemuan peninjauan pasca insiden yang terbuka untuk semua (open-invite) untuk membahas pelajaran yang didapatkan.

Langkah-langkah implementasi

- Saat melakukan analisis pasca-insiden, pastikan untuk tidak menyalahkan siapa pun dalam proses tersebut. Dengan begitu, orang-orang yang terlibat dalam insiden tersebut bersikap rasional terhadap tindakan korektif yang diusulkan dan mendorong penilaian mandiri yang jujur serta kolaborasi di seluruh tim.
- Tentukan cara terstandardisasi untuk mendokumentasikan masalah-masalah kritis. Contoh struktur untuk dokumen tersebut adalah sebagai berikut:
  - Apa yang terjadi?
  - Apa dampaknya terhadap para pelanggan dan bisnis Anda?
  - Apa akar penyebabnya?
  - Data apa yang Anda miliki untuk mendukung hal ini?
    - Misalnya, metrik dan grafik
  - Apa implikasi pilar kritis, terutama keamanan?
    - Saat merancang beban kerja, Anda memilah pilar-pilar sesuai dengan konteks bisnis Anda. Keputusan bisnis ini dapat mendorong prioritas rekayasa Anda. Anda dapat melakukan optimalisasi untuk mengurangi biaya dengan mengorbankan keandalan dalam lingkungan pengembangan, atau, untuk solusi yang sangat penting, Anda dapat melakukan optimalisasi keandalan dengan biaya yang lebih tinggi. Keamanan selalu menjadi hal yang didahulukan dan diutamakan, karena Anda harus melindungi pelanggan Anda.
  - Pelajaran apa hal yang Anda dapatkan?
  - Tindakan-tindakan korektif apa yang Anda ambil?
    - Item tindakan
    - Item terkait
- Buat prosedur-prosed operasi terstandardisasi yang jelas untuk melakukan analisis pasca insiden.
- Siapkan proses pelaporan insiden terstandardisasi. Dokumentasikan semua insiden secara komprehensif, termasuk laporan insiden awal, log, komunikasi, dan tindakan-tindakan yang diambil saat insiden berlangsung.

- Ingatlah bahwa sebuah insiden tidak harus berupa terhentinya sistem (outage). Insiden juga bisa berupa kejadian yang hampir menyebabkan henti sistem (near-miss), atau performa sistem yang tidak sesuai harapan meski tetap memenuhi fungsi bisnisnya.
- Terus tingkatkan proses analisis pasca insiden Anda berdasarkan umpan balik dan pelajaran yang dipetik.
- Rekam temuan-temuan utama dalam sistem manajemen pengetahuan, dan pertimbangkan pola apa pun yang perlu ditambahkan ke dalam panduan developer atau daftar periksa sebelum deployment.

Sumber daya

Dokumen terkait:

- Mengapa Anda harus mengembangkan koreksi kesalahan (COE)

Video terkait:

- Pendekatan Amazon untuk gagal dengan sukses
- AWS re:invent 2021 - Perpustakaan Amazon Builders: Keunggulan Operasional di Amazon

REL12-BP03 Uji persyaratan fungsional

Gunakan teknik seperti pengujian unit dan pengujian integrasi yang memvalidasi fungsionalitas.

Anda akan meraih hasil terbaik saat pengujian ini dijalankan secara otomatis sebagai bagian dari tindakan deployment dan build. Misalnya, menggunakan AWS CodePipeline, pengembang melakukan perubahan ke repositori sumber di mana CodePipeline secara otomatis mendeteksi perubahan. Perubahan tersebut dibangun, dan pengujian dijalankan. Setelah pengujian selesai, kode yang dibangun di-deploy ke server penahapan untuk pengujian. Dari server pementasan, CodePipeline jalankan lebih banyak tes, seperti integrasi atau tes beban. Setelah berhasil menyelesaikan pengujian tersebut, CodePipeline menyebarkan kode yang diuji dan disetujui ke instance produksi.

Selain itu, berdasarkan pengalaman, pengujian transaksi sintetis (juga disebut pengujian canary, tetapi bedakan dengan deployment canary) yang dapat menjalankan dan menyimulasikan perilaku pelanggan adalah salah satu proses pengujian yang paling penting. Jalankan pengujian ini secara

konstan terhadap titik akhir beban kerja Anda dari beragam lokasi jarak jauh. Amazon CloudWatch Synthetics memungkinkan Anda membuat kenari untuk memantau titik akhir Anda dan. APIs

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Uji persyaratan fungsional. Hal ini termasuk pengujian unit dan pengujian integrasi yang memvalidasi fungsionalitas yang diperlukan.
  - Gunakan CodePipeline dengan AWS CodeBuild untuk menguji kode dan menjalankan build
  - AWS CodePipeline Menambahkan Support untuk Unit dan Pengujian Integrasi Kustom dengan AWS CodeBuild
  - Pengiriman Berkelanjutan dan Integrasi Berkelanjutan
  - Menggunakan Canaries (Amazon CloudWatch Synthetics)
  - Otomatisasi uji perangkat lunak

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu implementasi pipa integrasi berkelanjutan
- AWS CodePipeline Menambahkan Support untuk Unit dan Pengujian Integrasi Kustom dengan AWS CodeBuild
- AWS Marketplace: produk yang dapat digunakan untuk integrasi berkelanjutan
- Pengiriman Berkelanjutan dan Integrasi Berkelanjutan
- Otomatisasi uji perangkat lunak
- Gunakan CodePipeline dengan AWS CodeBuild untuk menguji kode dan menjalankan build
- Menggunakan Canaries (Amazon CloudWatch Synthetics)

REL12-BP04 Uji penskalaan dan persyaratan kinerja

Gunakan teknik-teknik seperti pengujian beban untuk memvalidasi bahwa beban kerja memenuhi persyaratan kinerja dan penskalaan.

Di dalam cloud, Anda dapat membuat sebuah lingkungan pengujian dalam skala produksi sesuai permintaan untuk beban kerja Anda. Jika Anda menjalankan pengujian ini di infrastruktur yang

skalanya diturunkan, Anda harus menskalakan hasil observasi Anda berdasarkan apa yang Anda perkirakan terjadi di dalam produksi. Pengujian kinerja dan beban juga dapat dilakukan dalam lingkungan produksi jika Anda ingin berhati-hati agar tidak berdampak pada pengguna aktual. Tandai data pengujian Anda agar tidak tercampur dengan data pengguna nyata dan mengubah laporan statistik atau produksi.

Dengan pengujian, Anda dapat memastikan bahwa sumber daya dasar, pengaturan penskalaan, kuota layanan (service quotas), dan desain ketahanan Anda beroperasi sebagaimana mestinya saat menerima beban.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Uji persyaratan penskalaan dan kinerja. Jalankan pengujian beban untuk memvalidasi bahwa beban kerja memenuhi persyaratan kinerja dan penskalaan.
  - Pengujian Beban Terdistribusi pada AWS: mensimulasikan ribuan pengguna yang terhubung
  - Apache JMeter
    - Lakukan deployment aplikasi ke lingkungan yang menyerupai lingkungan produksi Anda, lalu jalankan pengujian beban.
      - Gunakan infrastruktur sebagai konsep kode untuk membuat sebuah lingkungan semirip mungkin dengan lingkungan produksi Anda.

Sumber daya

Dokumen terkait:

- Pengujian Beban Terdistribusi pada AWS: mensimulasikan ribuan pengguna yang terhubung
- Apache JMeter

REL12-BP05 Uji ketahanan menggunakan rekayasa kekacauan

Jalankan eksperimen chaos secara rutin di lingkungan yang berada dalam atau sedekat mungkin dengan produksi untuk memahami bagaimana sistem Anda merespons kondisi yang merugikan.

Hasil yang diinginkan:

Ketahanan beban kerja diverifikasi secara rutin dengan menerapkan chaos engineering dalam bentuk eksperimen injeksi kesalahan atau injeksi beban tak terduga. Selain itu, terdapat pengujian

ketahanan yang memvalidasi perilaku yang diharapkan yang diketahui dari beban kerja Anda selama berlangsungnya sebuah peristiwa. Gabungkan chaos engineering dan pengujian ketahanan agar Anda meyakini bahwa beban kerja dapat bertahan dari kegagalan komponen dan dapat pulih dari gangguan tak terduga dengan dampak minimal hingga tanpa dampak.

Anti-pola umum:

- Menentukan desain untuk mendapatkan ketahanan, tetapi tidak memastikan bagaimana fungsi beban kerja secara keseluruhan saat terjadi kesalahan.
- Tidak pernah bereksperimen dalam kondisi dunia nyata dan dengan beban yang diharapkan.
- Tidak memperlakukan eksperimen Anda sebagai kode atau memeliharanya melalui siklus pengembangan.
- Tidak menjalankan eksperimen chaos baik sebagai bagian dari pipeline CI/CD Anda maupun di luar deployment.
- Tidak menggunakan analisis pasca-insiden terdahulu saat menentukan kesalahan mana yang akan digunakan dalam eksperimen.

Manfaat menerapkan praktik terbaik ini: Menginjeksi kesalahan untuk memverifikasi ketahanan beban kerja Anda akan membuat Anda percaya bahwa prosedur pemulihan yang dimiliki oleh desain Anda yang tangguh akan berfungsi efektif jika terjadi kesalahan nyata.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Chaos engineering memberi tim Anda kemampuan untuk terus menginjeksi gangguan (simulasi) dunia nyata dengan cara yang terkontrol di tingkat penyedia layanan, infrastruktur, beban kerja, dan komponen, dengan dampak minimal atau tanpa dampak terhadap pelanggan Anda. Hal ini akan memungkinkan tim Anda belajar dari kesalahan serta mengamati, mengukur, dan meningkatkan ketahanan beban kerja Anda, serta memvalidasi bahwa peringatan akan diluncurkan dan tim mendapatkan notifikasi jika terjadi suatu peristiwa.

Jika dilakukan terus-menerus, chaos engineering dapat menunjukkan kekurangan dalam beban kerja Anda yang, jika dibiarkan tidak ditangani, dapat berdampak negatif pada ketersediaan dan operasi.

> **Note**
>
> Chaos engineering adalah bidang ilmu yang bereksperimen pada suatu sistem guna membangun kepercayaan pada kemampuan sistem untuk bertahan dari kondisi gangguan dalam produksi. – Prinsip-prinsip Chaos Engineering

Jika sistem mampu bertahan dari gangguan ini, eksperimen chaos harus dipertahankan sebagai pengujian regresi otomatis. Dengan cara ini, eksperimen chaos harus dilakukan sebagai bagian dari siklus hidup pengembangan sistem Anda (SDLC) dan sebagai bagian dari pipeline CI/CD Anda.

Untuk memastikan bahwa beban kerja Anda dapat bertahan dari kegagalan komponen, lakukan injeksi peristiwa dunia nyata sebagai bagian dari eksperimen Anda. Misalnya, bereksperimenlah dengan hilangnya EC2 instans Amazon atau failover instans RDS database Amazon utama, dan verifikasi bahwa beban kerja Anda tidak terpengaruh (atau hanya terkena dampak minimal). Gunakan kombinasi kesalahan komponen untuk membuat simulasi peristiwa yang mungkin disebabkan oleh gangguan di Zona Ketersediaan.

Untuk kesalahan tingkat aplikasi (seperti crash), Anda dapat mulai dengan stresor seperti memori dan kelelahan. CPU

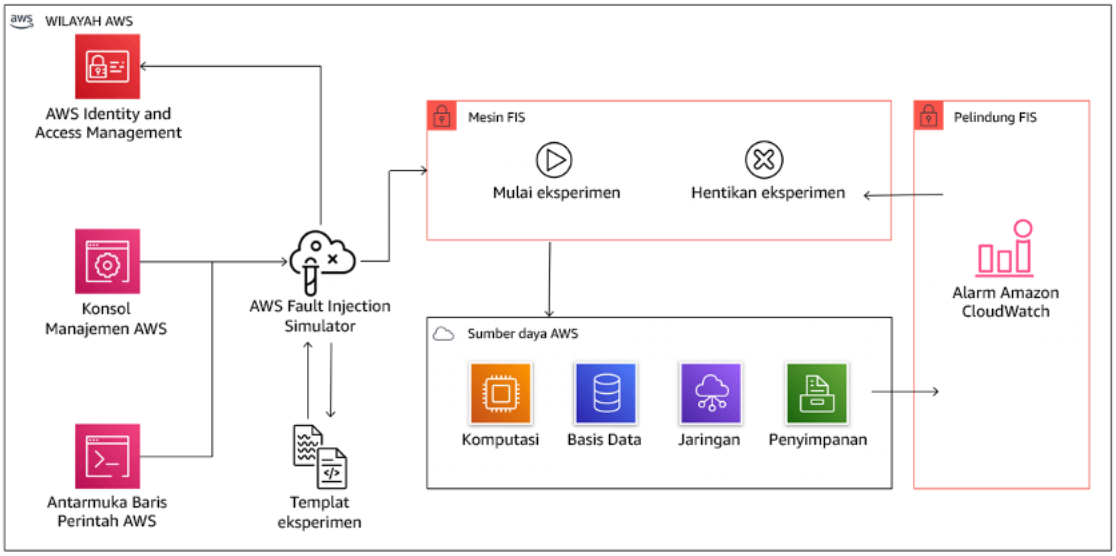
Untuk melakukan validasi mekanisme fallback atau failover untuk dependensi eksternal karena gangguan jaringan yang terputus-putus, komponen Anda harus menyimulasikan peristiwa tersebut dengan memblokir akses ke penyedia pihak ketiga selama durasi tertentu yang dapat berlangsung dari hitungan detik hingga jam.

Mode degradasi lainnya dapat menyebabkan berkurangnya fungsionalitas dan respons yang lambat, sehingga sering kali mengakibatkan gangguan terjadi pada layanan Anda. Degradasi ini umumnya disebabkan oleh terjadinya peningkatan latensi pada layanan yang sangat penting dan komunikasi jaringan yang tidak dapat diandalkan (paket yang tidak dikirim). Eksperimen dengan kesalahan ini, termasuk efek jaringan seperti latensi, pesan yang terputus, dan DNS kegagalan, dapat mencakup ketidakmampuan untuk menyelesaikan nama, mencapai DNS layanan, atau membuat koneksi ke layanan dependen.

Alat chaos engineering:

AWS Fault Injection Service (AWS FIS) adalah layanan yang dikelola sepenuhnya untuk menjalankan eksperimen injeksi kesalahan yang dapat digunakan sebagai bagian dari pipa CD Anda, atau di luar pipa. AWS FIS adalah pilihan yang baik untuk digunakan selama hari-hari permainan rekayasa

kekacauan. Ini mendukung secara bersamaan memperkenalkan kesalahan di berbagai jenis sumber daya termasuk Amazon, Amazon Elastic Container Service (EC2AmazonECS), Amazon Elastic Kubernetes Service (EKSAmazon), dan Amazon. RDS Kesalahan ini termasuk penghentian sumber daya, memaksa kegagalan, stres CPU atau memori, pelambatan, latensi, dan kehilangan paket. Karena terintegrasi dengan CloudWatch Alarm Amazon, Anda dapat mengatur kondisi berhenti sebagai pagar pembatas untuk mengembalikan eksperimen jika menyebabkan dampak yang tidak terduga.




AWS Fault Injection Service terintegrasi dengan AWS sumber daya untuk memungkinkan Anda menjalankan eksperimen injeksi kesalahan untuk beban kerja Anda.

Ada juga beberapa opsi pihak ketiga untuk melakukan eksperimen injeksi kesalahan. Ini termasuk alat-alat sumber terbuka seperti Chaos Toolkit, Chaos Mesh, dan Litmus Chaos, serta opsi komersial seperti Gremlin. Untuk memperluas cakupan kesalahan yang dapat disuntikkan AWS, AWS FIS terintegrasi dengan Chaos Mesh dan Litmus Chaos, memungkinkan Anda untuk mengoordinasikan alur kerja injeksi kesalahan di antara beberapa alat. Misalnya, Anda dapat menjalankan stress test pada pod CPU menggunakan kesalahan Chaos Mesh atau Lakmus sambil menghentikan persentase node cluster yang dipilih secara acak menggunakan tindakan kesalahan. AWS FIS

Langkah-langkah implementasi

1. Tentukan kesalahan mana yang akan digunakan untuk eksperimen.

   Lakukan penilaian desain beban kerja Anda untuk mengetahui ketahanannya. Desain semacam itu (dibuat menggunakan praktik terbaik Kerangka Well-Architected) memperhitungkan risiko berdasarkan dependensi kritis, peristiwa masa lalu, masalah yang diketahui, dan persyaratan

kepatuhan. Buat daftar yang berisi setiap elemen desain yang dimaksudkan untuk menjaga ketahanan dan kesalahan yang akan dimitigasi oleh elemen desain tersebut. Untuk informasi selengkapnya tentang cara membuat daftar tersebut, lihat laporan resmi Tinjauan Kesiapan Operasional yang memandu Anda tentang cara membuat proses untuk mencegah terulangnya insiden sebelumnya. Proses Failure Mode and Effects Analysis (FMEA) memberi Anda kerangka kerja untuk melakukan analisis kegagalan tingkat komponen dan bagaimana pengaruhnya terhadap beban kerja Anda. FMEAdiuraikan secara lebih rinci oleh Adrian Cockcroft dalam Mode Kegagalan dan Ketahanan Berkelanjutan.

2. Tetapkan prioritas untuk setiap kesalahan.

   Mulailah dengan kategorisasi yang umum seperti tinggi, sedang, atau rendah. Untuk menilai prioritas, pertimbangkan frekuensi kesalahan dan dampak kegagalan terhadap beban kerja secara keseluruhan.

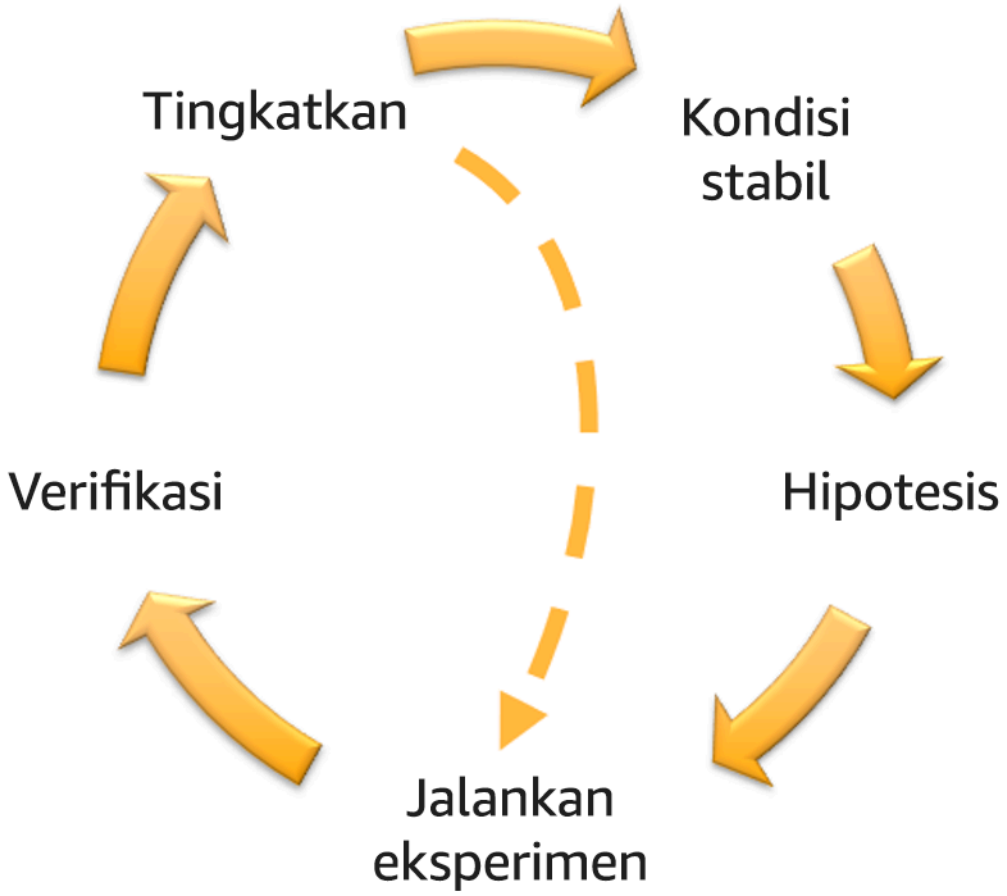
   Saat mempertimbangkan frekuensi kesalahan tertentu, lakukan analisis pada data terdahulu untuk beban kerja ini jika tersedia. Jika tidak tersedia, gunakan data dari beban kerja lain yang berjalan di lingkungan yang serupa.

   Ketika mempertimbangkan dampak dari kesalahan tertentu, makin besar cakupan kesalahan, biasanya makin besar dampaknya. Pertimbangkan juga desain dan tujuan beban kerja. Misalnya, kemampuan untuk mengakses penyimpanan data sumber sangat krusial untuk beban kerja yang melakukan transformasi dan analisis data. Dalam hal ini, Anda akan memprioritaskan eksperimen untuk kesalahan akses, serta akses yang di-throttling dan penyisipan latensi.

   Analisis pasca-insiden adalah sumber data yang baik untuk memahami frekuensi dan dampak mode kegagalan.

   Gunakan prioritas yang ditetapkan untuk menentukan kesalahan mana yang akan digunakan terlebih dahulu dalam eksperimen beserta urutannya agar dapat mengembangkan eksperimen injeksi kesalahan baru.

3. Untuk setiap eksperimen yang Anda lakukan, gunakan roda chaos engineering dan ketahanan berkelanjutan pada gambar berikut.




Roda chaos engineering dan ketahanan berkelanjutan yang menggunakan metode ilmiah dari Adrian Hornsby.

a. Definisikan kondisi stabil sebagai output terukur dari beban kerja yang menunjukkan perilaku normal.

   Beban kerja Anda menunjukkan kondisi stabil jika beroperasi dengan andal dan seperti yang diharapkan. Oleh karena itu, validasikan bahwa beban kerja Anda berkondisi baik sebelum menentukan kondisi stabil. Dalam kondisi stabil, bukan berarti tidak akan ada dampak pada beban kerja saat terjadi kesalahan, karena sejumlah kesalahan tertentu mungkin berada dalam batas yang dapat diterima. Kondisi stabil adalah acuan dasar yang akan Anda amati selama eksperimen, yang akan menunjukkan anomali jika hipotesis yang Anda tentukan pada langkah berikutnya tidak berjalan seperti yang diharapkan.

Misalnya, kondisi mapan sistem pembayaran dapat didefinisikan sebagai pemrosesan 300 TPS dengan tingkat keberhasilan 99% dan waktu pulang-pergi 500 ms.

b. Bentuk hipotesis tentang bagaimana beban kerja akan bereaksi terhadap kesalahan.

   Hipotesis yang baik didasarkan pada bagaimana beban kerja diharapkan akan memitigasi kesalahan untuk mempertahankan kondisi stabil. Hipotesis menyatakan bahwa dengan kesalahan jenis tertentu, sistem atau beban kerja akan terus berkondisi stabil karena beban kerja ini dirancang dengan mitigasi tertentu. Jenis spesifik kesalahan dan mitigasi harus ditentukan dalam hipotesis.

   Templat berikut dapat digunakan untuk hipotesis (tetapi pernyataan lain juga dapat diterima):

   > **Note**
   >
   > Jika *`specific fault`* terjadi, *`workload name`* beban kerja akan *`describe mitigating controls`* untuk mempertahankan *`business or technical metric impact`*.

   Sebagai contoh:

   - Jika 20% node di EKS grup node Amazon dihapus, Transaction Create API terus melayani persentil ke-99 permintaan dalam waktu kurang dari 100 ms (steady state). EKSNode Amazon akan pulih dalam waktu lima menit, dan pod akan mendapatkan jadwal dan memproses lalu lintas dalam waktu delapan menit setelah inisiasi percobaan. Peringatan akan diaktifkan dalam waktu tiga menit.
   - Jika terjadi kegagalan EC2 instans Amazon tunggal, pemeriksaan kesehatan Elastic Load Balancing sistem order akan menyebabkan Elastic Load Balancing hanya mengirim permintaan ke instans sehat yang tersisa sementara EC2 Auto Scaling Amazon menggantikan instans yang gagal, mempertahankan peningkatan kesalahan sisi server (5xx) di sisi server (kondisi stabil) kurang dari 0,01%.
   - Jika instans RDS database Amazon utama gagal, beban kerja pengumpulan data Rantai Pasokan akan gagal dan tersambung ke instans database RDS Amazon siaga untuk mempertahankan kesalahan baca atau tulis database kurang dari 1 menit (status tunak).

c. Jalankan eksperimen dengan menginjeksikan kesalahan.

Eksperimen secara default harus memiliki kemampuan fail-safe dan ditoleransi oleh beban kerja. Jika Anda tahu bahwa beban kerja akan gagal, jangan jalankan eksperimen. Chaos engineering harus digunakan untuk menemukan “known-unknown” atau “unknown-unknown”. Dikenal-tidak dikenali adalah hal-hal yang Anda sadari tetapi tidak sepenuhnya mengerti, dan yang tidak dikenal-tidak dikenali adalah hal-hal yang Anda pahami atau tidak pahami sepenuhnya. Bereksperimen dengan beban kerja yang Anda tahu dalam kondisi rusak tidak akan memberi Anda wawasan baru. Eksperimen Anda harus direncanakan dengan cermat, memiliki cakupan dampak yang jelas, dan menyediakan mekanisme rollback yang dapat diterapkan jika terjadi gangguan tak terduga. Jika uji tuntas Anda menunjukkan bahwa beban kerja Anda dapat bertahan dalam eksperimen, lanjutkan eksperimen. Ada beberapa opsi untuk menginjeksikan kesalahan. Untuk beban kerja aktif pada AWS, AWS FIS menyediakan banyak simulasi kesalahan standar yang disebut tindakan. Anda juga dapat menentukan tindakan kustom yang berjalan dalam AWS FIS menggunakan AWS Systems Manager dokumen.

Kami tidak menyarankan penggunaan skrip kustom untuk eksperimen chaos, kecuali jika skrip tersebut memiliki kemampuan untuk memahami status terkini beban kerja, mampu menghasilkan log, dan menyediakan mekanisme untuk rollback dan kondisi berhenti jika memungkinkan.

Kerangka kerja atau kumpulan alat efektif yang mendukung chaos engineering harus melacak kondisi terkini eksperimen, menghasilkan log, dan menyediakan mekanisme rollback untuk mendukung pelaksanaan eksperimen yang terkontrol. Mulailah dengan layanan mapan seperti AWS FIS itu memungkinkan Anda untuk melakukan eksperimen dengan ruang lingkup yang jelas dan mekanisme keamanan yang mengembalikan eksperimen jika eksperimen memperkenalkan turbulensi yang tidak terduga. Untuk mempelajari tentang berbagai eksperimen yang lebih luas menggunakan AWS FIS, lihat juga lab Resilient and Well-Architected Apps with Chaos Engineering. Selain itu, AWS Resilience Hub akan menganalisis beban kerja Anda dan membuat eksperimen yang dapat Anda pilih untuk diterapkan dan dijalankan di AWS FIS.

> **Note**
>
> Untuk setiap eksperimen, pahami dengan jelas cakupan dan dampaknya. Kami merekomendasikan bahwa kesalahan harus disimulasikan terlebih dahulu di lingkungan nonproduksi sebelum dijalankan dalam produksi.

Eksperimen harus berjalan dalam produksi di bawah beban dunia nyata menggunakan deployment canary yang memutar kontrol dan deployment sistem eksperimental, jika memungkinkan. Menjalankan eksperimen selama waktu sepi adalah praktik yang baik untuk mengurangi potensi dampak saat pertama kali bereksperimen dalam produksi. Selain itu, jika menggunakan lalu lintas pelanggan yang sebenarnya akan menimbulkan terlalu banyak risiko, Anda dapat menjalankan eksperimen menggunakan lalu lintas sintetis di infrastruktur produksi terhadap deployment kontrol dan eksperimental. Jika tidak dapat menggunakan produksi, jalankan eksperimen di lingkungan praproduksi yang semirip mungkin dengan produksi.

Anda harus membuat dan memantau pagar pembatas untuk memastikan eksperimen tidak memengaruhi lalu lintas produksi atau sistem lain di luar batas yang dapat diterima. Tetapkan kondisi berhenti untuk menghentikan eksperimen jika mencapai ambang batas pada metrik pagar pembatas yang Anda tentukan. Hal ini harus mencakup metrik untuk kondisi stabil beban kerja, serta metrik berdasarkan komponen yang diinjeksi dengan kesalahan. Monitor sintetis (juga dikenal sebagai canary pengguna) adalah salah satu metrik yang biasanya harus Anda sertakan sebagai proksi pengguna. Hentikan kondisi untuk AWS FIS didukung sebagai bagian dari templat eksperimen, sehingga memungkinkan maksimal lima kondisi berhenti per templat.

Salah satu prinsip chaos adalah meminimalkan cakupan eksperimen dan dampaknya:

Meskipun harus ada kelonggaran untuk beberapa dampak negatif dalam jangka pendek, Chaos Engineer bertanggung jawab dan berkewajiban untuk memastikan gangguan dari eksperimen diminimalkan dan dikendalikan.

Metode untuk memverifikasi cakupan dan dampak potensial adalah dengan melakukan eksperimen di lingkungan nonproduksi terlebih dahulu, memverifikasi bahwa ambang batas untuk kondisi berhenti diaktifkan seperti yang diharapkan selama eksperimen dan kemampuan pengamatan (observabilitas) diterapkan untuk menemukan pengecualian, bukan langsung bereksperimen dalam produksi.

Saat menjalankan eksperimen injeksi kesalahan, verifikasikan bahwa semua pihak yang bertanggung jawab sudah mengetahui informasi yang jelas. Berkomunikasilah dengan tim yang sesuai seperti tim operasi, tim keandalan layanan, dan dukungan pelanggan untuk memberi tahu mereka kapan eksperimen akan dijalankan dan apa yang diharapkan. Berikan alat komunikasi kepada berbagai tim ini untuk memberi tahu tim tertentu yang menjalankan eksperimen jika muncul efek yang merugikan.

Anda harus memulihkan beban kerja dan sistem yang mendasarinya kembali ke kondisi awal yang diketahui berfungsi baik. Sering kali, desain beban kerja yang tangguh akan pulih sendiri. Namun, beberapa desain yang salah atau eksperimen yang gagal dapat membuat beban kerja Anda berada dalam kondisi kegagalan yang tidak terduga. Pada akhir eksperimen, Anda harus menyadari hal ini dan memulihkan beban kerja dan sistem. Dengan AWS FIS Anda dapat mengatur konfigurasi rollback (juga disebut post action) dalam parameter tindakan. Post action mengembalikan target ke keadaan sebelum tindakan dijalankan. Baik otomatis (seperti menggunakan AWS FIS) atau manual, tindakan posting ini harus menjadi bagian dari buku pedoman yang menjelaskan cara mendeteksi dan menangani kegagalan.

d. Verifikasikan hipotesisnya.

   Prinsip Chaos Engineering memberikan panduan tentang cara memverifikasi kondisi stabil beban kerja Anda:

   Fokus pada output terukur dari suatu sistem, bukan atribut internal sistem. Pengukuran output tersebut selama periode waktu yang singkat merupakan proksi untuk kondisi stabil sistem. Throughput sistem secara keseluruhan, tingkat kesalahan, dan persentil latensi semuanya dapat menjadi metrik penting yang merepresentasikan perilaku kondisi stabil. Dengan berfokus pada pola perilaku sistemik selama eksperimen, chaos engineering memverifikasi bahwa sistem berfungsi, bukan mencoba memvalidasi cara kerjanya.

   Dalam dua contoh sebelumnya, kami menyertakan metrik kondisi stabil dengan peningkatan kesalahan sisi server (5xx) sebanyak kurang dari 0,01% serta kesalahan baca dan tulis basis data selama kurang dari satu menit.

   Kesalahan 5xx adalah metrik yang baik karena merupakan konsekuensi dari mode kegagalan yang akan dialami langsung oleh klien yang menggunakan beban kerja. Pengukuran kesalahan basis data cocok digunakan sebagai konsekuensi langsung dari kesalahan, tetapi juga harus dilengkapi dengan pengukuran dampak klien seperti permintaan pelanggan yang gagal atau kesalahan yang muncul bagi klien. Selain itu, sertakan monitor sintetis (juga dikenal sebagai kenari pengguna) pada salah satu APIs atau URIs langsung diakses oleh klien dari beban kerja Anda.

e. Tingkatkan desain beban kerja agar memiliki ketahanan.

   Jika kondisi stabil tidak dipertahankan, selidiki cara desain beban kerja dapat ditingkatkan untuk mengurangi kesalahan, dengan menerapkan praktik terbaik dari pilar Keandalan Well-Architected AWS. Panduan dan sumber daya tambahan dapat ditemukan di Perpustakaan

Pembangun AWS, yang menghosting artikel tentang cara meningkatkan pemeriksaan kondisi Anda atau menggunakan percobaan ulang dengan backoff dalam kode aplikasi Anda, antara lain.

Setelah perubahan ini diterapkan, jalankan eksperimen lagi (ditunjukkan dengan garis putus-putus pada roda chaos engineering) untuk mengetahui keefektifannya. Jika langkah verifikasi menunjukkan bahwa hipotesisnya benar, beban kerja akan berada dalam kondisi stabil, dan siklusnya berlanjut.

4. Jalankan eksperimen secara rutin.

   Eksperimen chaos adalah sebuah siklus, dan eksperimen harus dijalankan secara rutin sebagai bagian dari chaos engineering. Setelah beban kerja memenuhi hipotesis eksperimen, eksperimen harus diotomatiskan untuk terus berjalan sebagai bagian regresi dalam alur CI/CD Anda. Untuk mempelajari cara melakukan ini, lihat blog ini tentang cara menjalankan AWS FIS eksperimen menggunakan AWS CodePipeline. Laboratorium tentang eksperimen AWS FIS berulang dalam pipeline CI/CD ini memungkinkan Anda untuk bekerja langsung.

   Eksperimen injeksi kesalahan juga merupakan bagian dari game day (lihat REL12-BP06 Lakukan hari permainan secara teratur). Game day mensimulasikan kegagalan atau peristiwa untuk memverifikasi sistem, proses, dan respons tim. Tujuannya adalah untuk benar-benar menerapkan tindakan yang perlu dilakukan oleh tim seolah memang terjadi peristiwa yang tidak diharapkan.

5. Catat dan simpan hasil eksperimen.

   Hasil eksperimen injeksi kesalahan harus dicatat dan dijadikan persisten. Sertakan semua data yang diperlukan (seperti waktu, beban kerja, dan kondisi) agar dapat menganalisis hasil dan tren eksperimen nantinya. Contoh hasil mungkin termasuk tangkapan layar dasbor, CSV dump dari database metrik Anda, atau catatan peristiwa dan pengamatan yang diketik dengan tangan dari eksperimen. Pencatatan eksperimen dengan AWS FIS dapat menjadi bagian dari pengambilan data ini.

Sumber daya

Praktik-praktik terbaik terkait:

- REL08-BP03 Mengintegrasikan pengujian ketahanan sebagai bagian dari penerapan Anda
- REL13-BP03 Uji implementasi pemulihan bencana untuk memvalidasi implementasi

Dokumen terkait:

- Apa itu AWS Fault Injection Service?
- Apa itu AWS Resilience Hub?
- Prinsip-prinsip Chaos Engineering
- Chaos Engineering: Merencanakan eksperimen pertama Anda
- Rekayasa Ketahanan: Belajar untuk Mengatasi Kegagalan
- Kisah Chaos Engineering
- Menghindari fallback dalam sistem terdistribusi
- Deployment Canary untuk Eksperimen Chaos

Video terkait:

- AWS re: invent 2020: Menguji ketahanan menggunakan rekayasa kekacauan () ARC316
- AWS Re:invent 2019: Meningkatkan ketahanan dengan rekayasa kekacauan (09-R1) DOP3
- AWS re: invent 2019: Melakukan rekayasa kekacauan di dunia tanpa server (01) CMY3

Contoh terkait:

- Lab Well-Architected: Level 300: Pengujian Ketahanan Amazon, AmazonEC2, dan Amazon S3 RDS
- Lab Chaos Engineering di AWS
- Lab Aplikasi Tangguh dan Well-Architected dengan Chaos Engineering
- Lab Chaos Nirserver
- Ukur dan Tingkatkan Ketahanan Aplikasi Anda dengan lab AWS Resilience Hub

Alat terkait:

- AWS Fault Injection Service
- AWS Marketplace: Platform Chaos Engineering Gremlin
- Chaos Toolkit
- Chaos Mesh

- Litmus

REL12-BP06 Lakukan hari permainan secara teratur

Manfaatkan game day untuk secara rutin melatih prosedur Anda dalam merespons peristiwa dan kegagalan. Buat game day semirip mungkin dengan produksi (termasuk lingkungan produksi) bersama orang-orang yang akan terlibat dalam skenario kegagalan aktual. Game day menerapkan tindakan yang diperlukan guna memastikan peristiwa produksi tidak berdampak pada pengguna.

Game day menyimulasikan kegagalan atau peristiwa untuk menguji respons tim, sistem, dan proses. Tujuannya adalah untuk benar-benar menerapkan tindakan yang perlu dilakukan oleh tim seolah memang terjadi peristiwa yang tidak diharapkan. Hal ini akan membantu Anda memahami sisi mana yang perlu ditingkatkan dan membantu mengembangkan pengalaman organisasi dalam menangani peristiwa. Ini harus dilakukan secara teratur sehingga tim Anda akan membangun memori otot tentang cara memberikan respons.

Setelah desain ketangguhan Anda diterapkan dan diuji dalam lingkungan nonproduksi, game day dapat menjadi cara untuk memastikan bahwa segala sesuatu akan berjalan sesuai rencana ketika produksi. Game day, terutama yang dilakukan untuk pertama kali, merupakan aktivitas “wajib untuk semua tim”. Rekayasawan dan operasi akan diberitahu kapan ini dilakukan, dan apa yang akan terjadi. Runbook telah diterapkan. Simulasi peristiwa, termasuk peristiwa kegagalan yang mungkin terjadi, dijalankan di sistem produksi dengan cara yang sudah ditentukan, dan dampaknya dievaluasi. Jika sistem beroperasi sesuai rancangan, deteksi dan pemulihan mandiri akan berlangsung dengan sedikit atau tanpa dampak. Namun, jika timbul dampak negatif, pengujian akan diulang dan masalah beban kerja diperbaiki, secara manual jika perlu (menggunakan runbook). Karena game day biasanya berlangsung di dalam produksi, semua pencegahan harus dilakukan guna memastikan bahwa ketersediaan untuk pelanggan tidak terganggu.

Anti-pola umum:

- Mendokumentasikan prosedur, tetapi tidak pernah melatihnya.
- Tidak melibatkan pembuat keputusan bisnis dalam pengujian pelatihan.

Manfaat menerapkan praktik terbaik ini: Mengadakan game day secara rutin akan memastikan bahwa para staf mengikuti kebijakan dan prosedur ketika insiden aktual terjadi, dan memvalidasi bahwa kebijakan dan prosedur tersebut sudah sesuai.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Jadwalkan game day untuk menggunakan runbook dan buku pedoman Anda secara rutin. Game day harus mengikutsertakan semua orang yang akan terlibat dalam kejadian produksi: pemilik bisnis, staf pengembangan, staf operasional, dan tim respons insiden.
  - Jalankan pengujian beban atau kinerja Anda, kemudian jalankan injeksi kegagalan.
  - Cari anomali dalam runbook Anda dan peluang untuk menggunakan buku pedoman Anda.
    - Jika Anda tidak mengikuti runbook, perbaiki runbook atau koreksi perilakunya. Jika Anda menggunakan runbook, identifikasi buku pedoman yang seharusnya digunakan atau buat yang baru.

Sumber daya

Video terkait:

- AWS Re:invent 2019: Meningkatkan ketahanan dengan rekayasa kekacauan (09-R1) DOP3

  Contoh terkait:

- Lab AWS Well-Architected - Pengujian Ketangguhan

## REL13. Bagaimana cara Anda mempersiapkan pemulihan bencana (DR)?

Memiliki cadangan dan komponen beban kerja berlebih adalah permulaan dari strategi DR Anda. RTOdan RPO merupakan tujuan Anda untuk memulihkan beban kerja Anda. Tetapkan ini berdasarkan kebutuhan bisnis. Implementasikan sebuah strategi untuk memenuhi tujuan-tujuan ini, sambil mempertimbangkan lokasi dan fungsi data dan sumber daya beban kerja. Probabilitas gangguan dan biaya pemulihan juga merupakan faktor penting yang akan membantu menginformasikan nilai bisnis dari penyediaan pemulihan bencana untuk beban kerja.

Praktik terbaik

- REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data
- REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan
- REL13-BP03 Uji implementasi pemulihan bencana untuk memvalidasi implementasi
- REL13-BP04 Mengelola penyimpangan konfigurasi di situs atau Wilayah DR
- REL13-BP05 Mengotomatiskan pemulihan

REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data

Beban kerja memiliki tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO).

Tujuan Waktu Pemulihan (RTO) adalah penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan. Ini menentukan apa yang dianggap sebagai jendela waktu yang dapat diterima ketika layanan tidak tersedia.

Recovery Point Objective (RPO) adalah jumlah waktu maksimum yang dapat diterima sejak titik pemulihan data terakhir. Ini menentukan apa yang dianggap sebagai kehilangan data yang dapat diterima antara titik pemulihan terakhir dan gangguan layanan.

RTOdan RPO nilai merupakan pertimbangan penting saat memilih strategi Disaster Recovery (DR) yang sesuai untuk beban kerja Anda. Sasaran-sasaran ini ditentukan oleh bisnis, kemudian digunakan oleh tim teknis untuk memilih dan mengimplementasikan strategi DR.

Hasil yang Diinginkan:

Setiap beban kerja memiliki tugas RTO danRPO, didefinisikan berdasarkan dampak bisnis. Beban kerja ditetapkan ke tingkat yang telah ditentukan, menentukan ketersediaan layanan dan hilangnya data yang dapat diterima, dengan yang terkait dan. RTO RPO Jika penetapan tingkat tersebut tidak dapat dilakukan, maka ini dapat diberi tingkat khusus yang disesuaikan per beban kerja, dengan maksud untuk membuat tingkat di lain waktu. RTOdan RPO digunakan sebagai salah satu pertimbangan utama untuk pemilihan implementasi strategi pemulihan bencana untuk beban kerja. Pertimbangan tambahan dalam memilih strategi DR yakni kendala biaya, ketergantungan beban kerja, dan persyaratan operasional.

UntukRTO, pahami dampak berdasarkan durasi pemadaman. Apakah implikasinya linier, atau adakah implikasi non-linier? (contohnya, setelah empat jam, Anda mematikan jalur produksi sampai dimulainya giliran kerja berikutnya).

Matriks pemulihan bencana, seperti berikut ini, dapat membantu Anda memahami bagaimana kritikalitas beban kerja berkaitan dengan sasaran pemulihan. (Perhatikan, nilai aktual untuk sumbu X dan Y harus disesuaikan dengan kebutuhan organisasi Anda).

| Matriks Pemulihan Bencana | | | | | | |
|---|---|---|---|---|---|---|
| | | Sasaran Titik Pemulihan | | | | |
| | | < 1 Menit | < 1 Jam | < 6 Jam | < 1 Hari | + 1 Hari |
| Sasaran Waktu Pemulihan | < 10 Menit | Kritis | Kritis | Tinggi | Sedang | Sedang |
| | < 2 Jam | Kritis | Tinggi | Sedang | Sedang | Rendah |
| | < 8 Jam | Tinggi | Sedang | Sedang | Rendah | Rendah |
| | < 24 Jam | Sedang | Sedang | Rendah | Rendah | Rendah |
| | Lebih dari 24 Jam | Sedang | Rendah | Rendah | Rendah | Rendah |

Gambar 16: Matriks pemulihan bencana

Anti-pola umum:

- Tidak ditetapkan sasaran pemulihan.
- Memilih sasaran pemulihan semaunya.
- Memilih sasaran pemulihan yang terlalu longgar dan tidak memenuhi sasaran bisnis.
- Tidak memahami dampak waktu henti dan kehilangan data.
- Memilih sasaran pemulihan yang tidak realistis, seperti tanpa adanya waktu untuk pemulihan dan tanpa adanya kehilangan data, yang mungkin tidak dapat dicapai untuk konfigurasi beban kerja Anda.
- Memilih sasaran pemulihan yang lebih ketat daripada sasaran bisnis yang sesungguhnya. Ini memaksakan implementasi DR yang lebih mahal dan lebih rumit dibandingkan yang dibutuhkan beban kerja.
- Memilih sasaran pemulihan yang tidak kompatibel dengan sasaran beban kerja yang bergantung.
- Sasaran pemulihan Anda tidak mempertimbangkan persyaratan kepatuhan terhadap peraturan.
- RTOdan RPO didefinisikan untuk beban kerja, tetapi tidak pernah diuji.

Manfaat menerapkan praktik terbaik ini: Sasaran pemulihan Anda untuk waktu dan kehilangan data diperlukan untuk memandu implementasi DR Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk beban kerja tertentu, Anda harus memahami dampak waktu henti dan kehilangan data pada bisnis Anda. Umumnya, dampak akan semakin meningkat jika waktu henti atau kehilangan data semakin besar, tetapi bentuk peningkatan ini bisa berbeda, tergantung pada jenis beban kerjanya. Contohnya, Anda mungkin dapat menoleransi waktu henti hingga satu jam dengan dampak kecil, tetapi setelah itu dampaknya meningkat dengan cepat. Ada banyak bentuk dampak pada bisnis, termasuk kerugian moneter (seperti hilangnya pendapatan), hilangnya kepercayaan pelanggan (dan dampak pada reputasi), masalah operasional (seperti penurunan produktivitas atau gaji tidak terbayarkan), dan risiko yang terkait dengan peraturan. Gunakan langkah-langkah berikut untuk memahami dampak ini, dan mengatur RTO dan RPO untuk beban kerja Anda.

Langkah-langkah Implementasi

1. Tentukan pemangku kepentingan bisnis Anda untuk beban kerja ini, dan libatkan mereka untuk mengimplementasikan langkah-langkah ini. Sasaran pemulihan untuk beban kerja merupakan keputusan bisnis. Kemudian tim teknis bekerja dengan pemangku kepentingan bisnis untuk menggunakan sasaran-sasaran ini untuk memilih strategi DR.

   > **Note**
   >
   > Untuk langkah 2 dan 3, Anda dapat menggunakan the section called “Lembar kerja implementasi”.

2. Kumpulkan informasi yang diperlukan untuk mengambil keputusan dengan menjawab pertanyaan-pertanyaan di bawah ini.
3. Apakah Anda memiliki kategori atau tingkat kritikalitas untuk dampak beban kerja di organisasi Anda?
   a. Jika ya, tetapkan beban kerja ini ke salah satu kategori
   b. Jika tidak, maka tetapkan kategori-kategori ini. Buat lima kategori atau lebih sedikit dan sempurnakan rentang sasaran waktu pemulihan Anda untuk setiap kategori. Contoh kategori antara lain: kritis, tinggi, sedang, rendah. Untuk memahami cara pemetaan beban kerja ke kategori, pertimbangkan apakah beban kerja itu kritis untuk misi perusahaan, penting bagi bisnis, atau tidak mendorong bisnis.
   c. Atur beban kerja RTO dan RPO berdasarkan kategori. Selalu pilih kategori yang lebih ketat (lebih rendah RTO danRPO) daripada nilai mentah yang dihitung memasuki langkah ini. Jika

ini menghasilkan perubahan nilai yang besar dan tidak sesuai, maka pertimbangkan untuk membuat kategori baru.

4. Berdasarkan jawaban ini, tetapkan RTO dan RPO nilai pada beban kerja. Ini dapat dilakukan secara langsung, atau dengan menetapkan beban kerja ke tingkat layanan yang ditetapkan sebelumnya.
5. Dokumentasikan rencana pemulihan bencana (DRP) untuk beban kerja ini, yang merupakan bagian dari rencana kelangsungan bisnis organisasi Anda (BCP), di lokasi yang dapat diakses oleh tim beban kerja dan pemangku kepentingan
   a. Catat RTO danRPO, dan informasi yang digunakan untuk menentukan nilai-nilai ini. Sertakan strategi yang digunakan untuk mengevaluasi dampak beban kerja pada bisnis
   b. Catat metrik lain selain itu RTO dan RPO apakah Anda melacak atau berencana untuk melacak tujuan pemulihan bencana
   c. Anda akan menambahkan detail strategi DR Anda dan runbook pada rencana ini ketika Anda membuat ini.
6. Dengan mencari kritikalitas beban kerja di dalam matriks seperti yang ada dalam Gambar 15, Anda dapat mulai menetapkan tingkat layanan yang ditetapkan di muka untuk organisasi Anda.
7. Setelah Anda menerapkan strategi DR (atau bukti konsep untuk strategi DR) sesuaithe section called “REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan”, uji strategi ini untuk menentukan beban kerja aktual RTC (Kemampuan Waktu Pemulihan) dan RPC (Kemampuan Titik Pemulihan). Jika ini tidak memenuhi sasaran pemulihan target, maka bekerjalah dengan pemangku kepentingan bisnis Anda untuk menyesuaikan sasaran-sasaran tersebut, atau buat perubahan pada strategi DR yang memungkinkan untuk memenuhi sasaran target.

Pertanyaan utama

1. Berapakah waktu henti maksimum untuk beban kerja sebelum timbul dampak serius pada bisnis
   a. Tentukan kerugian moneter (dampak finansial langsung) pada bisnis per menit jika beban kerja terganggu.
   b. Pertimbangkan bahwa dampak tidak selalu linier. Pada awalnya, dampak bisa terbatas, tetapi kemudian meningkat dengan cepat melampaui titik kritis dalam waktu.
2. Berapakah jumlah data maksimum yang bisa hilang sebelum timbul dampak serius pada bisnis
   a. Pertimbangkan nilai ini untuk penyimpanan data Anda yang paling kritis. Identifikasi kritikalitas masing-masing untuk penyimpanan data lainnya.

   b. Dapatkah data beban kerja dibuat jika hilang? Jika ini secara operasional lebih mudah daripada backup dan restore, maka pilih RPO berdasarkan kekritisan data sumber yang digunakan untuk membuat ulang data beban kerja.
3. Apa saja sasaran pemulihan dan harapan ketersediaan beban kerja yang hal ini andalkan (hilir), atau beban kerja yang mengandalkan hal ini (hulu)?
   a. Pilih sasaran pemulihan yang memungkinkan beban kerja ini untuk memenuhi persyaratan-persyaratan ketergantungan hulu
   b. Pilih sasaran pemulihan yang dapat dicapai mengingat kemampuan pemulihan ketergantungan hilir. Ketergantungan hilir non-kritis (yang dapat Anda “tangani”) dapat dikecualikan. Atau, bekerjalah dengan ketergantungan hilir kritis atau tingkatkan kemampuan pemulihannya apabila perlu.

Pertanyaan tambahan

Pertimbangkan pertanyaan-pertanyaan ini, dan bagaimana pertanyaan tersebut mungkin berlaku pada beban kerja ini:

4. Apakah Anda berbeda RTO dan RPO tergantung pada jenis pemadaman (Wilayah vs AZ, dll.)?
5. Apakah ada waktu tertentu (musiman, acara penjualan, peluncuran produk) ketika RTO RPO / Anda dapat berubah? Jika ya, apakah batas waktu dan pengukurannya yang berbeda?
6. Berapa jumlah pelanggan yang akan terkena dampak jika beban kerja terganggu?
7. Apakah dampak pada reputasi jika beban kerja terganggu?
8. Dampak operasional lain apakah yang dapat timbul jika beban kerja terganggu? Contohnya, dampak pada produktivitas karyawan jika sistem email tidak tersedia, atau jika sistem Gaji tidak dapat mengirimkan transaksi.
9. Bagaimana beban kerja RTO dan RPO selaras dengan Strategi Lini Bisnis dan Organisasi DR?
10. Apakah ada kewajiban kontrak internal untuk memberikan layanan? Apakah ada penalti jika tidak memenuhinya?
11. Apa saja kendala kepatuhan atau peraturan terkait data?

Lembar kerja implementasi

Anda dapat menggunakan lembar kerja ini untuk langkah implementasi 2 dan 3. Anda dapat menyesuaikan lembar kerja ini agar cocok dengan kebutuhan spesifik Anda, seperti menambahkan pertanyaan tambahan.

| Langkah 2: Pertanyaan utama | Berlaku untuk beban kerja? | RTO beban kerja | RPO beban kerja | Penyesuaian RTO. | Penyesuaian RPO. | Instruksi |
|---|---|---|---|---|---|---|
| [1] waktu maksimum beban kerja dapat mengalami waktu henti | | | | | | diukur pada waktunya dari mulai pemadaman hingga pemulihan |
| [2] jumlah maksimum data yang bisa hilang | | | | | | diukur pada waktunya sejak set data dapat dipulihkan berkualitas baik terakhir diketahui |
| [3a] dependensi hulu | | | | | | masukkan tujuan pemulihan hilir yang paling ketat |
| [3b] dependensi hilir | | | | | | masukkan tujuan pemulihan hilir yang paling longgar |
| [3a] dependensi hulu yang direkonsiliasi | | | | | | Jika nilai hulu kurang dari nilai saat ini dan nilai hilir lebih besar, bekerjalah dengan |
| [3b] dependensi hilir yang direkonsiliasi | | | | | | dependensi untuk merekonsiliasi dan masukkan nilai yang direkonsiliasi di sini |
| [3] dependensi | | | | | | turunkan nilai untuk memenuhi dependensi hulu atau naikkan berdasarkan kemampuan dependensi hilir |
| | | | | | | |
| **Langkah 2: Pertanyaan tambahan** | | | | | | Tunjukkan apakah pertanyaan berlaku. Jika tidak, lewati |
| RTO/RPO dasar | | | | | | Turunkan nilai RTO dan RPO dari atas ke sini |
| [4] tipe pemadaman | [ ]Y / [ ]N | | | | | Masukkan tujuan pemulihan untuk tipe peristiwa dengan persyaratan paling ketat |
| [5] tujuan berbasis waktu khusus | [ ]Y / [ ]N | | | | | Masukkan tujuan pemulihan untuk waktu-waktu dengan persyaratan paling ketat |
| [6] pelanggan terganggu | [ ]Y / [ ]N | | | | | Buat grafik pelanggan yang terkena dampak saat fungsi mengalami waktu henti atau data hilang. Gunakan grafik tersebut untuk memasukkan RTO dan RPO maksimum yang diizinkan berdasarkan dampak pelanggan |
| [7] dampak reputasi | [ ]Y / [ ]N | | | | | Bekerjalah dengan bisnis untuk menentukan RTO dan RPO berdasarkan dampak terhadap reputasi |
| [8] dampak operasi | [ ]Y / [ ]N | | | | | Masukkan RTO dan RPO maksimum berdasarkan dampak operasional |
| [9] penyelarasan organisasi | [ ]Y / [ ]N | | | | | Masukkan RTO dan RPO maksimum untuk beban kerja tipe ini sesuai LOB dan persyaratan organisasi |
| [10] kewajiban kontrak | [ ]Y / [ ]N | | | | | Masukkan RTO dan RPO maksimum berdasarkan kewajiban kontrak |
| [11] kepatuhan peraturan | [ ]Y / [ ]N | | | | | Masukkan RTO dan RPO maksimum berdasarkan kepatuhan peraturan yang berlaku |
| target berdasarkan pertanyaan tambahan | | | | | | Ambil nilai minimum (nilai yang lebih ketat) dari Q 4-11 dan masukkan di sini |
| target yang disesuaikan | | | | | | Jika tujuan di baris di atas tidak dapat diakomodasi, bekerjalah dengan pemangku kepentingan untuk mengurai hambatan, dan masukkan jumlah minimum baru di sini |
| | | | | | | |
| RTO/RPO yang disesuaikan | | | | | | Masukkan nilai RPO/RTO acuan, atau target yang disesuaikan, mana saja yang lebih rendah |
| | | | | | | |
| **Langkah 3** | | | | | | |
| Peta ke kategori atau tingkat yang ditetapkan sebelumnya | | | | | | Sesuaikan kedua nilai ke bawah (lebih ketat) agar selaras dengan tingkat yang ditetapkan terdekat |

Lembar kerja

Tingkat upaya untuk Rencana Implementasi: Rendah

Sumber daya

Praktik-Praktik Terbaik Terkait:

- the section called “REL09-BP04 Lakukan pemulihan data secara berkala untuk memverifikasi integritas dan proses cadangan”
- the section called “REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan”
- the section called “REL13-BP03 Uji implementasi pemulihan bencana untuk memvalidasi implementasi”

Dokumen terkait:

- Blog Arsitektur AWS : Seri Pemulihan Bencana
- Pemulihan Bencana Beban Kerja di AWS: Pemulihan di Cloud (AWS Whitepaper)

- Mengelola kebijakan ketahanan dengan AWS Resilience Hub
- APNMitra: mitra yang dapat membantu pemulihan bencana
- AWS Marketplace: produk yang dapat digunakan untuk pemulihan bencana

Video terkait:

- AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2
- Pemulihan Bencana Beban Kerja pada AWS

REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan

Tentukan strategi pemulihan bencana (DR) yang memenuhi sasaran pemulihan beban kerja. Pilih strategi seperti pencadangan dan pemulihan, standby (aktif/pasif), atau aktif/aktif.

Hasil yang diinginkan: Strategi DR ditentukan dan diimplementasikan untuk setiap beban kerja agar beban kerja dapat mencapai sasaran DR. Strategi DR antara beban kerja menggunakan pola yang dapat digunakan kembali (seperti strategi yang telah dijelaskan sebelumnya),

Anti-pola umum:

- Mengimplementasikan prosedur pemulihan yang tidak konsisten untuk beban kerja dengan sasaran DR yang serupa.
- Membiarkan strategi DR diimplementasikan secara ad-hoc saat bencana terjadi.
- Tidak memiliki rencana untuk pemulihan bencana.
- Dependensi pada operasi bidang kontrol selama pemulihan.

Manfaat menjalankan praktik terbaik ini:

- Dengan strategi pemulihan yang ditentukan, Anda dapat menggunakan prosedur tes dan peralatan umum.
- Menggunakan strategi pemulihan yang ditentukan akan meningkatkan penyebaran pengetahuan antara tim dan implementasi DR pada beban kerja milik mereka.

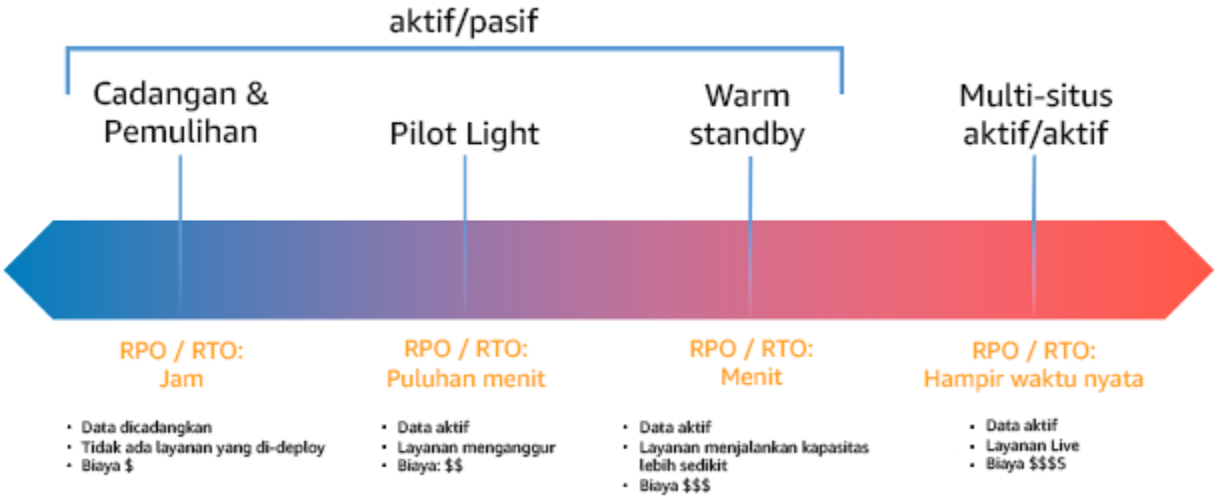
Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi. Tanpa strategi DR yang direncanakan, diimplementasikan, dan diuji, Anda akan kesulitan mencapai sasaran pemulihan ketika bencana terjadi.

Panduan implementasi

Strategi DR mengandalkan kemampuan untuk mempertahankan beban kerja di situs pemulihan jika lokasi utama tidak dapat menjalankan beban kerja. Tujuan pemulihan yang paling umum adalah RTO danRPO, seperti yang dibahas dalamREL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data.

Strategi DR di beberapa Availability Zone (AZs) dalam satu Wilayah AWS, dapat memberikan mitigasi terhadap peristiwa bencana seperti kebakaran, banjir, dan pemadaman listrik besar. Jika merupakan persyaratan untuk menerapkan perlindungan terhadap peristiwa yang tidak mungkin yang mencegah beban kerja Anda agar tidak dapat berjalan di tempat tertentu Wilayah AWS, Anda dapat menggunakan strategi DR yang menggunakan beberapa Wilayah.

Anda harus memilih salah satu dari strategi berikut saat merancang strategi DR di beberapa Wilayah. Mereka terdaftar dalam urutan peningkatan biaya dan kompleksitas, dan penurunan urutan dan. RTO RPO Wilayah Pemulihan mengacu pada yang Wilayah AWS lain selain yang utama yang digunakan untuk beban kerja Anda.




Gambar 17: Strategi pemulihan bencana (DR)

- Backup dan restore (RPOdalam jam, RTO dalam 24 jam atau kurang): Cadangkan data dan aplikasi Anda ke Wilayah pemulihan. Menggunakan backup otomatis atau kontinu akan memungkinkan point in time recovery (PITR), yang dapat turun RPO hingga serendah 5 menit dalam beberapa kasus. Jika terjadi bencana, Anda akan menggunakan infrastruktur Anda

(menggunakan infrastruktur sebagai kode untuk mengurangiRTO), menyebarkan kode Anda, dan memulihkan data cadangan untuk pulih dari bencana di Wilayah pemulihan.

- Lampu pilot (RPOdalam hitungan menit, RTO dalam puluhan menit): Menyediakan salinan infrastruktur beban kerja inti Anda di Wilayah pemulihan. Replikasikan data ke Wilayah pemulihan dan buat cadangan di sana. Sumber daya yang diperlukan untuk mendukung replikasi dan pencadangan data, misalnya basis data dan penyimpanan objek, selalu aktif. Elemen lainnya seperti server aplikasi atau komputasi nirserver tidak di-deploy, tetapi dapat dibuat saat dibutuhkan dengan kode aplikasi dan konfigurasi yang diperlukan.
- Siaga hangat (RPOdalam hitungan detik, RTO dalam hitungan menit): Pertahankan versi beban kerja Anda yang diperkecil namun berfungsi penuh yang selalu berjalan di Wilayah pemulihan. Sistem yang vital untuk bisnis sepenuhnya digandakan dan selalu aktif, tetapi dengan armada yang diturunkan skalanya. Data direplikasi dan berada dalam Wilayah pemulihan. Ketika pemulihan diperlukan, sistem dinaikkan skalanya dengan cepat untuk menangani beban produksi. Semakin ditingkatkan siaga hangat, semakin rendah RTO dan ketergantungan bidang kontrol. Saat skala sesuai sepenuhnya, ini disebut sebagai hot standby.
- Multi-Region (multi-situs) aktif-aktif (RPOmendekati nol, RTO berpotensi nol): Beban kerja Anda diterapkan ke, dan secara aktif melayani lalu lintas dari, beberapa. Wilayah AWS Strategi ini perlu menyinkronkan data di seluruh Wilayah. Konflik potensial yang disebabkan oleh menulis catatan yang sama di dua replika wilayah yang berbeda harus dihindari atau ditangani, karena bisa menjadi kompleks. Replikasi data berguna untuk sinkronisasi data dan akan melindungi Anda dari beberapa jenis bencana, tetapi tidak akan melindungi Anda dari korupsi atau kerusakan data kecuali solusi Anda juga menyertakan opsi untuk point-in-time pemulihan.

> **Note**
>
> Perbedaan antara pilot light dan warm standby terkadang sulit dimengerti. Keduanya menyertakan lingkungan di Wilayah pemulihan dengan salinan aset wilayah utama. Perbedaannya adalah pilot light tidak dapat memproses permintaan tanpa lebih dulu melakukan tindakan tambahan, sedangkan warm standby dapat menangani lalu lintas (pada kapasitas yang dikurangi) dengan cepat. Pilot light mengharuskan Anda mengaktifkan server, menaikkan skala, dan mungkin mengharuskan Anda melakukan deployment infrastruktur tambahan (bukan inti). Sementara itu, warm standby hanya meminta Anda untuk menaikkan skala (semuanya sudah di-deploy dan dijalankan). Pilih antara ini berdasarkan RPO kebutuhan RTO dan kebutuhan Anda.
>
> Ketika biaya menjadi perhatian, dan Anda ingin mencapai RTO tujuan yang sama RPO dan seperti yang didefinisikan dalam strategi siaga hangat, Anda dapat mempertimbangkan solusi

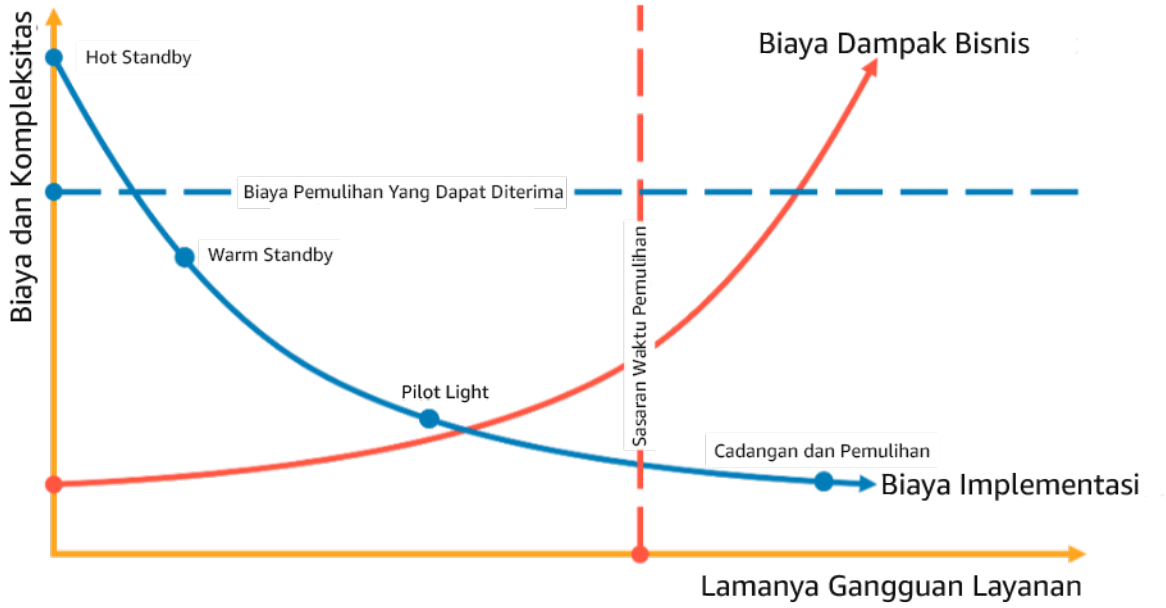
cloud native, seperti AWS Elastic Disaster Recovery, yang mengambil pendekatan ringan pilot dan menawarkan peningkatan RPO dan RTO target.

Langkah-langkah implementasi

1. Tentukan strategi DR yang akan memenuhi persyaratan pemulihan untuk beban kerja ini.

   Memilih strategi DR adalah trade-off antara mengurangi downtime dan kehilangan data (danRPO) RTO dan biaya dan kompleksitas penerapan strategi. Sebaiknya hindari strategi yang lebih sulit dari yang dibutuhkan, karena hal ini akan menambah biaya yang tidak perlu.

   Misalnya, dalam diagram berikut, bisnis telah menentukan maksimum yang diizinkan RTO serta batas dari apa yang dapat mereka belanjakan untuk strategi pemulihan layanan mereka. Mengingat tujuan bisnis, strategi DR pilot light atau warm standby akan memenuhi kriteria RTO dan biaya.




   Gambar 18: Memilih strategi DR berdasarkan RTO dan biaya

   Untuk mempelajari lebih lanjut, lihat Business Continuity Plan (BCP).

2. Tinjau pola tentang bagaimana strategi DR yang dipilih dapat diimplementasikan.

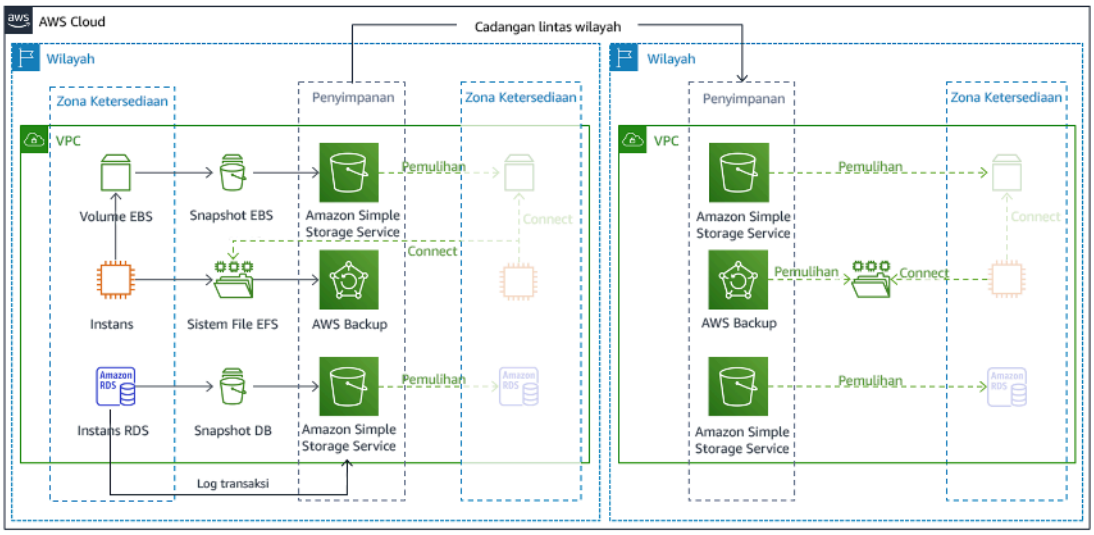
   Langkah ini digunakan untuk memahami cara Anda mengimplementasikan strategi yang dipilih. Strategi dijelaskan menggunakan Wilayah AWS sebagai situs utama dan pemulihan. Namun,

Anda juga dapat memilih untuk menggunakan Zona Ketersediaan dalam Wilayah tunggal sebagai strategi DR, yang menggunakan beberapa elemen dari berbagai strategi tersebut.

Dalam langkah berikut ini, Anda dapat menerapkan strategi pada beban kerja spesifik Anda.

Pencadangan dan pemulihan

Backup dan restore adalah strategi yang paling tidak rumit untuk diterapkan, tetapi akan membutuhkan lebih banyak waktu dan upaya untuk memulihkan beban kerja, yang mengarah ke lebih tinggi RTO danRPO. Merupakan praktik yang baik untuk selalu membuat cadangan data Anda, dan menyalinnya ke situs lain (seperti yang lain Wilayah AWS).

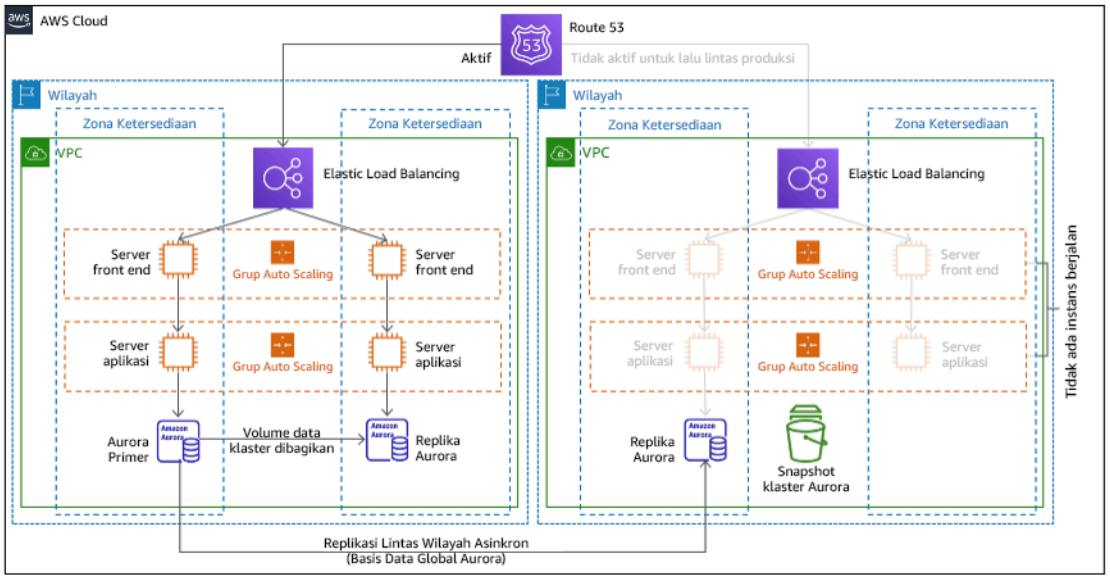



Gambar 19: Arsitektur pencadangan dan pemulihan

Untuk detail lebih lanjut tentang strategi ini, lihat Arsitektur Disaster Recovery (DR) pada AWS, Bagian II: Backup and Restore with Rapid Recovery.

Pilot light

Dengan pendekatan pilot light, Anda mereplikasi data dari Wilayah utama ke Wilayah pemulihan. Sumber daya inti yang digunakan untuk infrastruktur beban kerja di-deploy di Wilayah pemulihan. Namun, sumber daya tambahan dan dependensi lainnya masih diperlukan untuk membuat tumpukan fungsional ini. Misalnya, dalam gambar 20, tidak ada instans komputasi yang di-deploy.

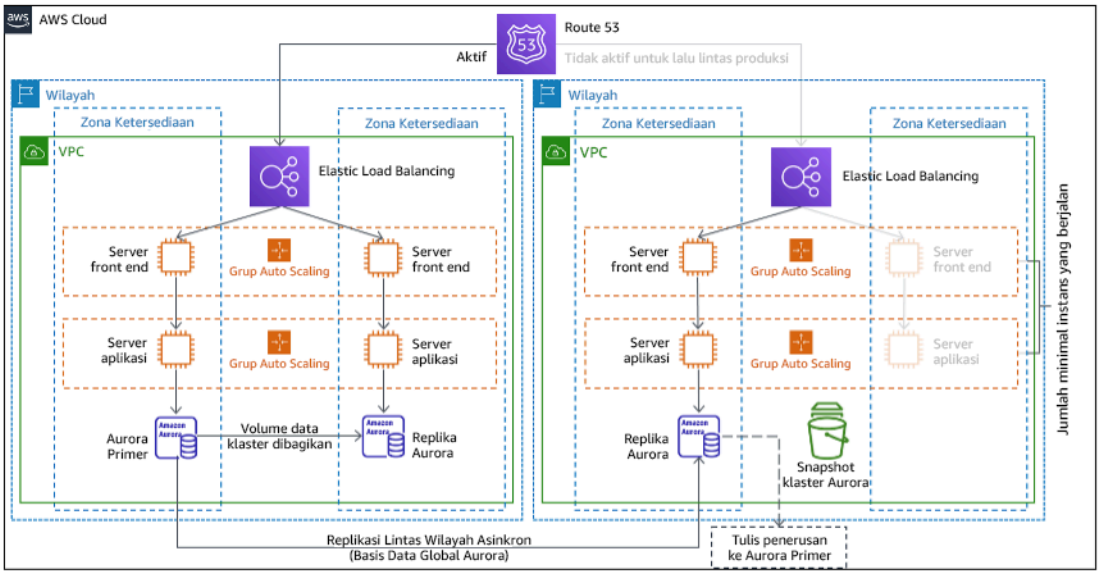



Gambar 20: Arsitektur pilot light

Untuk detail lebih lanjut tentang strategi ini, lihat Arsitektur Pemulihan Bencana (DR) di AWS, BagianIII: Pilot Light dan Warm Standby.

Warm standby

Pendekatan warm standby melibatkan memastikan ada salinan lingkungan produksi yang skalanya diturunkan tetapi berfungsi sepenuhnya di Wilayah lainnya. Pendekatan ini memperpanjang konsep pilot light dan mempercepat waktu pemulihan karena beban kerja selalu aktif di Wilayah lainnya. Jika Wilayah pemulihan di-deploy pada kapasitas penuh, hal ini disebut dengan hot standby.



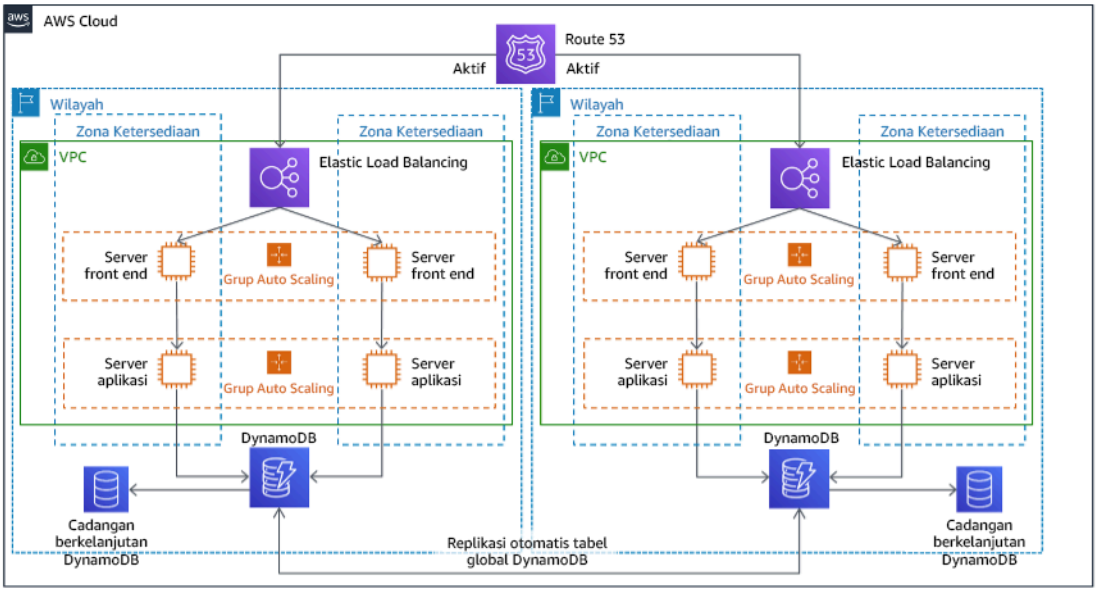

Gambar 21: Arsitektur warm standby

Saat menggunakan warm standby atau pilot light, Anda perlu menaikkan skala sumber daya di Wilayah pemulihan. Untuk memverifikasi kapasitas yang tersedia bila diperlukan, pertimbangkan penggunaan untuk reservasi kapasitas untuk EC2 instans. Jika menggunakan AWS Lambda, maka konkurensi yang disediakan dapat menyediakan lingkungan runtime sehingga mereka siap untuk segera merespons pemanggilan fungsi Anda.

Untuk detail lebih lanjut tentang strategi ini, lihat Arsitektur Pemulihan Bencana (DR) di AWS, BagianIII: Pilot Light dan Warm Standby.

Multi-situs aktif/aktif

Anda dapat menjalankan beban kerja secara berkelanjutan di beberapa Wilayah sebagai bagian dari strategi multi-situs aktif/aktif. Multi-situs aktif/aktif menjalankan lalu lintas dari semua wilayah ke wilayah tempatnya di-deploy. Konsumen dapat memilih strategi ini untuk alasan selain dari DR. Strategi ini dapat digunakan untuk meningkatkan ketersediaan, atau saat melakukan deployment beban kerja ke audiens global (untuk menempatkan titik akhir lebih dekat dengan pengguna dan/atau melakukan deployment tumpukan yang dilokalkan untuk audiens di wilayah tersebut). Sebagai strategi DR, jika beban kerja tidak dapat didukung di salah satu tempat penyebarannya, maka Wilayah tersebut dievakuasi, dan Wilayah yang tersisa digunakan untuk menjaga ketersediaan. Wilayah AWS Multi-situs aktif/aktif adalah strategi DR yang paling sulit dioperasikan, dan sebaiknya hanya dipilih saat persyaratan bisnis mengharuskannya.

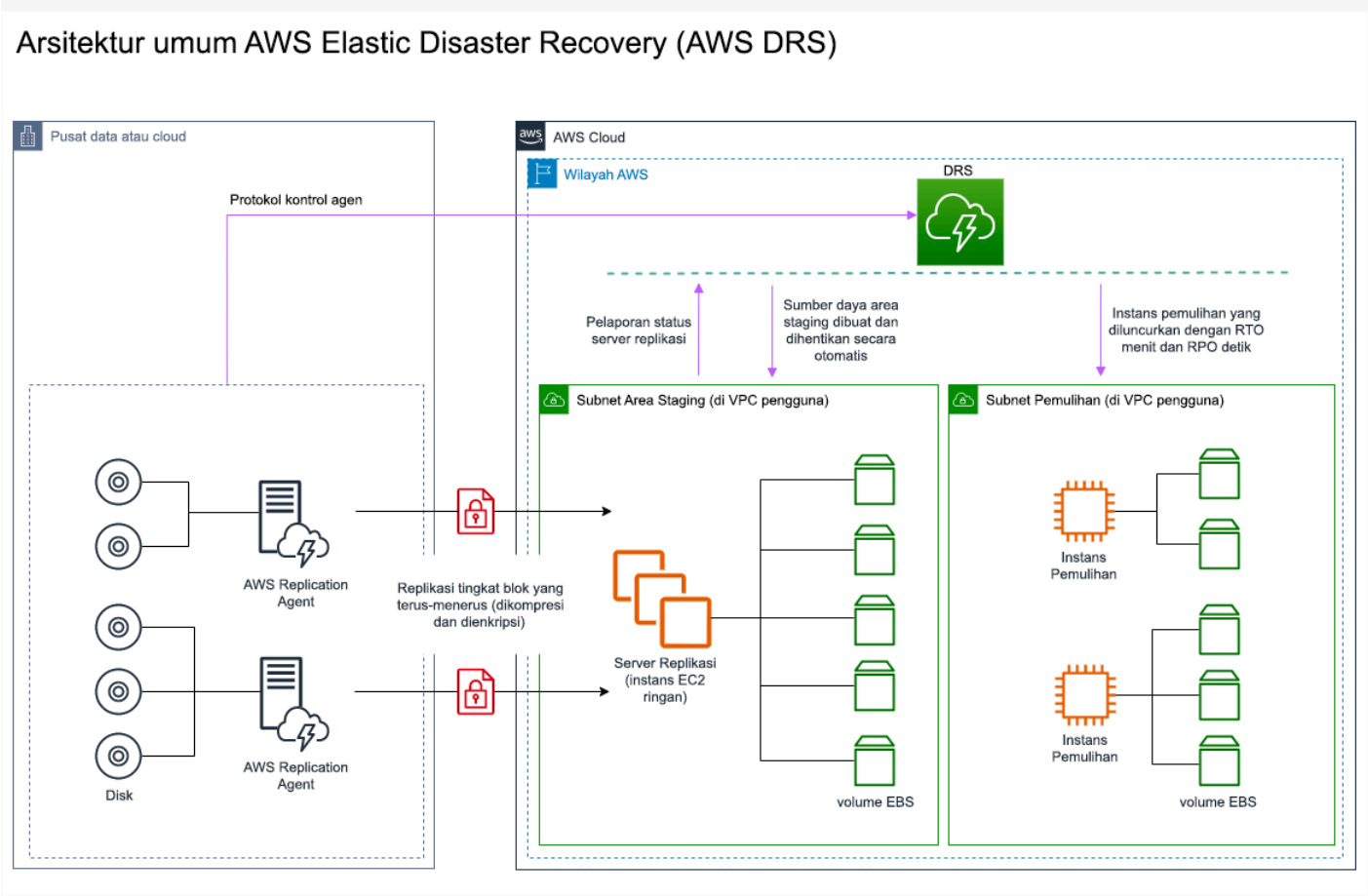



Gambar 22: Arsitektur multi-situs aktif/aktif

Untuk detail lebih lanjut tentang strategi ini, lihat Arsitektur Pemulihan Bencana (DR) di AWS, Bagian IV: Multi-situs Aktif/Aktif.

AWS Elastic Disaster Recovery

Jika Anda mempertimbangkan lampu pilot atau strategi siaga hangat untuk pemulihan bencana, AWS Elastic Disaster Recovery dapat memberikan pendekatan alternatif dengan manfaat yang lebih baik. Elastic Disaster Recovery dapat menawarkan RPO dan RTO menargetkan yang mirip dengan siaga hangat, tetapi mempertahankan pendekatan lampu pilot yang berbiaya rendah. Elastic Disaster Recovery mereplikasi data Anda dari wilayah utama Anda ke Wilayah pemulihan Anda, menggunakan perlindungan data berkelanjutan untuk mencapai RPO pengukuran dalam hitungan detik dan RTO yang dapat diukur dalam hitungan menit. Hanya sumber daya yang diperlukan untuk mereplikasi data yang di-deploy di wilayah pemulihan, yang menekan biaya tetap rendah, serupa dengan strategi pilot light. Ketika menggunakan Pemulihan Bencana Elastis, layanan mengoordinasi dan mengatur pemulihan sumber daya komputasi ketika dimulai sebagai bagian dari failover atau latihan.




Gambar 23: AWS Elastic Disaster Recovery arsitektur

Praktik tambahan untuk melindungi data

Dengan semua strategi, Anda juga harus melakukan mitigasi terhadap bencana data. Replikasi data berkelanjutan melindungi Anda dari beberapa jenis bencana, tetapi mungkin tidak melindungi Anda dari korupsi atau penghancuran data kecuali strategi Anda juga mencakup versi data yang disimpan atau opsi untuk pemulihan. point-in-time Anda juga harus mencadangkan data yang direplikasi di situs pemulihan untuk membuat point-in-time cadangan selain replika.

Menggunakan beberapa Availability Zones (AZs) dalam satu Wilayah AWS

Saat menggunakan beberapa AZs dalam satu Wilayah, implementasi DR Anda menggunakan beberapa elemen dari strategi di atas. Pertama, Anda harus membuat arsitektur ketersediaan tinggi (HA), menggunakan beberapa AZs seperti yang ditunjukkan pada Gambar 23. Arsitektur ini menggunakan pendekatan aktif/aktif multi-situs, karena EC2instans Amazon dan Elastic Load Balancer memiliki sumber daya yang digunakan dalam beberapa permintaan yang secara aktif

menyerahkan. AZs Arsitektur juga menunjukkan siaga panas, di mana jika RDS instance Amazon utama gagal (atau AZ itu sendiri gagal), maka instance siaga dipromosikan ke primer.


Gambar 24: Arsitektur Multi-AZ

Selain arsitektur HA ini, Anda perlu menambahkan cadangan data yang dibutuhkan untuk menjalankan beban kerja. Ini sangat penting untuk data yang dibatasi ke satu zona seperti EBSvolume Amazon atau cluster Amazon Redshift. Jika sebuah AZ gagal, Anda perlu memulihkan data ini ke AZ lainnya. Jika memungkinkan, Anda juga harus menyalin cadangan data ke yang lain Wilayah AWS sebagai lapisan perlindungan tambahan.

Pendekatan alternatif yang kurang umum untuk Wilayah tunggal, Multi-AZ DR diilustrasikan dalam posting blog, Membangun aplikasi yang sangat tangguh menggunakan Pengontrol Pemulihan Aplikasi Amazon, Bagian 1: tumpukan Wilayah Tunggal. Di sini, strateginya adalah mempertahankan isolasi sebanyak mungkin antara yang AZs mungkin, seperti bagaimana Wilayah beroperasi. Dengan menggunakan strategi alternatif ini, Anda dapat memilih pendekatan aktif/aktif atau aktif/pasif.

> **Note**
>
> Beberapa beban kerja memiliki persyaratan residensi data peraturan. Jika ini berlaku untuk beban kerja Anda di wilayah yang saat ini hanya memiliki satu Wilayah AWS, maka Multi-region tidak akan sesuai dengan kebutuhan bisnis Anda. Strategi multi-AZ memberikan perlindungan yang baik terhadap sebagian besar bencana.

3. Evaluasikan sumber daya beban kerja, dan seperti apa konfigurasinya di Wilayah pemulihan sebelum failover (selama operasi normal).

   Untuk infrastruktur dan AWS sumber daya gunakan infrastruktur sebagai kode seperti AWS CloudFormationatau alat pihak ketiga seperti Hashicorp Terraform. Untuk menyebarkan di beberapa akun dan Wilayah dengan satu operasi yang dapat Anda gunakan AWS CloudFormation StackSets. Untuk strategi Multi-situs aktif/aktif dan Hot Standby, infrastruktur yang di-deploy di Wilayah pemulihan memiliki sumber daya yang sama seperti Wilayah utama. Untuk strategi Pilot Light dan Warm Standby, infrastruktur yang di-deploy memerlukan tindakan tambahan agar berubah menjadi siap produksi. Dengan menggunakan CloudFormation parameter dan logika bersyarat, Anda dapat mengontrol apakah tumpukan yang digunakan aktif atau siaga dengan satu templat. Ketika menggunakan Pemulihan Bencana Elastis, layanan akan mereplikasi dan mengatur pemulihan konfigurasi aplikasi dan sumber daya komputasi.

   Semua strategi DR mengharuskan sumber data dicadangkan di dalam Wilayah AWS, dan kemudian cadangan tersebut disalin ke Wilayah pemulihan. AWS Backupmenyediakan tampilan terpusat di mana Anda dapat mengonfigurasi, menjadwalkan, dan memantau cadangan untuk sumber daya ini. Untuk Pilot Light, Warm Standby, dan Multi-situs aktif/aktif, Anda juga harus mereplikasi data dari Wilayah utama ke sumber daya data di Wilayah pemulihan, seperti instans RDSDB Amazon Relational Database Service (Amazon) atau tabel Amazon DynamoDB. Dengan demikian, sumber data ini aktif dan siap menangani permintaan di Wilayah pemulihan.

   Untuk mempelajari lebih lanjut tentang cara AWS layanan beroperasi di seluruh Wilayah, lihat seri blog ini tentang Membuat Aplikasi Multi-Wilayah dengan AWS Layanan.

4. Tentukan dan implementasikan cara Anda mempersiapkan Wilayah untuk failover saat dibutuhkan (selama peristiwa bencana).

   Untuk multi-situs aktif/aktif, failover berarti mengevakuasi Wilayah dan mengandalkan Wilayah aktif yang tersisa. Secara umum, Wilayah tersebut siap menerima lalu lintas. Untuk strategi Pilot Light dan Warm Standby, tindakan pemulihan Anda perlu menerapkan sumber daya yang hilang, seperti EC2 instance pada Gambar 20, ditambah sumber daya lain yang hilang.

   Untuk semua strategi di atas, Anda mungkin perlu mengubah instans hanya-baca basis data menjadi instans baca/tulis.

   Untuk pencadangan dan pemulihan, memulihkan data dari cadangan menciptakan sumber daya untuk data tersebut seperti EBS volume, instans RDS DB, dan tabel DynamoDB. Anda juga perlu memulihkan infrastruktur dan melakukan deployment kode. Anda dapat menggunakan AWS

Backup untuk memulihkan data di wilayah pemulihan. Lihat REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber untuk detail selengkapnya. Membangun kembali infrastruktur termasuk membuat sumber daya seperti EC2 instance selain Amazon Virtual Private Cloud VPC (Amazon), subnet, dan grup keamanan yang diperlukan. Anda dapat mengotomatiskan banyak proses pemulihan. Untuk mempelajari caranya, silakan lihat posting blog ini.

5. Tentukan dan implementasikan cara Anda akan merutekan kembali lalu lintas ke failover saat dibutuhkan (selama peristiwa bencana).

   Operasi failover ini dapat dimulai secara otomatis dan manual. Failover yang dimulai secara otomatis berdasarkan pemeriksaan kondisi atau alarm harus digunakan dengan hati-hati karena failover yang tidak perlu (alarm palsu) dapat dikenakan biaya seperti ketidaktersediaan dan kehilangan data. Oleh karena itu, Failover yang dimulai secara manual sering digunakan. Dalam kasus ini, Anda masih harus mengotomatiskan langkah failover, sehingga inisiasi manual akan seperti menekan tombol.

   Ada beberapa opsi manajemen lalu lintas yang perlu dipertimbangkan saat menggunakan AWS layanan. Salah satu opsinya adalah menggunakan Amazon Route 53. Menggunakan Amazon Route 53, Anda dapat mengaitkan beberapa titik akhir IP dalam satu atau lebih Wilayah AWS dengan nama domain Route 53. Untuk mengimplementasikan failover yang dimulai secara manual, Anda dapat menggunakan Amazon Application Recovery Controller, yang menyediakan pesawat data yang sangat tersedia API untuk mengalihkan lalu lintas ke Wilayah pemulihan. Saat mengimplementasikan failover, gunakan operasi bidang data dan hindari bidang kontrol yang dideskripsikan di REL11-BP04 Mengandalkan bidang data dan bukan bidang kontrol selama pemulihan.

   Untuk mempelajari lebih lanjut tentang ini dan opsi lainnya, lihat bagian ini dari Laporan Resmi Pemulihan Bencana.

6. Rancang rencana terkait bagaimana beban kerja akan failback.

   Failback adalah saat Anda mengembalikan operasi beban kerja ke Wilayah utama, setelah bencana berakhir. Penyediaan infrastruktur dan kode untuk Wilayah utama umumnya mengikuti langkah yang sama yang digunakan saat memulai, dengan mengandalkan infrastruktur sebagai kode dan pipeline deployment kode. Tantangan failback adalah mengembalikan penyimpanan data, dan memastikan konsistensi dengan Wilayah pemulihan dalam operasi.

Dalam keadaan gagal, database di Wilayah pemulihan hidup dan memiliki up-to-date data. Tujuannya kemudian adalah untuk menyinkronkan kembali dari Wilayah pemulihan ke Wilayah utama, memastikannya. up-to-date

Beberapa AWS layanan akan melakukan ini secara otomatis. Jika menggunakan tabel global Amazon DynamoDB, meskipun tabel di Wilayah utama menjadi tidak tersedia, saat kembali online, DynamoDB akan melanjutkan penulisan yang tertunda. Jika menggunakan Basis Data Global Amazon Aurora dan menggunakan failover terencana dan terkelola, topologi replikasi Aurora basis data global yang ada dipertahankan. Dengan demikian, instans baca/tulis sebelumnya di Wilayah utama akan menjadi replika dan menerima pembaruan dari Wilayah pemulihan.

Dalam kasus saat ini tidak dibuat otomatis, Anda perlu menetapkan ulang basis data di Wilayah utama sebagai replika dari basis data di Wilayah pemulihan. Dalam banyak kasus, ini akan melibatkan penghapusan basis data utama yang lama dan membuat replika yang baru.

Setelah failover, jika Anda dapat tetap menjalankannya di Wilayah pemulihan, pertimbangkan untuk membuat ini menjadi Wilayah utama yang baru. Anda masih harus melakukan semua langkah di atas untuk membuat Wilayah utama sebelumnya menjadi Wilayah pemulihan. Beberapa organisasi melakukan rotasi terjadwal, menukar Wilayah utama dan pemulihan secara berkala (misalnya setiap tiga bulan).

Semua langkah yang diperlukan untuk failover dan failback harus diperiksa di buku pedoman yang tersedia untuk semua anggota tim dan ditinjau secara berkala.

Ketika menggunakan Pemulihan Bencana Elastis, layanan akan membantu mengatur dan mengotomatiskan proses failback. Untuk detail selengkapnya, lihat Melakukan failback.

Tingkat upaya untuk Rencana Implementasi: Tinggi

Sumber daya

Praktik-praktik terbaik terkait:

- the section called “REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber”
- the section called “REL11-BP04 Mengandalkan bidang data dan bukan bidang kontrol selama pemulihan”

- the section called “REL13-BP01 Menentukan tujuan pemulihan untuk downtime dan kehilangan data”

Dokumen terkait:

- Blog Arsitektur AWS : Seri Pemulihan Bencana
- Pemulihan Bencana Beban Kerja di AWS: Pemulihan di Cloud (AWS Whitepaper)
- Opsi pemulihan bencana di cloud
- Bangun solusi backend aktif-aktif nirserver multi-wilayah dalam satu jam
- Backend nirserver multi-wilayah — dimuat ulang
- RDS: Mereplikasi Replika Baca di Seluruh Wilayah
- Route 53: Mengkonfigurasi DNS Failover
- S3: Replika Lintas-Wilayah
- Apa itu AWS Backup?
- Apa itu Pengontrol Pemulihan Aplikasi Amazon?
- Pemulihan Bencana Elastis AWS
- HashiCorpTerraform: Memulai - AWS
- APNMitra: mitra yang dapat membantu pemulihan bencana
- AWS Marketplace: produk yang dapat digunakan untuk pemulihan bencana

Video terkait:

- Pemulihan Bencana Beban Kerja pada AWS
- AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2
- Memulai AWS Elastic Disaster Recovery | Amazon Web Services

Contoh terkait:

- Lab Well-Architected - Pemulihan Bencana - Rangkaian lokakarya yang menggambarkan strategi DR

REL13-BP03 Uji implementasi pemulihan bencana untuk memvalidasi implementasi

Uji failover secara teratur ke situs pemulihan Anda untuk memverifikasi bahwa ia beroperasi dengan benar dan itu RTO dan RPO terpenuhi.

Anti-pola umum:

- Tidak pernah melakukan failover di lingkungan produksi.

Manfaat menerapkan praktik terbaik ini: Pengujian rencana pemulihan bencana secara rutin akan memverifikasi bahwa rencana tersebut akan berfungsi saat diperlukan, dan tim Anda tahu cara mengeksekusi strategi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pola untuk dihindari adalah mengembangkan jalur pemulihan yang sangat jarang dilakukan. Misalnya, Anda mungkin memiliki penyimpanan data sekunder yang digunakan untuk kueri hanya-baca. Saat Anda menulis ke penyimpanan data dan penyimpanan primer gagal, Anda mungkin ingin melakukan failover ke penyimpanan data sekunder. Jika Anda tidak sering menguji failover ini, Anda mungkin akan mendapati bahwa asumsi Anda tentang kemampuan penyimpanan data sekunder ternyata salah. Kapasitas sekunder, yang selama ini mungkin mencukupi saat terakhir Anda uji, mungkin sudah tidak mampu mentoleransi beban di bawah skenario ini. Pengalaman kami menunjukkan bahwa satu-satunya pemulihan kesalahan yang dapat diterapkan adalah jalur yang sering Anda uji. Inilah alasan memiliki sedikit jalur pemulihan adalah yang terbaik. Anda dapat membuat pola pemulihan dan mengujinya secara rutin. Jika Anda memiliki jalur pemulihan yang kompleks atau kritis, Anda tetap perlu secara rutin melatih kegagalan tersebut dalam lingkungan produksi agar Anda yakin bahwa jalur pemulihan tersebut berfungsi. Pada contoh yang baru saja kita bahas, Anda harus melakukan failover ke penyimpanan siaga secara rutin, terlepas ada tidaknya kebutuhan.

Langkah-langkah implementasi

1. Rekayasa beban kerja Anda untuk pemulihan. Uji jalur pemulihan Anda secara rutin. Komputasi yang berorientasi pada pemulihan mengidentifikasi karakteristik dalam sistem yang meningkatkan pemulihan: isolasi dan redundansi, kemampuan di seluruh sistem untuk membatalkan perubahan, kemampuan untuk memantau dan menentukan kondisi, kemampuan untuk menyediakan diagnostik, pemulihan otomatis, desain modular, dan kemampuan untuk memulai ulang. Latih jalur pemulihan untuk memverifikasi bahwa Anda dapat menyelesaikan pemulihan dalam waktu

yang ditentukan ke status yang ditentukan. Gunakan runbook selama pemulihan ini untuk mendokumentasikan masalah dan menemukan solusinya sebelum pengujian berikutnya.

2. Untuk beban kerja EC2 berbasis Amazon, gunakan AWS Elastic Disaster Recoveryuntuk menerapkan dan meluncurkan instans latihan untuk strategi DR Anda. AWS Elastic Disaster Recovery menyediakan kemampuan untuk menjalankan latihan secara efisien, yang membantu Anda mempersiapkan acara failover. Anda juga dapat sering-sering meluncurkan instans menggunakan Pemulihan Bencana Elastis untuk tujuan pengujian dan latihan tanpa mengarahkan ulang lalu lintas.

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu pemulihan bencana
- Blog Arsitektur AWS : Seri Pemulihan Bencana
- AWS Marketplace: produk yang dapat digunakan untuk pemulihan bencana
- AWS Elastic Disaster Recovery
- Pemulihan Bencana Beban Kerja di AWS: Pemulihan di Cloud (AWS Whitepaper)
- AWS Elastic Disaster Recovery Mempersiapkan Failover
- Proyek komputasi Berkeley/Stanford berorientasi pemulihan
- Apa itu AWS Fault Injection Simulator?

Video terkait:

- AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah
- AWS re:invent 2019: Backup-and-restore dan solusi pemulihan bencana dengan AWS

Contoh terkait:

- Lab Well-Architected - Pengujian Ketangguhan

REL13-BP04 Mengelola penyimpangan konfigurasi di situs atau Wilayah DR

Pastikan infrastruktur, data, dan konfigurasi sesuai dengan yang diperlukan di lokasi atau Wilayah DR. Misalnya, periksa itu AMIs dan kuota layanan mutakhir.

AWS Config terus memantau dan merekam konfigurasi AWS sumber daya Anda. Hal ini dapat mendeteksi drift dan memanggil AWS Systems Manager Automation untuk memperbaikinya dan meningkatkan alarm. AWS CloudFormation juga dapat mendeteksi penyimpangan di tumpukan yang telah Anda terapkan.

Anti-pola umum:

- Gagal melakukan pembaruan pada lokasi pemulihan Anda, saat Anda membuat perubahan konfigurasi atau infrastruktur pada lokasi primer.
- Tidak mempertimbangkan potensi pembatasan (seperti perbedaan layanan) di lokasi primer dan pemulihan Anda.

Manfaat menerapkan praktik terbaik ini: Lingkungan DR yang sesuai dengan lingkungan Anda saat ini menjamin pemulihan yang lengkap.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Pastikan pipeline pengiriman Anda menjangkau lokasi primer dan cadangan Anda. Pipeline pengiriman untuk men-deploy aplikasi ke lingkungan produksi harus menyebarkan ke semua lokasi strategi pemulihan bencana yang ditentukan, termasuk lingkungan pengembangan dan pengujian.
- Izinkan AWS Config untuk melacak lokasi drift potensial. Gunakan AWS Config aturan untuk membuat sistem yang menegakkan strategi pemulihan bencana Anda dan menghasilkan peringatan saat mendeteksi penyimpangan.
  - Memulihkan Sumber Daya yang Tidak Sesuai dengan AWSAturan AWS Config
  - AWS Otomatisasi Systems Manager
- Gunakan AWS CloudFormation untuk menyebarkan infrastruktur Anda. AWS CloudFormation dapat mendeteksi penyimpangan antara apa yang ditentukan CloudFormation template Anda dan apa yang sebenarnya digunakan.
  - AWS CloudFormation: Mendeteksi Drift di Seluruh Tumpukan CloudFormation

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu pemulihan bencana

- Blog Arsitektur AWS : Seri Pemulihan Bencana
- AWS CloudFormation: Mendeteksi Drift di Seluruh Tumpukan CloudFormation
- AWS Marketplace: produk yang dapat digunakan untuk pemulihan bencana
- AWS Otomatisasi Systems Manager
- Pemulihan Bencana Beban Kerja di AWS: Pemulihan di Cloud (AWS Whitepaper)
- Bagaimana cara mengimplementasikan solusi Manajemen Konfigurasi Infrastruktur di AWS?
- Memulihkan Sumber Daya yang Tidak Sesuai dengan AWSAturan AWS Config

Video terkait:

- AWS re: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2

REL13-BP05 Mengotomatiskan pemulihan

Gunakan AWS atau alat pihak ketiga untuk mengotomatiskan pemulihan sistem dan mengarahkan lalu lintas ke situs atau Wilayah DR.

Berdasarkan pemeriksaan kesehatan yang dikonfigurasi, AWS layanan, seperti Elastic Load Balancing dan AWS Auto Scaling, dapat mendistribusikan beban ke Availability Zone yang sehat sementara layanan, seperti Amazon Route 53 dan AWS Global Accelerator, dapat merutekan beban ke kondisi sehat. Wilayah AWS Amazon Application Recovery Controller membantu Anda mengelola dan mengoordinasikan failover menggunakan fitur pemeriksaan kesiapan dan kontrol perutean. Fitur-fitur ini terus memantau kemampuan aplikasi Anda untuk pulih dari kegagalan, sehingga Anda dapat mengontrol pemulihan aplikasi di beberapa Wilayah AWS, Availability Zone, dan di lokasi.

Untuk beban kerja pada pusat data fisik atau virtual atau cloud pribadi yang ada, AWS Elastic Disaster Recovery memungkinkan perusahaan untuk menyiapkan strategi pemulihan bencana otomatis di AWS. Pemulihan Bencana Elastis juga mendukung pemulihan bencana Lintas-wilayah dan Lintas-zona-ketersediaan di AWS.

Anti-pola umum:

- Mengimplementasikan failover dan failback otomatis yang serupa dapat menyebabkan flapping saat kesalahan terjadi.

Manfaat menerapkan praktik terbaik ini: Pemulihan otomatis mengurangi waktu pemulihan dengan menghilangkan peluang untuk kesalahan manual.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Otomatiskan jalur pemulihan. Untuk waktu pemulihan yang singkat, ikuti rencana pemulihan bencana Anda agar sistem TI Anda kembali online dengan cepat jika terjadi gangguan.
  - Gunakan Pemulihan Bencana Elastis untuk Failover dan Failback. Elastic Disaster Recovery terus mereplikasi mesin Anda (termasuk sistem operasi, konfigurasi status sistem, database, aplikasi, dan file) ke area pementasan berbiaya rendah di wilayah target Akun AWS dan pilihan Anda. Jika terjadi bencana, setelah memilih untuk melakukan pemulihan menggunakan Pemulihan Bencana Elastis, Pemulihan Bencana Elastis akan mengotomatiskan konversi server yang direplikasi menjadi beban kerja yang sepenuhnya disediakan di Wilayah pemulihan Anda pada AWS.
    - Menggunakan Pemulihan Bencana Elastis untuk Failover dan Failback
    - AWS Elastic Disaster Recovery sumber daya

Sumber daya

Dokumen terkait:

- APNMitra: mitra yang dapat membantu pemulihan bencana
- Blog Arsitektur AWS : Seri Pemulihan Bencana
- AWS Marketplace: produk yang dapat digunakan untuk pemulihan bencana
- AWS Otomatisasi Systems Manager
- AWS Elastic Disaster Recovery
- Pemulihan Bencana Beban Kerja di AWS: Pemulihan di Cloud (AWS Whitepaper)

Video terkait:

- AWS RE: Invent 2018: Pola Arsitektur untuk Aplikasi Aktif-Aktif Multi-Wilayah (09-R2) ARC2

# Efisiensi kinerja

Pilar Efisiensi Kinerja mencakup kemampuan untuk menggunakan sumber daya cloud secara efisien untuk memenuhi persyaratan-persyaratan kinerja, dan untuk mempertahankan efisiensi tersebut

seiring dengan perubahan permintaan dan perkembangan teknologi yang terjadi. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Efisiensi Kinerja.

Area praktik terbaik

- Pemilihan arsitektur
- Komputasi dan perangkat keras
- Manajemen data
- Jaringan dan Pengiriman Konten
- Proses dan budaya

## Pemilihan arsitektur

Pertanyaan

- PERF1. Bagaimana cara memilih sumber daya dan arsitektur cloud yang sesuai untuk beban kerja Anda?

### PERF1. Bagaimana cara memilih sumber daya dan arsitektur cloud yang sesuai untuk beban kerja Anda?

Solusi yang optimal bervariasi untuk beban kerja tertentu, dan solusi sering kali menggabungkan beberapa pendekatan. Beban kerja yang dirancang dengan baik menggunakan beberapa solusi dan memungkinkan berbagai fitur guna meningkatkan kinerja.

Praktik terbaik

- PERF01-BP01 Pelajari dan pahami layanan dan fitur cloud yang tersedia
- PERF01-BP02 Gunakan panduan dari penyedia cloud Anda atau mitra yang tepat untuk mempelajari pola arsitektur dan praktik terbaik
- PERF01-BP03 Faktor biaya ke dalam keputusan arsitektur
- PERF01-BP04 Mengevaluasi bagaimana trade-off berdampak pada pelanggan dan efisiensi arsitektur
- PERF01-BP05 Menggunakan kebijakan dan arsitektur referensi
- PERF01-BP06 Gunakan benchmarking untuk mendorong keputusan arsitektur
- PERF01-BP07 Gunakan pendekatan berbasis data untuk pilihan arsitektur

PERF01-BP01 Pelajari dan pahami layanan dan fitur cloud yang tersedia

Terus pelajari dan temukan layanan serta konfigurasi yang tersedia yang membantu Anda mengambil keputusan arsitektur yang lebih baik dan meningkatkan efisiensi kinerja dalam arsitektur beban kerja Anda.

Anti-pola umum:

- Anda menggunakan cloud sebagai pusat data kolokasi.
- Anda tidak memodernisasi aplikasi Anda setelah migrasi ke cloud.
- Anda hanya menggunakan satu tipe penyimpanan untuk semua hal yang perlu dipertahankan.
- Anda menggunakan tipe instans yang paling sesuai dengan standar Anda saat ini, tetapi lebih besar dari yang diperlukan.
- Anda melakukan deployment dan mengelola teknologi yang tersedia sebagai layanan terkelola.

Manfaat menerapkan praktik terbaik ini: Dengan mempertimbangkan layanan dan konfigurasi baru, Anda mungkin dapat meningkatkan kinerja, mengurangi biaya, dan mengoptimalkan upaya yang diperlukan untuk memelihara beban kerja Anda. Ini juga dapat membantu Anda mempercepat time-to-value untuk produk berkemampuan cloud.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS terus merilis layanan dan fitur baru yang dapat meningkatkan kinerja dan mengurangi biaya beban kerja cloud. Tetap up-to-date menggunakan layanan dan fitur baru ini sangat penting untuk menjaga kemanjuran kinerja di cloud. Modernisasi arsitektur beban kerja juga membantu Anda mempercepat produktivitas, mendorong inovasi, dan membuka lebih banyak peluang pertumbuhan.

Langkah-langkah implementasi

- Buat inventaris arsitektur dan perangkat lunak beban kerja untuk layanan terkait. Tentukan kategori produk mana yang akan dipelajari lebih lanjut.
- Jelajahi AWS penawaran untuk mengidentifikasi dan mempelajari tentang layanan dan opsi konfigurasi yang relevan yang dapat membantu Anda meningkatkan kinerja dan mengurangi biaya dan kompleksitas operasional.
  - Amazon Web Services Cloud

  - AWS Akademi
  - Apa yang baru dengan AWS?
  - AWS Blog
  - AWS Pembangun Keterampilan
  - AWS Acara dan Webinar
  - AWS Training dan Sertifikasi
  - AWS Youtube Kanal
  - AWS Lokakarya
  - Komunitas AWS
- Gunakan Amazon Q untuk mendapatkan informasi dan saran yang relevan tentang layanan.
- Gunakan lingkungan sandbox (non-produksi) untuk mempelajari dan bereksperimen dengan layanan baru tanpa dikenakan biaya tambahan.
- Terus pelajari layanan dan fitur cloud baru.

Sumber daya

Dokumen terkait:

- Ikhtisar Amazon Web Services
- EC2Fitur Amazon
- Belajar step-by-step dengan Rencana Pembelajaran AWS Mitra
- AWS Pelatihan dan Sertifikasi
- Jalur pembelajaran saya untuk menjadi arsitek AWS solusi
- AWS Pusat Arsitektur
- AWS Partner Network
- AWS Pustaka Solusi
- AWS Pusat Pengetahuan
- Membangun aplikasi modern di AWS

Video terkait:

- AWS re:invent 2023 - Apa yang baru dengan Amazon EC2

- AWS re:invent 2022 - Kurangi biaya operasional dan infrastruktur Anda dengan Amazon ECS
- AWS re:invent 2023 - Membangun dengan efisiensi, kelincahan & inovasi cloud dengan AWS
- AWS re:invent 2022 - Terapkan model ML untuk inferensi dengan kinerja tinggi dan biaya rendah
- Ini Arsitektur saya

Contoh terkait:

- AWS Sampel
- AWS SDKContoh

PERF01-BP02 Gunakan panduan dari penyedia cloud Anda atau mitra yang tepat untuk mempelajari pola arsitektur dan praktik terbaik

Gunakan sumber daya perusahaan cloud, seperti dokumentasi, arsitek solusi, layanan profesional, atau partner yang tepat untuk memandu keputusan-keputusan Anda yang berkaitan dengan arsitektur. Semua sumber daya ini membantu meninjau dan meningkatkan arsitektur Anda untuk kinerja yang optimal.

Anti-pola umum:

- Anda gunakan AWS sebagai penyedia cloud umum.
- Anda menggunakan AWS layanan dengan cara yang tidak dirancang untuknya.
- Anda mengikuti semua panduan tanpa mempertimbangkan konteks bisnis Anda.

Manfaat menerapkan praktik terbaik ini: Menggunakan panduan dari sebuah penyedia cloud atau mitra yang tepat dapat membantu Anda membuat pilihan arsitektur yang tepat untuk beban kerja Anda dan memberi Anda kepercayaan diri dalam keputusan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS menawarkan berbagai panduan, dokumentasi, dan sumber daya yang dapat membantu Anda membangun dan mengelola beban kerja cloud yang efisien. AWS dokumentasi menyediakan contoh kode, tutorial, dan penjelasan layanan terperinci. Selain dokumentasi, AWS menyediakan program pelatihan dan sertifikasi, arsitek solusi, dan layanan profesional yang dapat membantu pelanggan menjelajahi berbagai aspek layanan cloud dan menerapkan arsitektur cloud yang efisien. AWS

Manfaatkan semua sumber daya ini untuk mendapatkan wawasan tentang pengetahuan dan praktik terbaik yang berharga, menghemat waktu, dan mencapai hasil yang lebih baik di AWS Cloud.

Langkah-langkah implementasi

- Tinjau AWS dokumentasi dan panduan dan ikuti praktik terbaik. Semua sumber daya ini dapat membantu Anda memilih dan mengonfigurasi layanan secara efektif dan mencapai kinerja yang lebih baik.
  - AWS dokumentasi (seperti panduan pengguna dan whitepaper)
  - AWS Blog
  - AWS Training dan Sertifikasi
  - AWS Youtube Kanal
- Bergabunglah dengan acara AWS mitra (seperti KTT AWS Global, AWS Re:Invent, grup pengguna, dan lokakarya) untuk belajar dari AWS para ahli tentang praktik terbaik untuk menggunakan layanan. AWS
  - Belajar step-by-step dengan Rencana Pembelajaran AWS Mitra
  - AWS Acara dan Webinar
  - AWS Lokakarya
  - AWS Komunitas
- Hubungi bantuan ketika Anda membutuhkan panduan tambahan atau informasi produk. AWS AWS Solusi Arsitek dan Layanan AWS Profesional memberikan panduan untuk implementasi solusi. AWS Mitra memberikan AWS keahlian untuk membantu Anda membuka kelincahan dan inovasi untuk bisnis Anda.
- Gunakan AWS Support jika Anda membutuhkan dukungan teknis untuk menggunakan layanan secara efektif. Paket Support kami dirancang untuk memberi Anda perpaduan alat yang tepat dan akses ke keahlian sehingga Anda dapat berhasil AWS sambil mengoptimalkan kinerja, mengelola risiko, dan menjaga biaya tetap terkendali.

Sumber daya

Dokumen terkait:

- Pusat Arsitektur AWS
- AWS Partner Network
- Pustaka Solusi AWS

- Pusat Pengetahuan AWS
- Dukungan Perusahaan AWS

Video terkait:

- Ini Arsitektur saya
- AWS re: invent 2023 - Pola berbasis peristiwa tingkat lanjut dengan Amazon EventBridge
- AWS re: invent 2023 - Menerapkan pola desain terdistribusi pada AWS
- AWS re:invent 2023 - Arsitektur aplikasi sebagai kode

Contoh terkait:

- Sampel AWS
- AWS SDKContoh
- AWS Arsitektur Referensi Analytics

PERF01-BP03 Faktor biaya ke dalam keputusan arsitektur

Pertimbangkan biaya dalam keputusan arsitektur Anda untuk meningkatkan pemanfaatan sumber daya dan efisiensi kinerja beban kerja cloud Anda. Ketika Anda menyadari implikasi biaya dari beban kerja cloud Anda, Anda kemungkinan akan memanfaatkan sumber daya yang efisien dan mengurangi praktik pemborosan.

Anti-pola umum:

- Anda hanya menggunakan satu kelompok instans.
- Anda tidak mengevaluasi solusi berlisensi dibandingkan dengan solusi sumber terbuka.
- Anda tidak menentukan kebijakan siklus hidup penyimpanan.
- Anda tidak meninjau layanan dan fitur baru dari AWS Cloud.
- Anda hanya menggunakan penyimpanan blok.

Manfaat menerapkan praktik terbaik ini: Dengan mempertimbangkan biaya dalam pengambilan keputusan, Anda dapat menggunakan sumber daya yang lebih efisien dan mengeksplorasi investasi lainnya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Mengoptimalkan beban kerja untuk biaya dapat meningkatkan pemanfaatan sumber daya dan menghindari pemborosan dalam beban kerja cloud. Mempertimbangkan biaya dalam keputusan arsitektur biasanya mencakup penyesuaian ukuran yang tepat untuk komponen beban kerja dan menghadirkan elastisitas, yang menghasilkan peningkatan efisiensi kinerja beban kerja cloud.

Langkah-langkah implementasi

- Tetapkan sasaran biaya seperti batas anggaran untuk beban kerja cloud Anda.
- Identifikasi komponen utama (seperti instans dan penyimpanan) yang menambah biaya beban kerja Anda. Anda dapat menggunakan AWS Pricing Calculator dan AWS Cost Explorer untuk mengidentifikasi pendorong biaya utama dalam beban kerja Anda.
- Pahami model penetapan harga di cloud, seperti Sesuai Permintaan, Instans Terpesan, Savings Plans, dan Instans Spot.
- Gunakan praktik terbaik optimasi biaya Well-Architected untuk mengoptimalkan komponen utama ini untuk biaya.
- Teruslah memantau dan menganalisis biaya untuk mengidentifikasi peluang pengoptimalan biaya dalam beban kerja Anda.
  - Gunakan AWS Budgets untuk mendapatkan pemberitahuan adanya biaya yang tidak dapat diterima.
  - Gunakan AWS Compute Optimizer atau AWS Trusted Advisor untuk mendapatkan rekomendasi pengoptimalan biaya.
  - Gunakan AWS Cost Anomaly Detection untuk mendapatkan deteksi anomali biaya dan analisis akar masalah secara otomatis.

Sumber daya

Dokumen terkait:

- Apa itu AWS Billing and Cost Management?
- Optimalisasi Biaya dengan AWS
- Memilih strategi manajemen AWS biaya
- Panduan Pemula untuk Manajemen AWS Biaya

- Tinjauan Mendetail tentang Dasbor Inteligensi Biaya
- Pusat Arsitektur AWS
- Pustaka Solusi AWS
- Pusat Pengetahuan AWS

Video terkait:

- Ini Arsitektur saya
- AWS Re:invent 2023 - Apa yang baru dengan optimasi biaya AWS
- AWS re:invent 2023 - Optimalkan biaya dan kinerja dan lacak kemajuan menuju mitigasi
- AWS RE: invent 2023 - AWS praktik terbaik pengoptimalan biaya penyimpanan
- AWS Re:invent 2023 - Optimalkan biaya di lingkungan multi-akun Anda

Contoh terkait:

- AWS Compute Optimizer Kode demo
- Lokakarya Optimisasi Biaya
- Panduan Implementasi Teknis Manajemen Keuangan Cloud
- Pengoptimalan perusahaan rintisan: Menyetel kinerja aplikasi untuk efisiensi maksimum
- Lokakarya Pengoptimalan Nirserver (Kinerja dan Biaya)
- Menskalakan arsitektur hemat biaya

PERF01-BP04 Mengevaluasi bagaimana trade-off berdampak pada pelanggan dan efisiensi arsitektur

Saat mengevaluasi peningkatan terkait kinerja, tentukan pilihan mana yang berdampak pada efisiensi beban kerja dan pelanggan Anda. Misalnya, jika menggunakan penyimpanan data nilai-kunci dapat meningkatkan kinerja sistem, penting untuk mengevaluasi bagaimana dampak sifat eventual consistency-nya nanti terhadap pelanggan.

Anti-pola umum:

- Anda berasumsi bahwa semua kinerja yang dimiliki harus diimplementasikan, meskipun ada kompromi untuk implementasi.
- Anda hanya mengevaluasi perubahan beban kerja ketika masalah kinerja telah mencapai titik kritis.

Manfaat menerapkan praktik terbaik ini: Ketika Anda mengevaluasi potensi peningkatan terkait performa, Anda harus menentukan apakah kompromi untuk perubahan dapat diterima dengan persyaratan beban kerja. Dalam beberapa kasus, Anda mungkin harus mengimplementasikan beberapa kontrol tambahan untuk mengimbangi kompensasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Identifikasi area kritis dalam arsitektur Anda dalam hal dampak terhadap kinerja dan pelanggan. Tentukan cara Anda mewujudkan peningkatan, kompromi seperti apa yang ditimbulkan peningkatan, serta bagaimana pengaruhnya terhadap sistem dan pengalaman pengguna. Misalnya, mengimplementasikan pembuatan cache data dapat membantu meningkatkan kinerja secara signifikan tetapi memerlukan strategi yang jelas terkait cara dan waktu untuk memperbarui atau menonaktifkan data yang di-cache guna mencegah perilaku sistem yang tidak sesuai.

Langkah-langkah implementasi

- Pahami persyaratan beban kerja Anda danSLAs.
- Tentukan faktor evaluasi secara jelas. Faktor-faktor mungkin berhubungan dengan biaya, keandalan, keamanan, dan kinerja beban kerja Anda.
- Pilih arsitektur dan layanan yang dapat memenuhi kebutuhan Anda.
- Melakukan eksperimen dan bukti konsep (POCs) untuk mengevaluasi faktor trade-off dan dampak pada pelanggan dan efisiensi arsitektur. Biasanya, beban kerja dengan ketersediaan tinggi, berkinerja tinggi, dan aman mengonsumsi lebih banyak sumber daya cloud sekaligus memberikan pengalaman pelanggan yang lebih baik. Pahami kompromi antara kompleksitas, kinerja, dan biaya beban kerja Anda. Umumnya, ketika dua faktor diprioritaskan, faktor ketiga akan dikorbankan.

Sumber daya

Dokumen terkait:

- Amazon Builders' Library
- Amazon QuickSight KPIs
- Amazon CloudWatch RUM
- Dokumentasi X-Ray
- Memahami pola ketahanan dan kompromi untuk merancang secara efisien di cloud

Video terkait:

- Optimalkan aplikasi melalui Amazon CloudWatch RUM
- AWS RE: invent 2023 - Kapasitas, ketersediaan, efisiensi biaya: Pilih tiga
- AWS RE: invent 2023 - Pola integrasi lanjutan & trade-off untuk sistem yang digabungkan secara longgar

Contoh terkait:

- Ukur waktu buka halaman dengan Amazon CloudWatch Synthetics
- Klien CloudWatch RUM Web Amazon

PERF01-BP05 Menggunakan kebijakan dan arsitektur referensi

Gunakan kebijakan internal dan arsitektur referensi yang ada saat memilih layanan dan konfigurasi agar lebih efisien saat merancang dan mengimplementasikan beban kerja Anda.

Anti-pola umum:

- Anda mengizinkan berbagai macam teknologi yang berdampak pada biaya manajemen biaya perusahaan.

Manfaat menerapkan praktik terbaik ini: Dengan menetapkan kebijakan untuk pilihan arsitektur, teknologi, dan vendor, keputusan dapat diambil dengan lebih cepat.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Adanya kebijakan internal dalam memilih sumber daya dan arsitektur memberikan standar dan pedoman untuk diikuti ketika membuat pilihan arsitektur. Pedoman tersebut merampingkan proses pengambilan keputusan saat memilih layanan cloud yang tepat dan dapat membantu meningkatkan efisiensi kinerja. Lakukan deployment beban kerja Anda menggunakan arsitektur referensi atau kebijakan. Integrasikan layanan ke dalam deployment cloud, lalu gunakan pengujian kinerja untuk memastikan bahwa Anda dapat terus memenuhi persyaratan kinerja.

Langkah-langkah implementasi

- Pahami dengan jelas persyaratan beban kerja cloud Anda.

- Tinjau kebijakan internal dan eksternal untuk mengidentifikasi kebijakan yang paling relevan.
- Gunakan arsitektur referensi yang sesuai yang disediakan oleh AWS atau praktik terbaik industri Anda.
- Buat rangkaian yang terdiri dari kebijakan, standar, arsitektur referensi, dan pedoman preskriptif untuk situasi umum. Tindakan tersebut memungkinkan tim Anda bergerak lebih cepat. Sesuaikan aset untuk bidang Anda jika perlu.
- Validasi kebijakan dan arsitektur referensi ini untuk beban kerja Anda di lingkungan sandbox.
- Ikuti up-to-date standar dan AWS pembaruan industri untuk memastikan kebijakan dan arsitektur referensi membantu mengoptimalkan beban kerja cloud Anda.

Sumber daya

Dokumen terkait:

- Pusat Arsitektur AWS
- AWS Partner Network
- Pustaka Solusi AWS
- Pusat Pengetahuan AWS
- AWS Blog Arsitektur

Video terkait:

- Ini Arsitektur saya
- AWS Re: invent 2022 - Percepat nilai untuk bisnis Anda dengan & arsitektur referensi SAP AWS

Contoh terkait:

- Sampel AWS
- AWS SDKContoh

PERF01-BP06 Gunakan benchmarking untuk mendorong keputusan arsitektur

Lakukan tolok ukur pada kinerja beban kerja yang ada untuk memahami kinerjanya di cloud dan mendorong keputusan arsitektur berdasarkan data tersebut.

Anti-pola umum:

- Anda mengandalkan tolok ukur umum yang tidak mewakili karakteristik beban kerja Anda.
- Anda bergantung pada persepsi dan tanggapan pelanggan sebagai satu-satunya tolok ukur.

Manfaat menerapkan praktik terbaik ini: Melakukan tolok ukur terhadap implementasi Anda saat ini akan memungkinkan Anda untuk mengukur peningkatan kinerja yang berhasil dicapai.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Gunakan benchmarking dengan pengujian sintetis untuk menilai kinerja komponen beban kerja Anda. Benchmarking umumnya dapat disiapkan dengan lebih cepat daripada pengujian beban dan digunakan untuk mengevaluasi teknologi untuk komponen tertentu. Benchmarking sering digunakan pada awal proyek baru, saat Anda tidak memiliki solusi lengkap untuk memuat pengujian.

Anda dapat membuat tes benchmark kustom Anda sendiri atau menggunakan tes standar industri, seperti TPC-DS, untuk membandingkan beban kerja Anda. Tolok ukur industri sangat membantu saat memperbandingkan lingkungan. Tolok ukur kustom bermanfaat untuk menargetkan jenis operasi tertentu yang ingin dibuat dalam arsitektur.

Saat melakukan tolok ukur, penting untuk menyiapkan lingkungan terlebih dahulu untuk memastikan hasil yang valid. Jalankan tolok ukur yang sama beberapa kali untuk memastikan Anda memperoleh variasi apa pun dari waktu ke waktu.

Karena tolok ukur umumnya lebih cepat untuk menjalankan pengujian daripada memuatnya, maka tolok ukur dapat digunakan terlebih dahulu dalam deployment pipeline dan memberikan umpan balik pada deviasi kinerja. Saat Anda mengevaluasi perubahan yang signifikan dalam komponen atau layanan, tolok ukur dapat menjadi cara cepat guna menentukan apakah perubahan memang perlu dibuat. Menggunakan benchmarking bersama dengan pengujian beban begitu penting karena pengujian beban memberi tahu Anda tentang bagaimana kinerja beban kerja Anda dalam produksi.

Langkah-langkah implementasi

- Rencanakan dan tentukan:
  - Tentukan tujuan, baseline, skenario pengujian, metrik (seperti CPU pemanfaatan, latensi, atau throughput), dan untuk tolok ukur Anda. KPIs
  - Fokus pada persyaratan pengguna dalam hal pengalaman pengguna dan faktor-faktor seperti waktu respons dan aksesibilitas.

    - Identifikasi alat tolok ukur yang sesuai dengan beban kerja Anda. Anda dapat menggunakan AWS layanan seperti Amazon CloudWatch atau alat pihak ketiga yang kompatibel dengan beban kerja Anda.
- Konfigurasi dan persiapkan:
    - Siapkan lingkungan Anda dan konfigurasikan sumber daya Anda.
    - Implementasikan pemantauan dan pembuatan log untuk merekam hasil pengujian.
- Lakukan tolok ukur dan pemantauan:
    - Lakukan pengujian tolok ukur Anda dan pantau metrik selama pengujian.
- Analisis dan dokumentasikan:
    - Dokumentasikan proses dan temuan tolok ukur Anda.
    - Analisis hasil untuk mengidentifikasi hambatan, tren, dan area perbaikan.
    - Gunakan hasil pengujian untuk mengambil keputusan arsitektur dan menyesuaikan beban kerja Anda. Termasuk di dalamnya mungkin adalah mengubah layanan atau mengadopsi fitur baru.
- Optimalkan dan ulangi:
    - Sesuaikan konfigurasi dan alokasi sumber daya berdasarkan tolok ukur Anda.
    - Uji ulang beban kerja Anda setelah penyesuaian untuk memvalidasi perbaikan Anda.
    - Dokumentasikan pembelajaran Anda, dan ulangi proses untuk mengidentifikasi area perbaikan lainnya.

Sumber daya

Dokumen terkait:

- AWS Pusat Arsitektur
- AWS Partner Network
- AWS Pustaka Solusi
- AWS Pusat Pengetahuan
- Amazon CloudWatch RUM
- Amazon CloudWatch Synthetics
- Alur kerja genomik, Bagian 5: penolokukuran otomatis
- Benchmark dan optimalkan penerapan titik akhir di Amazon SageMaker JumpStart

Video terkait:

- AWS Re: invent 2023 - Pembandingan dingin dimulai AWS Lambda
- Penolokukuran layanan stateful di cloud
- Ini Arsitektur saya
- Optimalkan aplikasi melalui Amazon CloudWatch RUM
- Demo dari Amazon CloudWatch Synthetics

Contoh terkait:

- AWS Sampel
- AWS SDKContoh
- Pengujian Beban Terdistribusi
- Ukur waktu buka halaman dengan Amazon CloudWatch Synthetics
- Klien CloudWatch RUM Web Amazon

PERF01-BP07 Gunakan pendekatan berbasis data untuk pilihan arsitektur

Tentukan pendekatan yang jelas dan berbasis data untuk pilihan arsitektur guna memastikan layanan dan konfigurasi cloud yang tepat digunakan untuk memenuhi kebutuhan bisnis spesifik Anda.

Anti-pola umum:

- Anda berasumsi bahwa arsitektur Anda saat ini statis dan tidak perlu diperbarui dari waktu ke waktu.
- Pilihan arsitektur Anda didasarkan pada tebakan dan asumsi.
- Anda memperkenalkan perubahan arsitektur seiring waktu tanpa justifikasi.

Manfaat menerapkan praktik terbaik ini: Dengan memiliki pendekatan yang terdefinisi dengan baik dalam membuat pilihan arsitektur, Anda menggunakan data untuk memengaruhi desain beban kerja Anda dan mengambil keputusan berdasarkan informasi dari waktu ke waktu.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Gunakan pengalaman internal dan pengetahuan tentang cloud, atau sumber daya eksternal seperti kasus penggunaan yang dipublikasi atau laporan resmi untuk memilih sumber daya dan layanan di arsitektur Anda. Anda harus memiliki proses yang terdefinisi dengan baik yang mendorong eksperimen dan tolok ukur dengan layanan yang bisa digunakan pada beban kerja Anda.

Backlog untuk beban kerja kritis tidak boleh hanya terdiri dari cerita pengguna yang memberikan fungsionalitas yang relevan dengan bisnis dan pengguna, melainkan juga harus berisi cerita teknis yang membentuk landasan arsitektur untuk beban kerja. Landasan ini didasarkan pada kemajuan teknologi baru serta layanan baru dan mengadopsinya berdasarkan data dan pembenaran yang tepat. Hal ini memastikan bahwa arsitektur tetap relevan di masa depan dan tidak jalan di tempat.

Langkah-langkah implementasi

- Lakukan interaksi dengan pemangku kepentingan utama untuk menentukan persyaratan beban kerja, termasuk kinerja, ketersediaan, dan pertimbangan biaya. Pertimbangkan faktor-faktor seperti jumlah pengguna dan pola penggunaan untuk beban kerja Anda.
- Ciptakan landasan arsitektur atau backlog teknologi yang diprioritaskan bersamaan dengan backlog fungsional.
- Evaluasi dan nilai berbagai layanan cloud (untuk detail selengkapnya, lihat PERF01-BP01 Pelajari dan pahami layanan dan fitur cloud yang tersedia).
- Jelajahi pola-pola arsitektur yang berbeda, seperti layanan mikro atau nirserver, yang memenuhi persyaratan kinerja Anda (untuk detail selengkapnya, lihat PERF01-BP02 Gunakan panduan dari penyedia cloud Anda atau mitra yang tepat untuk mempelajari pola arsitektur dan praktik terbaik).

- Konsultasikan dengan tim lain, diagram arsitektur, dan sumber daya, seperti Arsitek AWS Solusi, Pusat AWS Arsitektur, dan AWS Partner Network, untuk membantu Anda memilih arsitektur yang tepat untuk beban kerja Anda.

- Tentukan metrik kinerja seperti throughput dan waktu respons yang dapat membantu Anda mengevaluasi kinerja beban kerja Anda.
- Lakukan eksperimen dan gunakan metrik yang ditentukan untuk memvalidasi kinerja arsitektur yang dipilih.
- Teruslah memantau dan melakukan penyesuaian sesuai kebutuhan untuk mempertahankan kinerja optimal arsitektur Anda.

- Dokumentasikan arsitektur dan keputusan pilihan Anda sebagai referensi untuk pembaruan dan pembelajaran di masa mendatang.
- Teruslah meninjau dan memperbarui pendekatan pemilihan arsitektur berdasarkan pembelajaran, teknologi baru, dan metrik yang menunjukkan kebutuhan perubahan atau masalah dalam pendekatan saat ini.

Sumber daya

Dokumen terkait:

- Pustaka Solusi AWS
- Pusat Pengetahuan AWS
- Pola Arsitektur untuk Membangun Aplikasi Berbasis End-to-End Data AWS

Video terkait:

- Ini Arsitektur saya
- AWS Re:invent 2021 - Perusahaan berbasis data: Beranjak dari visi ke nilai
- AWS re:invent 2022 - Memberikan arsitektur yang berkelanjutan dan berkinerja tinggi
- AWS re:invent 2023 - Optimalkan biaya dan kinerja dan lacak kemajuan menuju mitigasi
- AWS re:invent 2022 - AWS optimasi: Langkah-langkah yang dapat ditindaklanjuti untuk hasil langsung

Contoh terkait:

- Sampel AWS
- AWS SDKContoh

## Komputasi dan perangkat keras

Pertanyaan

- PERF2. Bagaimana cara memilih dan menggunakan sumber daya komputasi dalam beban kerja Anda?

## PERF2. Bagaimana cara memilih dan menggunakan sumber daya komputasi dalam beban kerja Anda?

Pilihan komputasi yang optimal untuk beban kerja tertentu bervariasi berdasarkan desain aplikasi, pola penggunaan, dan pengaturan konfigurasi. Arsitektur dapat menggunakan pilihan komputasi yang berbeda untuk berbagai komponen, dan memungkinkan fitur yang berbeda untuk meningkatkan kinerja. Memilih pilihan komputasi yang salah untuk arsitektur dapat menyebabkan efisiensi kinerja menjadi lebih rendah.

Praktik terbaik

- PERF02-BP01 Pilih opsi komputasi terbaik untuk beban kerja Anda
- PERF02-BP02 Memahami konfigurasi dan fitur komputasi yang tersedia
- PERF02-BP03 Kumpulkan metrik terkait komputasi
- PERF02-BP04 Mengkonfigurasi dan sumber daya komputasi ukuran kanan
- PERF02-BP05 Menskalakan sumber daya komputasi Anda secara dinamis
- PERF02-BP06 Gunakan akselerator komputasi berbasis perangkat keras yang dioptimalkan

### PERF02-BP01 Pilih opsi komputasi terbaik untuk beban kerja Anda

Dengan memilih opsi komputasi yang paling tepat untuk beban kerja, Anda dapat meningkatkan kinerja, mengurangi biaya infrastruktur yang tidak perlu, dan menurunkan upaya operasional yang diperlukan untuk memelihara beban kerja Anda.

Anti-pola umum:

- Anda menggunakan opsi komputasi yang sama yang digunakan secara on-premise.
- Anda tidak mengetahui opsi, fitur, dan solusi komputasi cloud, dan bagaimana solusi tersebut dapat meningkatkan kinerja komputasi Anda.
- Anda melakukan pengadaan opsi komputasi yang berlebihan untuk memenuhi persyaratan penskalaan atau kinerja ketika ada opsi komputasi lain yang lebih sesuai dengan karakteristik beban kerja Anda.

Manfaat menerapkan praktik terbaik ini: Dengan mengidentifikasi persyaratan komputasi dan mengevaluasi opsi-opsi yang tersedia, Anda dapat membuat beban kerja Anda lebih hemat sumber daya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk mengoptimalkan beban kerja cloud Anda demi efisiensi kinerja, penting untuk memilih opsi komputasi yang paling tepat untuk kasus penggunaan dan persyaratan kinerja Anda. AWS menyediakan berbagai opsi komputasi yang melayani beban kerja yang berbeda di cloud. Misalnya, Anda dapat menggunakan Amazon EC2 untuk meluncurkan dan mengelola server virtual, AWS Lambdamenjalankan kode tanpa harus menyediakan atau mengelola server, Amazon ECS atau Amazon EKS untuk menjalankan dan mengelola kontainer, atau AWS Batchuntuk memproses volume data yang besar secara paralel. Berdasarkan skala dan kebutuhan komputasi Anda, Anda harus memilih dan mengonfigurasi solusi komputasi yang optimal untuk situasi Anda. Anda juga dapat mempertimbangkan untuk menggunakan beberapa jenis solusi komputasi dalam satu beban kerja, karena masing-masing memiliki kelebihan dan kekurangannya sendiri.

Langkah-langkah berikut ini memandu Anda dalam memilih opsi komputasi yang tepat agar sesuai dengan karakteristik beban kerja dan persyaratan kinerja Anda.

Langkah-langkah implementasi

- Pahami persyaratan komputasi beban kerja Anda. Persyaratan utama yang harus dipertimbangkan antara lain kebutuhan pemrosesan, pola lalu lintas, pola akses data, kebutuhan penskalaan, dan persyaratan latensi.
- Pelajari tentang berbagai layanan komputasi AWS untuk beban kerja Anda. Untuk informasi selengkapnya, lihat PERF01-BP01 Pelajari dan pahami layanan dan fitur cloud yang tersedia. Berikut adalah beberapa opsi komputasi AWS utama, karakteristiknya, dan kasus penggunaan umumnya:

| AWS layanan | Karakteristik utama | Kasus penggunaan umum |
| --- | --- | --- |
| Amazon Elastic Compute Cloud (AmazonEC2) | Memiliki opsi khusus untuk perangkat keras, persyarat an lisensi, banyak pilihan rangkaian instans yang berbeda, jenis prosesor, dan akselerator komputasi | Migrasi angkat dan geser, aplikasi monolitik, lingkungan hybrid, aplikasi perusahaan |

| AWS layanan | Karakteristik utama | Kasus penggunaan umum |
| --- | --- | --- |
| Layanan Kontainer Elastis Amazon (AmazonECS), Layanan Amazon Elastic Kubernetes (Amazon) EKS | Deployment mudah, lingkungan konsisten, dapat diskalakan | Layanan mikro, lingkungan hibrida |
| AWS Lambda | Layanan komputasi nirserver yang menjalankan kode sebagai respons terhadap peristiwa dan secara otomatis mengelola sumber daya komputasi yang mendasari nya. | Layanan mikro, aplikasi yang didorong peristiwa |
| AWS Batch | Menyediakan dan menskalakan Amazon Elastic Container Service (AmazonECS) secara efisien dan dinamis, Amazon Elastic Kubernetes Service EKS (Amazon AWS Fargate), dan sumber daya komputasi, dengan opsi untuk menggunakan Instans Sesuai Permintaan atau Spot berdasarkan kebutuhan pekerjaan Anda | HPC, melatih model ML |
| Amazon Lightsail | Aplikasi Linux dan Windows yang telah dikonfigurasi sebelumnya untuk menjalankan beban kerja kecil | Aplikasi web sederhana, situs web kustom |

- Lakukan evaluasi biaya (seperti biaya per jam atau transfer data) dan overhead manajemen (seperti patching dan penskalaan) yang terkait dengan setiap opsi komputasi.
- Lakukan uji coba dan uji tolok ukur di lingkungan nonproduksi untuk mengidentifikasi opsi komputasi mana yang paling sesuai dengan kebutuhan beban kerja Anda.

- Setelah menguji coba dan mengidentifikasi solusi komputasi baru Anda, rencanakan migrasi dan validasikan metrik kinerja Anda.
- Gunakan alat AWS pemantauan seperti Amazon CloudWatch dan layanan pengoptimalan AWS Compute Optimizeruntuk terus mengoptimalkan sumber daya komputasi Anda berdasarkan pola penggunaan dunia nyata.

Sumber daya

Dokumen terkait:

- Komputasi Cloud dengan AWS
- Jenis EC2 Instans Amazon
- EKSWadah Amazon: Node EKS Pekerja Amazon
- ECSWadah Amazon: Contoh ECS Kontainer Amazon
- Fungsi: Konfigurasi Fungsi Lambda
- Panduan Preskriptif untuk Kontainer
- Panduan Preskriptif untuk Nirserver

Video terkait:

- AWS Re: invent 2023 - AWS Graviton: Kinerja harga terbaik untuk beban kerja Anda AWS
- AWS re: invent 2023 - Kemampuan AI generatif Amazon Elastic Compute Cloud baru di AMS
- AWS re:Invent 2023 - Apa yang baru dengan Amazon Elastic Compute Cloud
- AWS re:Invent 2023 - Penghematan cerdas: Strategi optimalisasi Amazon Elastic Compute Cloud
- AWS re:Invent 2021 - Mendukung Amazon Elastic Compute Cloud generasi berikutnya: Memahami lebih dalam Sistem Nitro
- AWS Re:invent 2019 - Optimalkan kinerja dan biaya untuk komputasi Anda AWS
- AWS re:Invent 2019 - Pondasi Amazon Elastic Compute Cloud
- AWS re:invent 2022 - Terapkan model ML untuk inferensi dengan kinerja tinggi dan biaya rendah
- AWS Re:invent 2019 - Optimalkan kinerja dan biaya untuk komputasi Anda AWS
- EC2Yayasan Amazon
- Melakukan deployment model ML untuk inferensi dengan performa tinggi dan biaya rendah

Contoh terkait:

- Memigrasikan Aplikasi web ke kontainer
- Menjalankan Hello World Nirserver
- EKSLokakarya Amazon
- EC2Lokakarya Amazon
- Beban Kerja yang Efisien dan Tangguh dengan Amazon Elastic Compute Cloud Auto Scaling
- Bermigrasi ke AWS Graviton dengan Layanan Kontainer

PERF02-BP02 Memahami konfigurasi dan fitur komputasi yang tersedia

Pahami opsi dan fitur konfigurasi yang tersedia bagi layanan komputasi Anda untuk membantu Anda menyediakan jumlah sumber daya yang tepat dan meningkatkan efisiensi kinerja.

Anti-pola umum:

- Anda tidak mengevaluasi opsi komputasi atau keluarga instans yang tersedia berdasarkan karakteristik beban kerja.
- Anda menyediakan sumber daya komputasi secara berlebihan untuk memenuhi persyaratan-persyaratan saat permintaan puncak.

Manfaat membangun praktik terbaik ini: Biasakan diri dengan fitur dan konfigurasi AWS komputasi sehingga Anda dapat menggunakan solusi komputasi yang dioptimalkan untuk memenuhi karakteristik dan kebutuhan beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Setiap solusi komputasi memiliki konfigurasi dan fitur unik yang tersedia untuk mendukung berbagai karakteristik dan persyaratan beban kerja. Pelajari bagaimana opsi-opsi tersebut melengkapi beban kerja Anda, dan tentukan opsi konfigurasi seperti apa yang terbaik untuk aplikasi Anda. Contoh opsi ini termasuk keluarga instance, ukuran, fitur (, I/O)GPU, bursting, time-out, ukuran fungsi, instance kontainer, dan konkurensi. Jika beban kerja Anda telah menggunakan opsi komputasi yang sama selama lebih dari empat minggu dan Anda mengantisipasi bahwa karakteristiknya akan tetap sama di masa depan, Anda dapat menggunakan AWS Compute Optimizeruntuk mengetahui apakah opsi komputasi Anda saat ini cocok untuk beban kerja dari dan perspektif memori. CPU

Langkah-langkah implementasi

- Memahami persyaratan beban kerja (seperti CPU kebutuhan, memori, dan latensi).
- Tinjau AWS dokumentasi dan praktik terbaik untuk mempelajari opsi konfigurasi yang direkomendasikan yang dapat membantu meningkatkan kinerja komputasi. Berikut adalah beberapa opsi konfigurasi utama yang perlu Anda pertimbangkan:

| Opsi Konfigurasi | Contoh |
| --- | --- |
| Jenis instans | • Instans yang dioptimalkan komputasi ideal untuk beban kerja yang membutuhkan rasio v terhadap memori yang lebih tinggi. CPU<br>• Instans memori yang dioptimalkan mengirimkan sejumlah besar memori untuk mendukung beban kerja yang intensif memori.<br>• Instans yang dioptimalkan untuk penyimpanan dirancang untuk beban kerja yang memerlukan akses baca dan tulis () IOPS yang tinggi dan berurutan ke penyimpanan lokal. |
| Model penentuan harga | • Instans Sesuai Permintaan memungkinkan Anda untuk menggunakan kapasitas komputasi per jam atau per detik tanpa perlu membuat komitmen jangka panjang. Instans ini bagus untuk lonjakan di atas kebutuhan dasar kinerja.<br>• Savings Plans menawarkan Anda penghematan yang signifikan atas Instans Sesuai Permintaan dengan komitmen untuk menggunakan daya komputasi dalam jumlah tertentu selama jangka waktu satu atau tiga tahun.<br>• Instans spot akan memungkinkan Anda untuk memanfaatkan kapasitas instans |

| Opsi Konfigurasi | Contoh |
| --- | --- |
| | yang tidak terpakai untuk beban kerja stateless dan toleran terhadap kesalahan. |
| Auto Scaling | Gunakan Auto Scaling (penskalaan otomatis) untuk mencocokkan sumber daya komputasi dengan pola lalu lintas. |
| Penyesuaian ukuran | • Gunakan Pengoptimal Komputasi untuk mendapatkan rekomendasi berbasis machine learning mengenai konfigura si komputasi yang paling cocok dengan karakteristik komputasi yang Anda miliki.<br>• Gunakan AWS Lambda Power Tuning untuk memilih konfigurasi terbaik untuk fungsi Lambda Anda. |
| Akselerator komputasi berbasis perangkat keras | • Instans komputasi yang dipercepat melakukan fungsi seperti pemrosesan grafis atau pencocokan pola data lebih efisien daripada alternatif CPU berbasis.<br>• Untuk beban kerja pembelajaran mesin, manfaatkan perangkat keras yang dibuat khusus untuk beban kerja Anda, seperti AWS Trainium, Inferentia, dan Amazon AWS EC2 DL1 |

Sumber daya

Dokumen terkait:

- Komputasi Cloud dengan AWS
- Jenis EC2 Instans Amazon
- Kontrol Status Prosesor untuk EC2 Instans Amazon Anda
- EKSWadah Amazon: Node EKS Pekerja Amazon
- ECSWadah Amazon: Contoh ECS Kontainer Amazon

- Fungsi: Konfigurasi Fungsi Lambda

Video terkait:

- AWS re: Invent 2023 - AWS Graviton: Kinerja harga terbaik untuk beban kerja Anda AWS
- AWS re: invent 2023 - Kemampuan AI generatif Amazon EC2 baru di AWS Management Console
- AWS re: Invent 2023 - Apa yang baru dengan Amazon EC2
- AWS re: Invent 2023 - Penghematan cerdas: Strategi pengoptimalan biaya Amazon EC2
- AWS Re:invent 2021 - Memberdayakan EC2 Amazon generasi berikutnya: Menyelam jauh pada Sistem Nitro
- AWS re: Ciptakan 2019 - Yayasan Amazon EC2
- AWS re:invent 2022 — Mengoptimalkan Amazon EKS untuk kinerja dan biaya AWS

Contoh terkait:

- Kode demo Pengoptimal Komputasi
- Lokakarya instans EC2 spot Amazon
- Beban Kerja yang Efisien dan Tangguh dengan Amazon EC2 AWS Auto Scaling
- Lokakarya pengembang Graviton
- AWS untuk hari perendaman beban kerja Microsoft
- AWS untuk hari perendaman beban kerja Linux
- AWS Compute Optimizer Kode demo
- EKSLokakarya Amazon

PERF02-BP03 Kumpulkan metrik terkait komputasi

Rekam dan lacak metrik-metrik terkait komputasi untuk lebih memahami kinerja sumber daya komputasi Anda dan meningkatkan kinerja serta pemanfaatannya.

Anti-pola umum:

- Anda hanya menggunakan pencarian file log manual untuk mencari metrik.
- Anda hanya menggunakan metrik-metrik default yang dicatat oleh perangkat lunak pemantauan Anda.

- Anda hanya meninjau metrik-metrik tersebut ketika terdapat masalah.

Manfaat menerapkan praktik terbaik ini: Mengumpulkan metrik terkait kinerja akan membantu Anda menyelaraskan kinerja aplikasi dengan persyaratan bisnis untuk memastikan Anda memenuhi kebutuhan beban kerja Anda. Ini juga dapat membantu Anda untuk terus meningkatkan kinerja dan pemanfaatan sumber daya dalam beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Beban kerja dapat menghasilkan data dalam jumlah besar seperti metrik, log, dan peristiwa. Dalam hal ini AWS Cloud, mengumpulkan metrik merupakan langkah penting untuk meningkatkan keamanan, efisiensi biaya, kinerja, dan keberlanjutan. AWS menyediakan berbagai metrik terkait kinerja menggunakan layanan pemantauan seperti Amazon CloudWatch untuk memberi Anda wawasan berharga. Metrik seperti CPU pemanfaatan, pemanfaatan memori, disk I/O, dan inbound dan outbound jaringan dapat memberikan wawasan tentang tingkat pemanfaatan atau kemacetan kinerja. Gunakan metrik-metrik tersebut sebagai bagian dari pendekatan berdasarkan data yang digunakan untuk mengatur dan mengoptimalkan sumber daya beban kerja Anda.  Dalam kasus yang ideal, Anda harus mengumpulkan semua metrik yang terkait dengan sumber daya komputasi Anda dalam satu platform dengan kebijakan retensi yang diterapkan untuk mendukung tujuan-tujuan biaya dan operasional.

Langkah-langkah implementasi

- Identifikasi metrik-metrik terkait kinerja apa saja yang relevan dengan beban kerja Anda. Anda harus mengumpulkan metrik seputar pemanfaatan sumber daya dan cara beban kerja cloud Anda beroperasi (seperti waktu respons dan throughput).
  - Metrik EC2 default Amazon
  - Metrik ECS default Amazon
  - Metrik EKS default Amazon
  - Metrik default Lambda
  - EC2Memori Amazon dan metrik disk
- Pilih dan siapkan solusi pembuatan log dan pemantauan yang tepat untuk beban kerja Anda.
  - Observabilitas native AWS
  - AWS Distro untuk OpenTelemetry
  - Layanan Terkelola Amazon untuk Prometheus

- Tentukan filter dan agregasi yang diperlukan untuk metrik-metrik tersebut berdasarkan persyaratan beban kerja Anda.
  - Mengukur metrik aplikasi kustom dengan Amazon CloudWatch Logs dan filter metrik
  - Kumpulkan metrik khusus dengan penandaan CloudWatch strategis Amazon
- Konfigurasikan kebijakan-kebijakan retensi data untuk metrik Anda agar sesuai dengan tujuan-tujuan keamanan dan operasional Anda.
  - Retensi data default untuk CloudWatch metrik
  - Penyimpanan data default untuk CloudWatch Log
- Jika diperlukan, buatlah alarm dan notifikasi untuk metrik Anda agar membantu Anda dalam merespons masalah terkait kinerja secara proaktif.
  - Buat alarm untuk metrik kustom menggunakan deteksi anomali Amazon CloudWatch
  - Buat metrik dan alarm untuk halaman web tertentu dengan Amazon CloudWatch RUM
- Gunakan otomatisasi untuk melakukan deployment agen agregasi log dan metrik Anda.
  - AWS Systems Manager otomatisasi
  - OpenTelemetryKolektor

Sumber daya

Dokumen terkait:

- Pemantauan dan observabilitas
- Praktik terbaik: menerapkan observabilitas dengan AWS
- CloudWatch Dokumentasi Amazon
- Kumpulkan metrik dan log dari EC2 instans Amazon dan server lokal dengan Agen CloudWatch
- Mengakses CloudWatch Log Amazon untuk AWS Lambda
- Menggunakan CloudWatch Log dengan instance kontainer
- Menerbitkan metrik kustom
- Jawaban AWS : Pencatatan Log Terpusat
- AWS Layanan yang Mempublikasikan CloudWatch Metrik
- Memantau Amazon EKS di AWS Fargate

Video terkait:

- AWS RE: invent 2023 - [LAUNCH] Pemantauan aplikasi untuk beban kerja modern
- AWS re:invent 2023 - Menerapkan observabilitas aplikasi
- AWS re:invent 2023 — Membangun strategi observabilitas yang efektif
- AWS Re:invent 2023 - Observabilitas mulus dengan Distro untuk AWS OpenTelemetry
- Manajemen Kinerja Aplikasi pada AWS

Contoh terkait:

- AWS untuk Hari Perendaman Beban Kerja Linux- Amazon CloudWatch
- Memantau ECS cluster dan kontainer Amazon
- Pemantauan dengan CloudWatch dasbor Amazon
- EKSLokakarya Amazon

PERF02-BP04 Mengkonfigurasi dan sumber daya komputasi ukuran kanan

Konfigurasikan dan tentukan ukuran yang tepat untuk sumber daya agar sesuai dengan persyaratan-persyaratan kinerja beban kerja Anda dan hindari sumber daya dengan pemanfaatan yang terlalu rendah atau terlalu tinggi.

Anti-pola umum:

- Anda mengabaikan persyaratan kinerja beban kerja yang mengakibatkan sumber daya komputasi mengalami pemanfaatan yang terlalu rendah atau terlalu tinggi.
- Anda hanya memilih instans terbesar atau terkecil untuk semua beban kerja.
- Anda hanya menggunakan satu keluarga instans untuk kemudahan manajemen.
- Anda mengabaikan rekomendasi dari AWS Cost Explorer atau Compute Optimizer untuk ukuran yang tepat.
- Anda tidak mengevaluasi ulang beban kerja untuk memeriksa kesesuaian tipe instans baru.
- Anda hanya mengesahkan sejumlah kecil konfigurasi instans untuk organisasi Anda.

Manfaat menerapkan praktik terbaik ini: Penyesuaian ukuran yang tepat untuk sumber daya komputasi akan memastikan pengoperasian yang optimal di cloud dengan menghindari penyediaan sumber daya yang terlalu banyak dan terlalu sedikit. Penyesuaian ukuran sumber daya komputasi secara tepat biasanya menghasilkan kinerja yang lebih baik dan pengalaman pelanggan yang ditingkatkan, sekaligus juga dapat menurunkan biaya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Penentuan ukuran yang tepat memungkinkan organisasi untuk mengoperasikan infrastruktur cloud mereka dengan cara yang efisien dan hemat biaya sambil menangani kebutuhan-kebutuhan bisnis mereka. Penyediaan sumber daya cloud yang berlebihan dapat menyebabkan biaya tambahan, sementara penyediaan yang kurang dapat mengakibatkan kinerja yang buruk dan pengalaman pelanggan yang negatif. AWS menyediakan alat seperti AWS Compute Optimizerdan AWS Trusted Advisoryang menggunakan data historis untuk memberikan rekomendasi untuk mengukur sumber daya komputasi Anda dengan benar.

Langkah-langkah implementasi

- Pilih tipe instans yang paling sesuai dengan kebutuhan Anda:
  - Bagaimana cara memilih jenis EC2 instans Amazon yang sesuai untuk beban kerja saya?
  - Pemilihan jenis instans berbasis atribut untuk Amazon Fleet EC2
  - Membuat grup Auto Scaling dengan menggunakan pemilihan jenis instans berdasarkan atribut
  - Mengoptimalkan biaya komputasi Kubernetes Anda dengan konsolidasi Karpenter
- Analisis berbagai karakteristik kinerja beban kerja Anda dan bagaimana karakteristik ini berhubungan dengan memori, jaringan, dan CPU penggunaan. Gunakan data ini untuk memilih sumber daya yang paling sesuai dengan profil beban kerja dan tujuan-tujuan kinerja Anda.
- Pantau penggunaan sumber daya Anda menggunakan alat AWS pemantauan seperti Amazon CloudWatch.
- Pilih konfigurasi yang tepat untuk sumber daya komputasi.
  - Untuk beban kerja sementara, evaluasi CloudWatch metrik Amazon instance seperti `CPUUtilization` untuk mengidentifikasi apakah instans kurang dimanfaatkan atau terlalu banyak digunakan.
  - Untuk beban kerja yang stabil, periksa alat AWS penentuan ukuran seperti AWS Compute Optimizer dan secara berkala untuk mengidentifikasi peluang untuk mengoptimalkan dan AWS Trusted Advisor mengukur sumber daya komputasi dengan benar.
- Uji perubahan konfigurasi di lingkungan non-produksi sebelum diimplementasikan di lingkungan langsung.
- Lakukan evaluasi ulang secara terus-menerus terhadap penawaran komputasi baru dan bandingkan dengan kebutuhan-kebutuhan beban kerja Anda.

Sumber daya

Dokumen terkait:

- Cloud Compute dengan AWS
- Jenis EC2 Instans Amazon
- ECSWadah Amazon: Contoh ECS Kontainer Amazon
- EKSWadah Amazon: Node EKS Pekerja Amazon
- Fungsi: Konfigurasi Fungsi Lambda
- Kontrol Status Prosesor untuk EC2 Instans Amazon Anda

Video terkait:

- EC2Yayasan Amazon
- AWS re: Invent 2023 - AWS Graviton: Performa harga terbaik untuk beban kerja Anda AWS
- AWS re: invent 2023 - Kemampuan AI generatif Amazon EC2 baru di AWS Management Console
- AWS re: Invent 2023 - Apa yang baru dengan Amazon EC2
- AWS re: Invent 2023 - Penghematan cerdas: Strategi pengoptimalan biaya Amazon EC2
- AWS Re:invent 2021 - Memberdayakan EC2 Amazon generasi berikutnya: Menyelam jauh pada Sistem Nitro
- AWS re: Ciptakan 2019 - Yayasan Amazon EC2

Contoh terkait:

- AWS Compute Optimizer Kode demo
- EKSLokakarya Amazon
- Rekomendasi penyesuaian ukuran

PERF02-BP05 Menskalakan sumber daya komputasi Anda secara dinamis

Gunakan elastisitas cloud untuk menaikkan atau menurunkan skala sumber daya komputasi Anda secara dinamis agar sesuai dengan kebutuhan Anda dan hindari kapasitas penyediaan yang terlalu tinggi atau terlalu rendah untuk beban kerja Anda.

Anti-pola umum:

- Anda bereaksi pada alarm-alarm dengan meningkatkan kapasitas secara manual.
- Anda menggunakan pedoman penyesuaian ukuran yang sama (umumnya infrastruktur statis) seperti yang digunakan di on-premise.
- Anda membiarkan peningkatan kapasitas setelah terjadi peristiwa penskalaan, bukannya menurunkan kembali skala.

Manfaat menerapkan praktik terbaik ini: Mengonfigurasi dan menguji elastisitas sumber daya komputasi dapat membantu Anda menghemat dana, mempertahankan tolok ukur kinerja, dan meningkatkan keandalan saat lalu lintas berubah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS memberikan fleksibilitas untuk meningkatkan atau menurunkan sumber daya Anda secara dinamis melalui berbagai mekanisme penskalaan untuk memenuhi perubahan permintaan. Digabungkan dengan metrik-metrik yang terkait dengan komputasi, penskalaan dinamis akan memungkinkan beban kerja untuk merespons perubahan secara otomatis dan menggunakan rangkaian optimal sumber daya komputasi untuk mencapai tujuannya.

Anda dapat menggunakan sejumlah pendekatan yang berbeda untuk menyesuaikan pasokan sumber daya dengan permintaan.

- Pendekatan pelacakan sasaran: Pantau metrik penskalaan Anda dan tingkatkan atau turunkan kapasitas secara otomatis sesuai kebutuhan.
- Penskalaan prediktif: Lakukan pengurangan skala (scale in) dalam mengantisipasi tren harian dan mingguan.
- Pendekatan berbasis jadwal: Tetapkan jadwal penskalaan Anda sendiri sesuai dengan perubahan-perubahan beban yang dapat diprediksi.
- Penskalaan layanan: Pilihlah layanan-layanan (seperti nirserver) yang secara otomatis menskalakan sesuai rancangan.

Anda harus memastikan bahwa deployment beban kerja tersebut dapat menangani peristiwa kenaikan (scale-up) dan penurunan skala (scale-down).

Langkah-langkah implementasi

- Instans, kontainer, dan fungsi komputasi menyediakan mekanisme bagi elastisitas, baik dengan dikombinasikan bersama penskalaan otomatis atau sebagai sebuah fitur layanan. Berikut beberapa contoh mekanisme penskalaan otomatis:

| Mekanisme Penskalaan Otomatis | Harus digunakan di mana |
| --- | --- |
| EC2Auto Scaling Amazon | Untuk memastikan Anda memiliki jumlah EC2 instans Amazon yang benar yang tersedia untuk menangani pemuatan pengguna untuk aplikasi Anda. |
| Penskalaan Otomatis Aplikasi | Untuk secara otomatis menskalakan sumber daya untuk AWS layanan individual di luar Amazon EC2 seperti AWS Lambdafungsi atau layanan Amazon Elastic Container Service (AmazonECS). |
| Penskala Otomatis Klaster Kubernetes/ Karpenter | Untuk secara otomatis menskalakan klaster Kubernetes. |

- Penskalaan sering dibahas terkait dengan layanan komputasi seperti EC2 Instans atau fungsi Amazon. AWS Lambda Pastikan juga untuk mempertimbangkan konfigurasi layanan non-komputasi seperti AWS Glue untuk mengimbangi permintaan.
- Pastikan bahwa metrik-metrik untuk penskalaan cocok dengan karakteristik beban kerja yang sedang di-deploy. Jika Anda menerapkan aplikasi transcoding video, CPU pemanfaatan 100% diharapkan dan seharusnya tidak menjadi metrik utama Anda. Gunakan kedalaman antrean tugas transkode sebagai gantinya. Anda dapat menggunakan metrik yang dikustom untuk kebijakan penskalaan Anda, jika diperlukan. Untuk memilih metrik yang tepat, pertimbangkan panduan berikut untuk AmazonEC2:
  - Metrik tersebut harus merupakan metrik pemanfaatan yang valid dan mendeskripsikan tingkat kesibukan suatu instans.
  - Nilai metrik harus meningkatkan atau menurunkan secara proporsional jumlah instance dalam grup Auto Scaling.

- Pastikan Anda menggunakan penskalaan dinamis alih-alih penskalaan manual untuk grup Auto Scaling Anda. Kami juga menyarankan agar Anda menggunakan kebijakan penskalaan pelacakan target dalam penskalaan dinamis Anda.
- Pastikan deployment beban kerja dapat menangani dua jenis peristiwa penskalaan (naik dan turun). Sebagai contoh, Anda dapat menggunakan Riwayat aktivitas untuk melakukan verifikasi terhadap aktivitas penskalaan untuk sebuah grup Auto Scaling.
- Lakukan evaluasi terhadap beban kerja Anda untuk memeriksa pola-pola terprediksi dan secara proaktif skalakan saat Anda mengantisipasi perubahan permintaan yang terencana dan terprediksi. Dengan penskalaan prediktif, Anda dapat menghilangkan kebutuhan untuk menyediakan kapasitas secara berlebih. Untuk detail selengkapnya, lihat Penskalaan Prediktif dengan Auto EC2 Scaling Amazon.

Sumber daya

Dokumen terkait:

- Cloud Compute dengan AWS
- Jenis EC2 Instans Amazon
- ECSWadah Amazon: Contoh ECS Kontainer Amazon
- EKSWadah Amazon: Node EKS Pekerja Amazon
- Fungsi: Konfigurasi Fungsi Lambda
- Kontrol Status Prosesor untuk EC2 Instans Amazon Anda
- Menyelam Jauh di Auto Scaling Amazon ECS Cluster
- Memperkenalkan Karpenter – Penskala Otomatis Klaster Kubernetes Sumber Terbuka dan Performa Tinggi

Video terkait:

- AWS re: Invent 2023 - AWS Graviton: Kinerja harga terbaik untuk beban kerja Anda AWS
- AWS re: invent 2023 - Kemampuan AI EC2 generatif Amazon baru di Konsol Manajemen AWS
- AWS re: Invent 2023 - Apa yang baru dengan Amazon EC2
- AWS re: Invent 2023 - Penghematan cerdas: Strategi pengoptimalan biaya Amazon EC2
- AWS Re:invent 2021 - Memberdayakan EC2 Amazon generasi berikutnya: Menyelam jauh pada Sistem Nitro

- AWS re: Ciptakan 2019 - Yayasan Amazon EC2

Contoh terkait:

- Contoh Grup EC2 Auto Scaling Amazon
- EKSLokakarya Amazon
- Skalakan EKS beban kerja Amazon Anda dengan menjalankannya IPv6

PERF02-BP06 Gunakan akselerator komputasi berbasis perangkat keras yang dioptimalkan

Gunakan akselerator perangkat keras untuk melakukan fungsi tertentu secara lebih efisien daripada alternatif CPU berbasis.

Anti-pola umum:

- Dalam beban kerja Anda, Anda belum melakukan uji tolok ukur instans tujuan umum dengan instans yang dibuat khusus yang dapat memberikan kinerja lebih tinggi dan biaya yang lebih rendah.
- Anda menggunakan akselerator komputasi berbasis perangkat keras untuk tugas-tugas yang dapat lebih efisien menggunakan alternatif berbasis. CPU
- Anda tidak memantau GPU penggunaan.

Manfaat membangun praktik terbaik ini: Dengan menggunakan akselerator berbasis perangkat keras, seperti unit pemrosesan grafis (GPUs) dan array gerbang yang dapat diprogram lapangan (FPGAs), Anda dapat melakukan fungsi pemrosesan tertentu dengan lebih efisien.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Instans komputasi yang dipercepat menyediakan akses ke akselerator komputasi berbasis perangkat keras seperti dan. GPUs FPGAs Akselerator perangkat keras ini melakukan fungsi tertentu seperti pemrosesan grafis atau pencocokan pola data lebih efisien daripada alternatif CPU berbasis. Banyak beban kerja terakselerasi, seperti perenderan, transkode, dan machine learning, yang memiliki variabel tinggi sehubungan dengan penggunaan sumber daya. Jalankan perangkat keras ini hanya ketika diperlukan, dan non-aktifkan instans GPU secara otomatis saat tidak diperlukan untuk meningkatkan efisiensi kinerja secara keseluruhan.

Langkah-langkah implementasi

- Identifikasi instans komputasi terakselerasi yang dapat memenuhi kebutuhan Anda.
- Untuk beban kerja pembelajaran mesin, manfaatkan perangkat keras yang dibuat khusus untuk beban kerja Anda, seperti AWS Trainium, Inferentia, dan Amazon.AWS EC2 DL1 AWS Instans Inferentia seperti instans Inf2 menawarkan kinerja/watt hingga 50% lebih baik dibandingkan instans Amazon yang sebanding. EC2
- Kumpulkan metrik-metrik penggunaan untuk instans komputasi terakselerasi Anda. Misalnya, Anda dapat menggunakan CloudWatch agen untuk mengumpulkan metrik seperti `utilization_gpu` dan `utilization_memory` untuk Anda GPUs seperti yang ditunjukkan dalam Kumpulkan NVIDIA GPU metrik dengan Amazon. CloudWatch
- Optimalkan kode, operasi jaringan, dan pengaturan akselerator perangkat keras untuk memastikan perangkat keras yang mendasarinya dimanfaatkan sepenuhnya.
  - Optimalkan GPU pengaturan
  - GPUPemantauan dan Optimalisasi dalam Pembelajaran Mendalam AMI
  - Mengoptimalkan I/O untuk penyetelan GPU kinerja pelatihan pembelajaran mendalam di Amazon SageMaker
- Gunakan pustaka dan driver berkinerja tinggi terbaru. GPU
- Gunakan otomatisasi untuk merilis GPU instance saat tidak digunakan.

Sumber daya

Dokumen terkait:

- Bekerja dengan GPUs Amazon Elastic Container Service
- GPUcontoh
- Contoh dengan Trainium AWS
- Contoh dengan AWS Inferensia
- Mari Merancang! Merancang arsitektur dengan chip dan akselerator kustom

- Komputasi yang Dipercepat
- EC2VT1Contoh Amazon
- Bagaimana cara memilih jenis EC2 instans Amazon yang sesuai untuk beban kerja saya?

- Pilih akselerator AI terbaik dan kompilasi model untuk inferensi visi komputer dengan Amazon SageMaker

Video terkait:

- AWS re:invent 2021 - Cara memilih instans Amazon Elastic Compute Cloud GPU untuk pembelajaran mendalam
- AWS Re: ciptakan 2022 - [!] NEW LAUNCH Memperkenalkan instans AWS Amazon Inf2 berbasis Inferensia2 EC2
- AWS re:Invent 2022 - Mempercepat deep learning dan berinovasi lebih cepat dengan AWS Trainium
- AWS re:invent 2022 - Pembelajaran mendalam AWS denganNVIDIA: Dari pelatihan hingga penerapan

Contoh terkait:

- Amazon SageMaker dan NVIDIA GPU Cloud (NGC)
- Gunakan SageMaker dengan Trainium dan Inferentia untuk pelatihan pembelajaran mendalam yang dioptimalkan dan menyimpulkan beban kerja
- Mengoptimalkan NLP model dengan instans Amazon Elastic Compute Cloud Inf1 di Amazon SageMaker

## Manajemen data

Pertanyaan

- PERF3. Bagaimana cara menyimpan, mengelola, dan mengakses data dalam beban kerja Anda?

### PERF3. Bagaimana cara menyimpan, mengelola, dan mengakses data dalam beban kerja Anda?

Solusi manajemen data yang optimal untuk sistem tertentu bervariasi berdasarkan jenis tipe data (blok, file, atau objek), pola akses (acak atau sekuensial), throughput yang diperlukan, frekuensi akses (online, offline, arsip), frekuensi pembaruan (, dinamis)WORM, dan kendala ketersediaan dan daya tahan. Beban kerja Well-Architected menggunakan penyimpanan data yang dibuat khusus yang memungkinkan berbagai fitur untuk meningkatkan kinerja.

Praktik terbaik

- PERF03-BP01 Gunakan penyimpanan data yang dibuat khusus yang paling mendukung persyaratan akses dan penyimpanan data Anda
- PERF03-BP02 Mengevaluasi opsi konfigurasi yang tersedia untuk penyimpanan data
- PERF03-BP03 Kumpulkan dan rekam metrik kinerja penyimpanan data
- PERF03-BP04 Menerapkan strategi untuk meningkatkan kinerja kueri di penyimpanan data
- PERF03-BP05 Menerapkan pola akses data yang memanfaatkan caching

PERF03-BP01 Gunakan penyimpanan data yang dibuat khusus yang paling mendukung persyaratan akses dan penyimpanan data Anda

Pahami karakteristik data (seperti dapat dibagikan, ukuran, ukuran cache, pola akses, latensi, throughput, dan persistensi data) untuk memilih penyimpanan data yang dibuat khusus (penyimpanan atau basis data) yang tepat untuk beban kerja Anda.

Anti-pola umum:

- Anda bertahan dengan satu solusi basis data disebabkan karena Anda hanya memiliki pengetahuan dan pengalaman internal tentang satu jenis solusi basis data tertentu.
- Anda berasumsi bahwa semua beban kerja memiliki persyaratan penyimpanan data dan akses data yang serupa.
- Anda belum mengimplementasikan katalog data untuk menginventarisasi aset data Anda.

Manfaat menerapkan praktik terbaik ini: Dengan memahami karakteristik dan persyaratan data, Anda dapat menentukan teknologi penyimpanan yang paling efisien dan berkinerja paling tinggi sesuai dengan kebutuhan beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Saat memilih dan menerapkan penyimpanan data, pastikan bahwa karakteristik kueri, penskalaan, dan penyimpanan mendukung persyaratan data beban kerja. AWS menyediakan banyak penyimpanan data dan teknologi database termasuk penyimpanan blok, penyimpanan objek, penyimpanan streaming, sistem file, relasional, nilai kunci, dokumen, dalam memori, grafik, deret waktu, dan database buku besar. Setiap solusi manajemen data memiliki opsi dan konfigurasi yang bisa Anda gunakan untuk mendukung kasus penggunaan dan model data Anda. Dengan memahami

karakteristik dan persyaratan data, Anda dapat melepaskan diri dari teknologi penyimpanan monolitik dan one-size-fits-all pendekatan yang membatasi untuk fokus pada pengelolaan data dengan tepat.

Langkah-langkah implementasi

- Lakukan inventarisasi berbagai jenis data yang ada dalam beban kerja Anda.
- Pahami dan dokumentasikan karakteristik serta persyaratan data, termasuk:
  - Tipe data (tidak terstruktur, semi-terstruktur, relasional)
  - Volume dan pertumbuhan data
  - Ketahanan data: persisten, sementara, transien
  - ACIDPersyaratan (atomisitas, konsistensi, isolasi, daya tahan)
  - Pola akses data (intensif baca atau intensif tulis)
  - Latensi
  - Throughput
  - IOPS(operasi input/output per detik)
  - Periode retensi data
- Pelajari tentang berbagai penyimpanan data (penyimpanan dan layanan basis data) yang tersedia untuk beban kerja Anda AWS yang dapat memenuhi karakteristik data Anda, seperti yang diuraikan dalam. PERF01-BP01 Pelajari dan pahami layanan dan fitur cloud yang tersedia Beberapa contoh teknologi penyimpanan AWS serta karakteristik utamanya antara lain:

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Penyimpanan objek | Amazon S3 | Skalabilitas tak terbatas, ketersediaan tinggi, dan berbagai opsi aksesibilitas. Mentransfer dan mengakses objek masuk dan keluar dari Amazon S3 dapat dilakukan dengan menggunakan layanan-layanan, seperti Akselerasi Transfer atau Titik Akses, untuk mendukung lokasi, kebutuhan keamanan, dan pola akses Anda. |

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Penyimpanan pengarsipan | Amazon S3 Glacier | Dirancang untuk pengarsipan data. |
| Penyimpanan streaming | Amazon Kinesis<br><br>Amazon Managed Streaming for Apache Kafka (Amazon) MSK | Penyerapan dan penyimpan an data streaming yang efisien. |
| Sistem file bersama | Amazon Elastic File System (AmazonEFS) | Sistem file yang dapat dipasang dan dapat diakses oleh berbagai jenis solusi komputasi. |
| Sistem file bersama | Amazon FSx | Dibangun di atas solusi AWS komputasi terbaru untuk mendukung empat sistem file yang umum digunakan: NetApp ONTAP, OpenZFS, Windows File Server, dan Lustre. FSxLatensi Amazon, throughput, dan IOPS bervariasi per sistem file dan harus dipertimbangkan saat memilih sistem file yang tepat untuk kebutuhan beban kerja Anda. |

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Penyimpanan blok | Toko Blok Elastis Amazon (AmazonEBS) | Layanan penyimpanan blok berkinerja tinggi yang dapat diskalakan yang dirancang untuk Amazon Elastic Compute Cloud (Amazon). EC2 Amazon EBS menyertakan penyimpanan yang SSD didukung untuk beban kerja transaksional dan intensif serta HDD penyimpanan yang IOPS didukung untuk beban kerja yang intensif throughput. |
| Basis data relasional | Amazon Aurora, AmazonRDS, Amazon Redshift. | Dirancang untuk mendukung transaksi ACID (atomisitas, konsistensi, isolasi, daya tahan), dan menjaga integritas referensial dan konsistensi data yang kuat. Banyak aplikasi tradisional, perencanaan sumber daya perusahaan (ERP), manajemen hubungan pelanggan (CRM), dan e-commerce menggunakan database relasional untuk menyimpan data mereka. |

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Basis data nilai-kunci | Amazon DynamoDB | Dioptimalkan untuk pola akses umum, biasanya digunakan untuk menyimpan dan mengambil data dalam volume besar. Aplikasi web dengan lalu lintas tinggi, sistem perdagangan elektroni k, dan aplikasi gaming merupakan kasus penggunaa n umum untuk basis data nilai kunci. |
| Basis data dokumen | Amazon DocumentDB | Dirancang untuk menyimpan data semi-terstruktur sebagai dokumen JSON -like. Basis data ini membantu para pengembang untuk dengan cepat membangun dan memperbarui aplikasi seperti manajemen konten, katalog, dan profil pengguna. |

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Basis data dalam memori | Amazon ElastiCache, Amazon MemoryDB untuk Redis | Digunakan untuk aplikasi-aplikasi yang memerlukan akses waktu nyata ke data, latensi rendah, dan throughput paling tinggi. Anda dapat menggunakan basis data dalam memori untuk melakukan caching aplikasi, manajemen sesi, papan peringkat game, penyimpanan fitur ML latensi rendah, sistem olah pesan layanan mikro, dan mekanisme streaming throughput tinggi |
| Basis data grafik | Amazon Neptune | Digunakan untuk aplikasi-aplikasi yang harus menavigasi dan melakukan kueri jutaan hubungan antara set data grafik yang sangat terhubung dengan latensi milidetik dalam skala besar. Banyak perusahaan menggunakan basis data grafik untuk mesin rekomendasi, jaringan sosial, dan deteksi penipuan. |

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Basis Data Deret Waktu | Amazon Timestream | Digunakan untuk mengumpulkan, mempersatukan, dan mengambil wawasan secara efisien dari data yang berubah seiring waktu. Aplikasi IoT, DevOps, dan telemetri industri dapat memanfaatkan database deret waktu. |
| Kolom lebar | Amazon Keyspaces (untuk Apache Cassandra) | Menggunakan tabel, baris, dan kolom, tetapi tidak seperti basis data relasional, nama dan format kolomnya berbeda-beda dari baris ke baris di tabel yang sama. Biasanya Anda akan melihat penyimpanan kolom lebar di aplikasi industri skala tinggi untuk melakukan pemeliharaan perlengkapan, pengelolaan armada, dan pengoptimalan rute. |

| Jenis | AWS Jasa | Karakteristik utama |
| --- | --- | --- |
| Buku besar | Database Buku Besar Amazon Quantum (AmazonQLDB) | Memberikan otoritas tersentralisasi yang tepercaya untuk mempertahankan data transaksi yang dapat diskalakan, tetap, dan dapat diverifikasi secara kriptografis untuk setiap aplikasi. Basis data buku besar digunakan untuk sistem catatan, rantai pasokan, registrasi, dan bahkan transaksi perbankan. |

- Jika Anda membangun platform data, manfaatkan arsitektur data modern AWS untuk mengintegrasikan data lake, gudang data, dan penyimpanan data yang dibuat khusus.
- Pertanyaan kunci yang perlu Anda pertimbangkan saat memilih penyimpanan data untuk beban kerja Anda adalah sebagai berikut:

| Pertanyaan | Hal-hal yang perlu dipertimbangkan |
| --- | --- |
| Bagaimana data terstruktur? | • Jika data tidak terstruktur, pertimbangkan penyimpanan objek seperti Amazon S3 atau database Tidak ada SQL seperti Amazon DocumentDB<br>• Untuk data nilai kunci, pertimbangkan DynamoDB, Amazon ( ElastiCache Redis) atau Amazon MemoryDB OSS |
| Apa tingkat integritas referensial yang dibutuhkan? | • Untuk kendala kunci asing, database relasional seperti Amazon RDS dan Aurora dapat memberikan tingkat integritas ini.<br>• Biasanya, dalam SQL model No data, Anda akan de-normalisasi data Anda menjadi satu dokumen atau kumpulan dokumen |

| Pertanyaan | Hal-hal yang perlu dipertimbangkan |
| --- | --- |
| | yang akan diambil dalam satu permintaa n daripada bergabung di seluruh dokumen atau tabel. |
| Apakah kepatuhan ACID (atomisitas, konsistensi, isolasi, daya tahan) diperlukan? | • Jika ACID properti yang terkait dengan database relasional diperlukan, pertimban gkan database relasional seperti Amazon dan RDS Aurora.<br>• Jika konsistensi yang kuat diperlukan untuk Tidak ada SQL database, Anda dapat menggunakan pembacaan yang sangat konsisten dengan DynamoDB. |
| Bagaimana persyaratan penyimpanan akan berubah seiring waktu? Bagaimana dampaknya pada skalabilitas? | • Database tanpa server seperti DynamoDB dan Amazon Quantum Ledger Database (Amazon) akan diskalakan secara dinamis. QLDB<br>• Basis data relasional memiliki batas atas terkait penyimpanan yang tersedia, dan sering kali harus dipartisi secara horizonta l dengan menggunakan mekanisme seperti serpihan (sharding) setelah penyimpanan tersebut mencapai batas ini. |
| Berapakah proporsi kueri baca dibanding kan dengan kueri tulis? Apakah caching akan meningkatkan performa? | • Beban kerja read-heavy dapat mengambil manfaat dari lapisan caching, seperti ElastiCacheatau DAXjika database DynamoDB.<br>• Bacaan juga dapat diturunkan untuk membaca replika dengan database relasional seperti Amazon. RDS |

| Pertanyaan | Hal-hal yang perlu dipertimbangkan |
| --- | --- |
| Apakah penyimpanan dan modifikasi (OLTP- Pemrosesan Transaksi Online) atau pengambilan dan pelaporan (OLAP- Pemrosesan Analitik Online) memiliki prioritas yang lebih tinggi? | • Untuk pemrosesan transaksional read as-is throughput tinggi, pertimbangkan database SQL No seperti DynamoDB.<br>• Untuk throughput tinggi dan pola baca yang kompleks (seperti bergabung) dengan konsistensi, gunakan Amazon. RDS<br>• Untuk kueri analitis, pertimbangkan database kolumnar seperti Amazon Redshift atau mengekspor data ke Amazon S3 dan melakukan analisis menggunakan Athena atau Amazon. QuickSight |
| Tingkat durabilitas apa yang diperlukan data? | • Aurora secara otomatis mereplikasi data Anda di tiga Zona Ketersediaan dalam satu Wilayah, yang artinya data Anda sangat tahan lama dengan lebih sedikit kemungkin an hilangnya data.<br>• DynamoDB secara otomatis direplikasi di beberapa Zona Ketersediaan, memberikan durabilitas data dan ketersediaan tinggi.<br>• Amazon S3 memberikan 11 sembilan durabilitas. Banyak layanan database, seperti Amazon RDS dan DynamoDB, mendukung ekspor data ke Amazon S3 untuk retensi dan arsip jangka panjang. |
| Apakah ada keinginan untuk beralih dari mesin basis data komersial atau biaya lisensi? | • Pertimbangkan mesin open-source seperti Postgre SQL dan My di SQL Amazon atau RDS Aurora.<br>• Manfaatkan AWS Database Migration Service dan AWS Schema Conversion Tool untuk melakukan migrasi dari mesin basis data komersial ke mesin sumber terbuka |

| Pertanyaan | Hal-hal yang perlu dipertimbangkan |
| --- | --- |
| Apa harapan operasional untuk basis data tersebut? Apakah beralih ke layanan terkelola menjadi perhatian utama? | • Memanfaatkan Amazon RDS alih-alih AmazonEC2, dan DynamoDB atau Amazon DocumentDB alih-alih menghosting sendiri database Tidak dapat mengurangi overhead operasional. SQL |
| Bagaimana basis data diakses saat ini? Apakah hanya akses aplikasi, atau apakah ada pengguna intelijen bisnis (BI) dan off-the-s helf aplikasi terhubung lainnya? | • Jika Anda memiliki dependensi pada peralatan eksternal maka Anda mungkin harus menjaga kompatibilitas dengan basis data yang mendukungnya. Amazon sepenuhnya RDS kompatibel dengan versi mesin perbedaan yang didukungnya termasuk Microsoft SQL Server, Oracle, MySQL, dan SQL Postgre. |

- Lakukan uji coba dan uji tolok ukur di lingkungan non-produksi untuk mengidentifikasi penyimpanan data mana yang paling sesuai dengan kebutuhan beban kerja Anda.

Sumber daya

Dokumen terkait:

- Jenis EBS Volume Amazon
- EC2Penyimpanan Amazon
- AmazonEFS: EFS Kinerja Amazon
- Amazon FSx untuk Kinerja Kilau
- Amazon FSx untuk Kinerja Server File Windows
- Amazon S3 Glacier: Dokumentasi Glacier S3
- Amazon S3: Pertimbangan Tingkat Permintaan dan Performa
- Cloud Storage dengan AWS
- Karakteristik Amazon EBS I/O
- Basis Data Cloud dengan AWS
- AWS Caching Basis Data

- DynamoDB Accelerator
- Video praktik terbaik Amazon Aurora
- Kinerja Amazon Redshift
- 10 kiat kinerja teratas Amazon Athena
- Praktik terbaik Amazon Redshift Spectrum
- Praktik terbaik Amazon DynamoDB
- Pilih antara Amazon EC2 dan Amazon RDS
- Praktik Terbaik untuk Menerapkan Amazon ElastiCache

Video terkait:

- AWS RE: invent 2023: Meningkatkan efisiensi Amazon Elastic Block Store dan menjadi lebih hemat biaya
- AWS RE: invent 2023: Mengoptimalkan harga dan kinerja penyimpanan dengan Amazon Simple Storage Service
- AWS re:invent 2023: Membangun dan mengoptimalkan data lake di Amazon Simple Storage Service
- AWS re:invent 2022: Membangun arsitektur data modern AWS
- AWS re:invent 2022: Membangun arsitektur data mesh AWS
- AWS re: invent 2023: Menyelam jauh ke Amazon Aurora dan inovasinya
- AWS RE: invent 2023: Pemodelan data tingkat lanjut dengan Amazon DynamoDB
- AWS re:invent 2022: Modernisasi aplikasi dengan database yang dibuat khusus
- Memahami lebih dalam Amazon DynamoDB: Pola desain tingkat lanjut

Contoh terkait:

- AWS Workshop Database yang Dibangun Tujuan
- Basis Data untuk Pengembang
- AWS Hari Perendaman Arsitektur Data Modern
- Membangun Data Mesh di AWS
- Contoh-contoh Amazon S3

- Optimalkan Pola Data Menggunakan Pembagian Data Amazon Redshift
- Migrasi Basis Data
- MS SQL Server - AWS Database Migration Service (AWS DMS) Demo Replikasi
- Lokakarya Praktik Langsung Modernisasi Basis Data
- Sampel Amazon Neptune

PERF03-BP02 Mengevaluasi opsi konfigurasi yang tersedia untuk penyimpanan data

Pahami dan evaluasi berbagai fitur dan opsi konfigurasi yang tersedia untuk penyimpanan data Anda guna mengoptimalkan ruang penyimpanan dan kinerja untuk beban kerja Anda.

Anti-pola umum:

- Anda hanya menggunakan satu jenis penyimpanan, seperti AmazonEBS, untuk semua beban kerja.
- Anda menggunakan provisioned IOPS untuk semua beban kerja tanpa pengujian dunia nyata terhadap semua tingkatan penyimpanan.
- Anda tidak memahami opsi-opsi konfigurasi dari solusi manajemen data yang Anda pilih.
- Anda hanya mengandalkan peningkatan ukuran instans tanpa mempertimbangkan opsi-opsi konfigurasi lain yang tersedia.
- Anda tidak melakukan pengujian terhadap karakteristik penskalaan penyimpanan data Anda.

Manfaat menerapkan praktik terbaik ini: Dengan menjelajahi dan melakukan eksperimen dengan konfigurasi penyimpanan data, Anda mungkin dapat mengurangi biaya infrastruktur, meningkatkan performa, serta mengurangi upaya pengelolaan beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Beban kerja dapat memiliki satu atau beberapa penyimpanan data yang digunakan berdasarkan persyaratan-persyaratan penyimpanan data dan akses data. Untuk mengoptimalkan biaya dan efisiensi kinerja, Anda harus melakukan evaluasi terhadap pola akses data untuk menentukan konfigurasi penyimpanan data yang sesuai. Saat mencoba berbagai opsi penyimpanan data tersebut, Anda harus mempertimbangkan beberapa aspek seperti opsi penyimpanan, memori, komputasi,

replika baca, persyaratan konsistensi, pooling koneksi, dan opsi cache. Cobalah berbagai opsi konfigurasi ini untuk meningkatkan metrik efisiensi kinerja.

Langkah-langkah implementasi

- Pahami konfigurasi (seperti tipe instans, ukuran penyimpanan, atau versi mesin basis data) penyimpanan data Anda saat ini.
- Tinjau AWS dokumentasi dan praktik terbaik untuk mempelajari opsi konfigurasi yang direkomendasikan yang dapat membantu meningkatkan kinerja penyimpanan data Anda. Berikut ini adalah opsi-opsi penyimpanan data utama yang perlu dipertimbangkan:

| Opsi Konfigurasi | Contoh |
| --- | --- |
| Melimpahkan beban baca (seperti replika baca dan caching) | • Untuk tabel DynamoDB, Anda dapat membongkar pembacaan menggunakan untuk caching. DAX<br>• Anda dapat membuat klaster Amazon ElastiCache (RedisOSS) dan mengonfig urasi aplikasi Anda untuk membaca dari cache terlebih dahulu, kembali ke database jika item yang diminta tidak ada.<br>• Database relasional seperti Amazon dan RDS Aurora, dan database Tidak ada yang disediakan seperti SQL Neptunus dan Amazon DocumentDB semuanya mendukung penambahan replika baca untuk menurunkan bagian baca dari beban kerja.<br>• Basis data nirserver seperti DynamoDB akan menskalakan secara otomatis. Pastikan Anda memiliki cukup unit kapasitas baca (RCU) yang disediakan untuk menangani beban kerja. |
| Menskalakan penulisan (seperti penyerpihan kunci partisi atau memperkenalkan antrean) | • Untuk database relasional, Anda dapat meningkatkan ukuran instans untuk mengakomodasi peningkatan beban kerja |

| Opsi Konfigurasi | Contoh |
| --- | --- |
| | atau meningkatkan penyediaan IOPs untuk memungkinkan peningkatan throughput ke penyimpanan yang mendasarinya.<br>• Anda juga dapat membuat antrean di depan basis data Anda, bukan menulis secara langsung ke basis data. Dengan pola-pola ini, Anda dapat memisahkan penyerapa n dari basis data dan mengontrol tingkat aliran, sehingga basis data tidak kewalahan .<br>• Mengganti pembuatan transaksi berdurasi pendek dengan pembuatan batch permintaa n penulisan dapat membantu Anda meningkatkan throughput dalam basis data relasional dengan volume penulisan yang tinggi.<br>• Database tanpa server seperti DynamoDB dapat menskalakan throughput penulisan secara otomatis atau dengan menyesuai kan unit kapasitas tulis yang disediakan () tergantung pada mode kapasitas. WCU<br>• Anda tetap dapat menjumpai masalah dengan partisi panas ketika Anda mencapai batas throughput pada kunci partisi tertentu. Hal ini dapat dimitigasi dengan memilih distribusi kunci partisi yang lebih merata atau dengan memisah penulisan kunci partisi (write-sharding). |

| Opsi Konfigurasi | Contoh |
| --- | --- |
| Kebijakan untuk mengelola siklus hidup set data Anda | • Anda dapat menggunakan Siklus Hidup Amazon S3 untuk mengelola objek-objek Anda di sepanjang siklus hidupnya. Jika pola akses Anda tidak diketahui, berubah, atau tidak dapat diprediksi, Anda dapat menggunakan Amazon S3 Intelligent-Tiering, yang akan memantau pola akses dan secara otomatis memindahkan objek yang belum diakses ke tingkat akses yang berbiaya lebih rendah. Anda dapat memanfaatkan metrik Lensa Penyimpanan Amazon S3 untuk melakukan identifikasi terhadap peluang dan celah pengoptimalan dalam manajemen siklus hidup.<br>• Manajemen EFS siklus hidup Amazon secara otomatis mengelola penyimpanan file untuk sistem file Anda. |
| Manajemen koneksi dan pooling | • Amazon RDS Proxy dapat digunakan dengan Amazon RDS dan Aurora untuk mengelola koneksi ke database.<br>• Basis data nirserver seperti DynamoDB tidak terkait dengan koneksi apa pun, tetapi pertimbangkan kapasitas yang tersedia atau kebijakan penskalaan otomatis untuk mengatasi lonjakan beban. |

- Lakukan uji coba dan uji tolok ukur di lingkungan non-produksi untuk mengidentifikasi opsi konfigurasi mana yang dapat memenuhi persyaratan-persyaratan beban kerja Anda.
- Setelah melakukan uji coba, rencanakan migrasi dan validasi metrik-metrik kinerja Anda.
- Gunakan alat AWS pemantauan (seperti Amazon CloudWatch) dan pengoptimalan (seperti Amazon S3 Storage Lens) untuk terus mengoptimalkan penyimpanan data Anda menggunakan pola penggunaan dunia nyata.

Sumber daya

Dokumen terkait:

- Penyimpanan Cloud dengan AWS
- Jenis EBS Volume Amazon
- EC2Penyimpanan Amazon
- AmazonEFS: EFS Kinerja Amazon
- Amazon FSx untuk Kinerja Kilau
- Amazon FSx untuk Kinerja Server File Windows
- Amazon S3 Glacier: Dokumentasi Glacier S3
- Amazon S3: Pertimbangan Tingkat Permintaan dan Performa
- Karakteristik Amazon EBS I/O
- Basis Data Cloud dengan AWS
- AWS Caching Basis Data
- DynamoDB Accelerator
- Video praktik terbaik Amazon Aurora
- Kinerja Amazon Redshift
- 10 kiat kinerja teratas Amazon Athena
- Praktik terbaik Amazon Redshift Spectrum
- Praktik terbaik Amazon DynamoDB

Video terkait:

- AWS re:Invent 2023: Meningkatkan efisiensi Amazon Elastic Block Store dan menjadi lebih hemat biaya
- AWS re:Invent 2023: Optimalisasi harga dan kinerja penyimpanan dengan Amazon Simple Storage Service
- AWS re:Invent 2023: Membangun dan mengoptimalkan danau data di Amazon Simple Storage Service
- AWS re:invent 2023: Apa yang baru dengan penyimpanan file AWS
- AWS re:Invent 2023: Memahami lebih dalam tentang Amazon DynamoDB

Contoh terkait:

- AWS Workshop Database yang Dibangun Tujuan
- Basis Data untuk Pengembang
- AWS Hari Perendaman Arsitektur Data Modern
- Amazon EBS Autoscale
- Contoh-contoh Amazon S3
- Contoh Amazon DynamoDB
- AWS Sampel migrasi basis data
- Lokakarya Modernisasi Basis Data
- Bekerja dengan parameter di Amazon Anda RDS untuk Postgress DB

PERF03-BP03 Kumpulkan dan rekam metrik kinerja penyimpanan data

Lacak dan rekam metrik-metrik kinerja yang relevan untuk penyimpanan data Anda guna memahami kinerja dari solusi manajemen data Anda. Metrik-metrik ini dapat membantu Anda mengoptimalkan penyimpanan data Anda, memastikan terpenuhinya persyaratan-persyaratan beban kerja Anda, dan memberikan gambaran umum yang jelas tentang kinerja beban kerja tersebut.

Anti-pola umum:

- Anda hanya menggunakan pencarian file log manual untuk mencari metrik.
- Anda hanya mempublikasikan metrik ke alat-alat internal yang digunakan tim Anda dan tidak memiliki gambaran yang komprehensif tentang beban kerja Anda.
- Anda hanya menggunakan metrik-metrik default yang dicatat oleh perangkat lunak pemantauan Anda yang dipilih.
- Anda hanya meninjau metrik-metrik tersebut ketika terdapat masalah.
- Anda hanya memantau metrik-metrik tingkat sistem dan tidak merekam metrik-metrik akses atau penggunaan data.

Manfaat menerapkan praktik terbaik ini: Memiliki dasar acuan kinerja membantu Anda memahami perilaku normal dan persyaratan beban kerja. Pola abnormal dapat diidentifikasi dan diperbaiki lebih cepat sehingga akan meningkatkan kinerja dan keandalan penyimpanan data.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk memantau kinerja penyimpanan data, Anda harus merekam beberapa metrik kinerja secara selama periode waktu tertentu. Dengan begitu Anda dapat mendeteksi anomali yang terjadi dan mengukur kinerja berdasarkan metrik bisnis untuk memastikan bahwa kebutuhan beban kerja Anda terpenuhi.

Metrik harus menyertakan sistem yang medasari yang mendukung metrik penyimpanan data dan metrik basis data. Metrik sistem yang mendasarinya mungkin mencakup CPU pemanfaatan, memori, penyimpanan disk yang tersedia, I/O disk, rasio hit cache, dan metrik masuk dan keluar jaringan, sedangkan metrik penyimpanan data mungkin mencakup transaksi per detik, kueri teratas, tingkat kueri rata-rata, waktu respons, penggunaan indeks, kunci tabel, batas waktu kueri, dan jumlah koneksi yang terbuka. Data-data ini sangat penting untuk memahami kinerja beban kerja dan bagaimana solusi manajemen data digunakan. Gunakan metrik-metrik ini sebagai bagian dari pendekatan berbasis data yang digunakan untuk mengatur dan mengoptimalkan sumber daya beban kerja Anda.

Gunakan alat, pustaka, dan sistem yang merekam pengukuran kinerja yang terkait dengan kinerja basis data.

Langkah-langkah implementasi

- Identifikasi metrik-metrik kinerja utama yang perlu dilacak oleh penyimpanan data Anda.
  - Metrik dan dimensi Amazon S3
  - Memantau metrik untuk instans Amazon RDS
  - Memantau beban DB dengan Performance Insights di Amazon RDS
  - Ringkasan Pemantauan yang Ditingkatkan
  - Dimensi dan Metrik DynamoDB
  - Memantau DynamoDB Accelerator
  - Memantau Amazon MemoryDB dengan Amazon CloudWatch
  - Metrik Apa yang Harus Saya Pantau?
  - Memantau kinerja klaster Amazon Redshift
  - Metrik dan Dimensi Timestream
  - CloudWatch Metrik Amazon untuk Amazon Aurora
  - Pencatatan log dan pemantauan di Amazon Keyspaces (untuk Apache Cassandra)
  - Memantau Sumber Daya Amazon Neptune

- Gunakan solusi pencatatan log dan pemantauan yang disetujui untuk mengumpulkan metrik-metrik ini. Amazon CloudWatch dapat mengumpulkan metrik di seluruh sumber daya dalam arsitektur Anda. Anda juga dapat mengumpulkan dan mempublikasikan metrik-metrik kustom untuk menampilkan metrik-metrik bisnis atau turunan. Gunakan CloudWatch atau solusi pihak ketiga untuk menyetel alarm yang menunjukkan kapan ambang batas dilanggar.
- Periksa apakah pemantauan penyimpanan data dapat terbantu dengan solusi machine learning yang mendeteksi adanya anomali kinerja.
  - Amazon DevOps Guru untuk Amazon RDS memberikan visibilitas ke masalah kinerja dan membuat rekomendasi untuk tindakan korektif.
- Atur konfigurasi retensi data dalam solusi pemantauan dan pencatatan log Anda agar sesuai dengan tujuan-tujuan keamanan dan operasional Anda.
  - Retensi data default untuk CloudWatch metrik
  - Retensi data default untuk CloudWatch Log

Sumber daya

Dokumen terkait:

- Caching Basis Data AWS
- 10 kiat kinerja teratas Amazon Athena
- Video praktik terbaik Amazon Aurora
- DynamoDB Accelerator
- Praktik terbaik Amazon DynamoDB
- Praktik terbaik Amazon Redshift Spectrum
- Kinerja Amazon Redshift
- Database Cloud dengan AWS
- RDSPerformance Insights Amazon

Video terkait:

- AWS re:invent 2022 - Pemantauan kinerja dengan Amazon dan RDS Aurora, menampilkan Autodesk
- Pemantauan dan Penyetelan Kinerja Basis Data dengan Amazon DevOps Guru untuk Amazon RDS

- AWS re:invent 2023 - Apa yang baru dengan penyimpanan file AWS
- AWS RE: invent 2023 - Menyelam jauh ke Amazon DynamoDB
- AWS re:invent 2023 - Membangun dan mengoptimalkan data lake di Amazon S3
- AWS re:invent 2023 - Apa yang baru dengan penyimpanan file AWS
- AWS RE: invent 2023 - Menyelam jauh ke Amazon DynamoDB
- Praktik Terbaik untuk Memantau Beban Kerja Redis di Amazon ElastiCache

Contoh terkait:

- Kerangka Kerja Pengumpulan Metrik Penyerapan Set Data AWS
- Lokakarya RDS Pemantauan Amazon
- AWS Workshop Database yang Dibangun Tujuan

PERF03-BP04 Menerapkan strategi untuk meningkatkan kinerja kueri di penyimpanan data

Terapkan strategi-strategi untuk mengoptimalkan data dan meningkatkan kueri data untuk memungkinkan skalabilitas yang lebih besar dan kinerja yang efisien untuk beban kerja Anda.

Anti-pola umum:

- Anda tidak mempartisi data yang ada di dalam penyimpanan data Anda.
- Anda menyimpan data hanya dalam satu format file di dalam penyimpanan data Anda.
- Anda tidak menggunakan indeks di penyimpanan data Anda.

Manfaat menerapkan praktik terbaik ini: Optimisasi data dan performa kueri menghasilkan efisiensi yang lebih tinggi, biaya lebih rendah, dan pengalaman pengguna yang lebih baik.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Optimalisasi data dan penyetelan kueri merupakan aspek-aspek penting efisiensi kinerja dalam sebuah penyimpanan data, karena berdampak pada kinerja dan responsivitas seluruh beban kerja cloud. Kueri yang tidak dioptimalkan dapat menyebabkan terjadinya penggunaan dan kemacetan sumber daya yang lebih besar, yang dapat mengurangi efisiensi penyimpanan data secara keseluruhan.

Pengoptimalan data mencakup beberapa teknik untuk memastikan penyimpanan data dan akses data yang efisien. Hal ini juga dapat membantu Anda meningkatkan performa kueri dalam penyimpanan data. Strategi utamanya mencakup partisi data, kompresi data, dan denormalisasi data, yang akan membantu optimalisasi penyimpanan dan akses data.

Langkah-langkah implementasi

- Pahami dan analisis kueri data penting yang dilakukan di dalam penyimpanan data Anda.
- Identifikasi kueri-kueri yang berjalan lambat di dalam penyimpanan data Anda dan gunakan rencana kueri untuk memahami statusnya saat ini.
  - Menganalisis rencana kueri di Amazon Redshift
  - Menggunakan EXPLAIN dan EXPLAIN ANALYZE di Athena
- Terapkan strategi-strategi untuk meningkatkan kinerja kueri. Beberapa strategi utamanya meliputi:
  - Menggunakan format file kolumnar (seperti Parket atau). ORC
  - Mengompresi data di dalam penyimpanan data untuk mengurangi ruang penyimpanan dan operasi I/O.
  - Melakukan partisi data untuk membagi data menjadi bagian-bagian yang lebih kecil dan mengurangi waktu pemindaian data.
    - Melakukan partisi data di Athena
    - Partisi dan distribusi data
  - Pengindeksan data pada kolom umum dalam kueri.
  - Gunakan tampilan terwujud untuk kueri yang sering dibuat.
    - Memahami tampilan terwujud
    - Membuat tampilan terwujud di Amazon Redshift
  - Pilih operasi gabungan yang tepat untuk kueri. Saat Anda menggabungkan dua tabel, tentukan tabel yang lebih besar berada di sisi kiri gabungan dan tabel yang lebih kecil berada di sisi kanan gabungan.
  - Solusi cache terdistribusi untuk meningkatkan latensi dan mengurangi jumlah operasi I/O basis data.
  - Pemeliharaan rutin seperti pemvakuman, pengindeksan ulang, dan menjalankan statistik.
- Lakukan eksperimen dan uji strategi di sebuah lingkungan non-produksi.

Sumber daya

Dokumen terkait:

- Video praktik terbaik Amazon Aurora
- Kinerja Amazon Redshift
- 10 kiat kinerja teratas Amazon Athena
- Caching Basis Data AWS
- Praktik Terbaik untuk Menerapkan Amazon ElastiCache
- Melakukan partisi data di Athena

Video terkait:

- AWS RE: invent 2023 - AWS praktik terbaik pengoptimalan biaya penyimpanan
- AWS re:invent 2022 - Pemantauan kinerja dengan Amazon dan RDS Aurora, menampilkan Autodesk
- Mengoptimalkan Kueri Amazon Athena dengan Alat Analisis Kueri Baru

Contoh terkait:

- Amazon S3 Select - Mengkueri data tanpa server atau basis data
- AWS Workshop Database yang Dibangun Tujuan

PERF03-BP05 Menerapkan pola akses data yang memanfaatkan caching

Implementasikan pola-pola akses yang dapat memanfaatkan caching data untuk pengambilan data yang sering diakses dengan cepat.

Anti-pola umum:

- Anda menyimpan cache data yang sering berubah.
- Anda mengandalkan data dalam cache seolah-olah data tersebut disimpan dengan durabilitas tinggi dan selalu tersedia.
- Anda tidak mempertimbangkan konsistensi data cache Anda.
- Anda tidak memantau efisiensi dari implementasi caching Anda.

Manfaat menerapkan praktik terbaik ini: Menyimpan data dalam cache dapat meningkatkan latensi baca, throughput baca, pengalaman pengguna, dan efisiensi secara keseluruhan, serta mengurangi biaya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Cache adalah sebuah komponen perangkat lunak atau perangkat keras yang dimaksudkan untuk menyimpan data sehingga permintaan di masa mendatang untuk data yang sama dapat dilayani dengan lebih cepat atau lebih efisien. Data yang disimpan dalam cache dapat direkonstruksi jika data tersebut hilang dengan mengulangi perhitungan sebelumnya atau mengambilnya dari tempat penyimpanan data lain.

Caching data dapat menjadi salah satu strategi yang paling efektif untuk meningkatkan performa aplikasi Anda secara keseluruhan dan mengurangi beban pada sumber data primer yang mendasarinya. Data dapat di-cache di berbagai tingkatan dalam aplikasi, seperti dalam aplikasi yang membuat panggilan jarak jauh, yang dikenal sebagai caching sisi klien, atau dengan menggunakan layanan sekunder cepat untuk menyimpan data, yang dikenal sebagai cache jarak jauh.

Caching sisi klien

Dengan melakukan caching sisi klien, setiap klien (aplikasi atau layanan yang mengkueri penyimpanan data backend) dapat menyimpan hasil kueri unik mereka secara lokal selama jangka waktu tertentu. Hal ini dapat mengurangi jumlah permintaan di seluruh jaringan ke sebuah penyimpanan data dengan memeriksa cache klien lokal terlebih dahulu. Jika hasilnya tidak ada, aplikasi kemudian dapat mengkueri penyimpanan data tersebut dan menyimpan hasilnya secara lokal. Dengan pola ini, setiap klien dapat menyimpan data di lokasi terdekat yang mungkin digunakan (klien itu sendiri), sehingga menghasilkan latensi yang serendah mungkin. Klien juga dapat terus melayani beberapa kueri ketika penyimpanan data backend tidak tersedia, sehingga akan meningkatkan ketersediaan sistem secara keseluruhan.

Salah satu kelemahan pendekatan ini adalah ketika ada beberapa klien yang terlibat, semuanya dapat menyimpan data cache yang sama secara lokal. Hal ini mengakibatkan adanya penggunaan penyimpanan duplikat dan inkonsistensi data antara klien-klien tersebut. Salah satu klien mungkin melakukan caching terhadap hasil suatu kueri, dan satu menit kemudian klien lainnya dapat menjalankan kueri yang sama dan mendapatkan hasil kueri yang berbeda.

Caching jarak jauh

Untuk mengatasi masalah duplikat data yang terjadi antar klien, suatu layanan eksternal cepat, atau cache jarak jauh, dapat digunakan untuk menyimpan data yang di-kueri. Alih-alih memeriksa penyimpanan data lokal, masing-masing klien akan memeriksa cache jarak jauh sebelum mengkueri penyimpanan data backend. Strategi ini memungkinkan respons yang lebih konsisten di antara klien, efisiensi yang lebih baik pada data yang disimpan, dan volume data cache yang lebih tinggi karena ruang penyimpanannya diskalakan secara independen tanpa terikat klien.

Kelemahan cache jarak jauh adalah sistem tersebut mungkin mengalami latensi yang lebih tinggi secara keseluruhan karena diperlukan lompatan jaringan tambahan untuk memeriksa cache jarak jauh. Caching sisi klien dapat digunakan bersama caching jarak jauh untuk melakukan caching multi-tingkat sehingga dapat meningkatkan latensi.

Langkah-langkah implementasi

- Identifikasi database, APIs dan layanan jaringan yang dapat mengambil manfaat dari caching. Layanan yang memiliki beban kerja baca berat, memiliki read-to-write rasio tinggi, atau mahal untuk skala adalah kandidat untuk caching.
  - Caching Basis Data
  - Mengaktifkan API caching untuk meningkatkan daya tanggap
- Identifikasi jenis strategi caching yang tepat yang paling sesuai dengan pola akses Anda.
  - Strategi pembuatan cache
  - Solusi Penerapan Cache AWS
- Ikuti Praktik Terbaik Caching untuk penyimpanan data Anda.
- Konfigurasikan strategi pembatalan cache, seperti a time-to-live (TTL), untuk semua data yang menyeimbangkan kesegaran data dan mengurangi tekanan pada backend datastore.
- Aktifkan fitur seperti percobaan ulang koneksi otomatis, penundaan eksponensial, batas waktu sisi klien, dan pooling koneksi di dalam klien, jika tersedia, karena fitur-fitur tersebut dapat meningkatkan performa dan keandalan.
  - Praktik terbaik: Klien Redis dan Amazon ElastiCache (OSSRedis)
- Pantau laju hit cache dengan target 80% atau lebih tinggi. Nilai yang lebih rendah mungkin menunjukkan ukuran cache yang tidak mencukupi atau pola akses yang tidak diuntungkan dengan melakukan caching.
  - Metrik Apa yang Harus Saya Pantau?
  - Praktik terbaik untuk memantau beban kerja Redis di Amazon ElastiCache

  - Memantau praktik terbaik dengan Amazon ElastiCache (RedisOSS) menggunakan Amazon CloudWatch
- Implementasikan replikasi data untuk melimpahkan beban baca ke beberapa instans dan meningkatkan performa dan ketersediaan pembacaan data.

Sumber daya

Dokumen terkait:

- Menggunakan Lensa Amazon ElastiCache Well-Architected
- Memantau praktik terbaik dengan Amazon ElastiCache (RedisOSS) menggunakan Amazon CloudWatch
- Metrik Apa yang Harus Saya Pantau?
- Kinerja dalam Skala dengan ElastiCache whitepaper Amazon
- Tantangan dan strategi caching

Video terkait:

- Jalur ElastiCache Pembelajaran Amazon
- Desain untuk sukses dengan praktik ElastiCache terbaik Amazon
- AWS Re:invent 2020 - Desain untuk sukses dengan praktik terbaik Amazon ElastiCache
- AWS re: invent 2023 - [] LAUNCH Memperkenalkan Amazon Tanpa Server ElastiCache
- AWS re:invent 2022 - 5 cara bagus untuk menata ulang lapisan data Anda dengan Redis
- AWS Re:invent 2021 - Menyelam jauh di Amazon ElastiCache (Redis) OSS

Contoh terkait:

- Meningkatkan kinerja SQL database saya dengan Amazon ElastiCache (OSSRedis)

## Jaringan dan Pengiriman Konten

Pertanyaan

- PERF4. Bagaimana cara memilih dan mengonfigurasi sumber daya jaringan dalam beban kerja Anda?

## PERF4. Bagaimana cara memilih dan mengonfigurasi sumber daya jaringan dalam beban kerja Anda?

Solusi jaringan optimal untuk beban kerja bervariasi berdasarkan latensi, persyaratan throughput, jitter, dan bandwith. Batas fisik, seperti sumber daya on-premise atau pengguna, menentukan opsi lokasi. Batas-batas ini dapat diimbangi dengan penempatan sumber daya atau lokasi edge.

Praktik terbaik

- PERF04-BP01 Memahami bagaimana jaringan memengaruhi kinerja
- PERF04-BP02 Mengevaluasi fitur jaringan yang tersedia
- PERF04-BP03 Pilih konektivitas khusus yang sesuai atau untuk beban kerja Anda VPN
- PERF04-BP04 Gunakan load balancing untuk mendistribusikan lalu lintas di berbagai sumber daya
- PERF04-BP05 Pilih protokol jaringan untuk meningkatkan kinerja
- PERF04-BP06 Pilih lokasi beban kerja Anda berdasarkan persyaratan jaringan
- PERF04-BP07 Optimalkan konfigurasi jaringan berdasarkan metrik

### PERF04-BP01 Memahami bagaimana jaringan memengaruhi kinerja

Lakukan analisis dan pahami bagaimana keputusan-keputusan terkait jaringan memengaruhi beban kerja Anda untuk memberikan performa yang efisien dan pengalaman pengguna yang lebih baik.

Anti-pola umum:

- Semua lalu lintas mengalir melalui pusat data Anda.
- Anda merutekan semua lalu lintas melalui firewall pusat, bukan menggunakan alat keamanan jaringan cloud-native.
- Anda menyediakan AWS Direct Connect koneksi tanpa memahami persyaratan penggunaan yang sebenarnya.
- Anda tidak mempertimbangkan karakteristik beban kerja dan biaya overhead enkripsi ketika menentukan solusi-solusi jaringan Anda.
- Anda menggunakan konsep dan strategi on-premise untuk solusi-solusi jaringan di cloud.

Manfaat menerapkan praktik terbaik ini: Memahami bagaimana jaringan memengaruhi kinerja beban kerja membantu Anda mengidentifikasi potensi hambatan, meningkatkan pengalaman pengguna, meningkatkan keandalan, dan menurunkan pemeliharaan operasional saat beban kerja berubah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Jaringan bertanggung jawab atas konektivitas antara komponen aplikasi, layanan cloud, jaringan edge, dan data on-premise, oleh karena itu, jaringan dapat sangat memengaruhi performa beban kerja. Selain performa beban kerja, pengalaman pengguna juga dapat terpengaruh oleh latensi jaringan, bandwidth, protokol, lokasi, kemacetan jaringan, jitter, throughput, dan aturan-aturan perutean.

Buatlah daftar terdokumentasi kebutuhan jaringan dari beban kerja termasuk latensi, ukuran paket, aturan perutean, protokol, dan pola lalu lintas pendukung. Tinjau solusi-solusi jaringan yang tersedia dan identifikasi layanan mana yang memenuhi karakteristik jaringan beban kerja Anda. Jaringan berbasis cloud dapat dengan cepat dibangun kembali, jadi Anda harus meningkatkan arsitektur jaringan Anda seiring berjalannya waktu untuk meningkatkan efisiensi kinerja.

Langkah-langkah implementasi:

- Tentukan dan dokumentasikan persyaratan performa jaringan, termasuk metrik-metrik seperti latensi jaringan, bandwidth, protokol, lokasi, pola lalu lintas (lonjakan dan frekuensi), throughput, enkripsi, inspeksi, dan aturan-aturan perutean.
- Pelajari tentang layanan AWS jaringan utama seperti VPCsAWS Direct Connect, Elastic Load Balancing (ELB), dan Amazon Route 53.
- Rekam karakteristik jaringan utama berikut:

| Karakteristik | Alat dan metrik |
| --- | --- |
| Karakteristik jaringan dasar | • Log Alur VPC<br>• AWS Transit Gateway Log Aliran<br>• AWS Transit Gateway metrik<br>• AWS PrivateLink metrik |
| Karakteristik jaringan aplikasi | • Elastic Fabric Adapter<br>• AWS App Mesh metrik<br>• Metrik Amazon API Gateway |
| Karakteristik jaringan edge | • CloudFront Metrik Amazon<br>• Metrik Amazon Route 53 |

| Karakteristik | Alat dan metrik |
| --- | --- |
| | • AWS Global Accelerator metrik |
| Karakteristik jaringan hibrida | • AWS Direct Connect metrik<br>• AWS Site-to-Site VPN metrik<br>• AWS Client VPN metrik<br>• AWS Cloud WANmetrik |
| Karakteristik jaringan keamanan | • AWS Shield, AWS WAF, dan AWS Network Firewall metrik |
| Karakteristik penelusuran | • AWS X-Ray<br>• VPCReachability Analyzer<br>• Penganalisis Akses Jaringan<br>• Amazon Inspector<br>• Amazon CloudWatch RUM |

- Buat tolok ukur dan uji kinerja jaringan:
  - Benchmark throughput jaringan, karena beberapa faktor dapat memengaruhi kinerja EC2 jaringan Amazon ketika instance berada dalam keadaan yang sama. VPC Ukur bandwidth jaringan antara instans Amazon EC2 Linux dalam hal yang samaVPC.
  - Lakukan uji beban untuk bereksperimen dengan solusi dan opsi jaringan.

Sumber daya

Dokumen terkait:

- Penyeimbang Beban Aplikasi
- EC2Jaringan yang Ditingkatkan di Linux
- EC2Jaringan yang Ditingkatkan di Windows
- EC2Grup Penempatan
- Mengaktifkan Jaringan yang Ditingkatkan dengan Adaptor Jaringan Elastis (ENA) di Instans Linux
- Penyeimbang Beban Jaringan
- Produk Networking dengan AWS

- Transit Gateway
- Transisi ke latensi berbasis perutean di Amazon Route 53
- VPCTitik akhir

Video terkait:

- AWS re: invent 2023 - yayasan jaringan AWS
- AWS re:invent 2023 - Apa yang dapat jaringan lakukan untuk aplikasi Anda?
- AWS Re:invent 2023 - Desain canggih VPC dan kemampuan baru
- AWS re:invent 2023 - Panduan pengembang untuk jaringan cloud
- AWS re:invent 2019 - Konektivitas ke AWS dan arsitektur jaringan hybrid AWS
- AWS re:invent 2019 - Mengoptimalkan Kinerja Jaringan untuk Instans Amazon EC2
- AWS Summit Online - Meningkatkan Kinerja Jaringan Global untuk Aplikasi
- AWS Re:invent 2020 - Melakukan praktik dan kiat terbaik jaringan dengan Well-Architected Framework
- AWS Re:invent 2020 - praktik terbaik AWS jaringan dalam migrasi skala besar

Contoh terkait:

- AWS Transit Gateway dan Solusi Keamanan yang Dapat Diskalakan
- AWS Lokakarya Jaringan
- Lokakarya Firewall Jaringan Langsung
- Mengamati dan Mendiagnosis Jaringan Anda AWS
- Menemukan dan menangani Kesalahan Konfigurasi Jaringan di AWS

PERF04-BP02 Mengevaluasi fitur jaringan yang tersedia

Evaluasi fitur jaringan yang ada di cloud yang dapat meningkatkan kinerja. Ukur dampak fitur-fitur ini melalui pengujian, metrik, dan analisis. Misalnya, manfaatkan fitur tingkat jaringan yang tersedia untuk mengurangi latensi, jarak jaringan, atau masalah kecepatan (jitter).

Anti-pola umum:

- Anda hanya menggunakan satu Wilayah karena di sanalah lokasi fisik kantor pusat Anda.

- Anda menggunakan firewall, bukan grup keamanan, untuk memfilter lalu lintas.
- Anda melanggar TLS pemeriksaan lalu lintas daripada mengandalkan grup keamanan, kebijakan titik akhir, dan fungsionalitas cloud-native lainnya.
- Anda hanya menggunakan segmentasi berbasis subnet, bukan grup keamanan.

Manfaat menerapkan praktik terbaik ini: Mengevaluasi semua fitur dan opsi layanan dapat meningkatkan performa beban kerja Anda, menurunkan biaya infrastruktur, mengurangi upaya yang diperlukan untuk memelihara beban kerja Anda, dan meningkatkan postur keamanan Anda secara keseluruhan. Anda dapat menggunakan AWS tulang punggung global untuk memberikan pengalaman jaringan yang optimal bagi pelanggan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS menawarkan layanan seperti AWS Global Acceleratordan Amazon CloudFront yang dapat membantu meningkatkan kinerja jaringan, sementara sebagian besar AWS layanan memiliki fitur produk (seperti fitur Amazon S3 Transfer Acceleration) untuk mengoptimalkan lalu lintas jaringan.

Tinjau opsi konfigurasi terkait jaringan mana yang tersedia untuk Anda serta bagaimana dampaknya terhadap beban kerja Anda. Optimalisasi performa bergantung pada pemahaman tentang bagaimana opsi-opsi ini berinteraksi dengan arsitektur Anda serta dampaknya terhadap performa terukur dan pengalaman pengguna.

Langkah-langkah implementasi

- Buatlah sebuah daftar komponen beban kerja.
  - Pertimbangkan AWS Cloud WANuntuk membangun, mengelola, dan memantau jaringan organisasi Anda saat membangun jaringan global terpadu.
  - Pantau jaringan global dan inti Anda dengan metrik Amazon CloudWatch Logs. Manfaatkan Amazon CloudWatch RUM, yang memberikan wawasan untuk membantu mengidentifikasi, memahami, dan meningkatkan pengalaman digital pengguna.
  - Lihat latensi jaringan agregat antara Wilayah AWS dan Availability Zones, serta dalam setiap Availability Zone, gunakan AWS Network Manageruntuk mendapatkan wawasan tentang bagaimana kinerja aplikasi Anda terkait dengan kinerja jaringan yang mendasarinya. AWS
  - Gunakan alat database manajemen konfigurasi (CMDB) yang ada atau layanan seperti AWS Configuntuk membuat inventaris beban kerja Anda dan cara konfigurasinya.

- Jika ini adalah beban kerja yang ada sekarang, identifikasi dan dokumentasikan tolok ukur untuk metrik-metrik performa Anda, yang fokus pada hambatan dan area yang perlu ditingkatkan. Metrik-metrik jaringan terkait performa akan berbeda untuk setiap beban kerja berdasarkan persyaratan bisnis dan karakteristik beban kerja. Sebagai permulaan, metrik-metrik berikut ini mungkin penting untuk ditinjau untuk beban kerja Anda: bandwidth, latensi, kehilangan paket, jitter, dan transmisi ulang.
- Jika ini adalah sebuah beban kerja baru, lakukan uji beban untuk mengidentifikasi kemacetan kinerja.
- Untuk hambatan-hambatan performa yang Anda identifikasi, tinjau opsi konfigurasi untuk solusi Anda guna mengidentifikasi peluang peningkatan performa. Lihat opsi dan fitur jaringan utama berikut ini:

| Peluang peningkatan | Solusi |
| --- | --- |
| Jalur atau rute jaringan | Gunakan Penganalisis Akses Jaringan untuk mengidentifikasi jalur atau rute. |
| Protokol jaringan | Lihat PERF04-BP05 Pilih protokol jaringan untuk meningkatkan kinerja |
| Topologi jaringan | Evaluasi pengorbanan operasional dan kinerja Anda antara VPCPeering dan AWS Transit Gatewaysaat menghubungkan beberapa akun. AWS Transit Gateway menyederhanakan cara Anda menghubun gkan semuaVPCs, yang dapat menjangka u ribuan Akun AWS dan ke jaringan lokal. Bagikan Anda AWS Transit Gateway di antara beberapa akun menggunakan AWS Resource Access Manager.<br><br>Lihat PERF04-BP03 Pilih konektivitas khusus yang sesuai atau untuk beban kerja Anda VPN |
| Layanan jaringan | AWS Global Acceleratoradalah layanan jaringan yang meningkatkan kinerja lalu lintas |

| Peluang peningkatan | Solusi |
| --- | --- |
| | pengguna Anda hingga 60% menggunakan infrastruktur jaringan AWS global.<br><br>Amazon CloudFront dapat meningkatkan kinerja pengiriman konten beban kerja dan latensi Anda secara global.<br><br>Gunakan Lambda @edge untuk menjalank an fungsi yang menyesuaikan konten yang CloudFront memberikan lebih dekat ke pengguna, mengurangi latensi, dan meningkat kan kinerja.<br><br>Amazon Route 53 menawarkan perutean berbasis latensi, perutean geolokasi, perutean kedekatan geografis, dan opsi perutean berbasis IP untuk membantu Anda meningkat kan kinerja beban kerja untuk audiens global. Identifikasi opsi-opsi perutean mana yang akan mengoptimalkan performa beban kerja Anda dengan meninjau lalu lintas beban kerja dan lokasi pengguna Anda saat beban kerja Anda terdistribusi secara global. |

| Peluang peningkatan | Solusi |
| --- | --- |
| Fitur sumber daya penyimpanan | Amazon S3 Transfer Acceleration adalah fitur yang memungkinkan pengguna eksternal mendapat manfaat dari pengoptimalan CloudFront jaringan untuk mengunggah data ke Amazon S3. Hal ini akan meningkatkan kemampuan transfer data dalam jumlah besar dari lokasi jarak jauh yang tidak memiliki koneksi khusus ke AWS Cloud.<br><br>Amazon S3 Multi-Region Access Points mereplikasi konten ke beberapa Wilayah dan menyederhanakan beban kerja dengan menyediakan satu titik akses. Saat Titik Akses Multi-Wilayah digunakan, Anda dapat meminta atau menulis data ke Amazon S3 dengan layanan yang mengidentifikasi bucket dengan latensi terendah. |

| Peluang peningkatan | Solusi |
| --- | --- |
| Fitur sumber daya komputasi | Elastic Network Interfaces (ENA) yang digunakan oleh EC2 instans Amazon, container, dan fungsi Lambda dibatasi pada basis per aliran. Tinjau grup penempatan Anda untuk mengoptimalkan throughput EC2 jaringan Anda. Untuk menghindari hambatan berdasarkan alur, rancang aplikasi Anda sedemikian rupa agar bisa menggunakan beberapa alur. Untuk memantau dan mendapatkan visibilitas ke metrik jaringan terkait komputasi Anda, gunakan CloudWatch Metrik dan ethtool. `ethtool`Perintah disertakan dalam ENA driver dan mengekspos metrik terkait jaringan tambahan yang dapat dipublikasikan sebagai metrik khusus. CloudWatch<br><br>Amazon Elastic Network Adapters (ENA) memberikan pengoptimalan lebih lanjut dengan memberikan throughput yang lebih baik untuk instans Anda dalam grup penempatan klaster.<br><br>Elastic Fabric Adapter (EFA) adalah antarmuka jaringan untuk EC2 instans Amazon yang memungkinkan Anda menjalankan beban kerja yang membutuhkan komunikasi internode tingkat tinggi dalam skala besar. AWS<br><br>Instans EBS yang dioptimalkan Amazon menggunakan tumpukan konfigurasi yang dioptimalkan dan menyediakan kapasitas tambahan khusus untuk meningkatkan I/O AmazonEBS. |

Sumber daya

Dokumen terkait:

- Penyeimbang Beban Aplikasi
- EC2Jaringan yang Ditingkatkan di Linux
- EC2Jaringan yang Ditingkatkan di Windows
- EC2Grup Penempatan
- Mengaktifkan Jaringan yang Ditingkatkan dengan Adaptor Jaringan Elastis (ENA) di Instans Linux
- Penyeimbang Beban Jaringan
- Produk Networking dengan AWS
- Transisi ke Latensi Berbasis Perutean di Amazon Route 53
- VPCTitik akhir
- Log Alur VPC

Video terkait:

- AWS Re: invent 2023 - Siap untuk apa selanjutnya? Merancang jaringan untuk pertumbuhan dan fleksibilitas
- AWS re: Invent 2023 - Desain canggih VPC dan kemampuan baru
- AWS re: Invent 2023 - Panduan pengembang untuk jaringan cloud
- AWS re:invent 2022 — Menyelam jauh pada infrastruktur jaringan AWS
- AWS re:invent 2019 - Konektivitas ke AWS dan arsitektur jaringan hybrid AWS
- AWS re: invent 2018 - Mengoptimalkan Kinerja Jaringan untuk Instans Amazon EC2
- AWS Global Accelerator

Contoh terkait:

- AWS Transit Gateway dan Solusi Keamanan yang Dapat Diskalakan
- AWS Lokakarya Jaringan
- Mengamati dan mendiagnosis jaringan Anda
- Menemukan dan menangani kesalahan konfigurasi jaringan pada AWS

PERF04-BP03 Pilih konektivitas khusus yang sesuai atau untuk beban kerja Anda VPN

Ketika diperlukan konektivitas hibrida untuk menghubungkan sumber daya on-premise dan cloud, sediakan bandwidth yang memadai untuk memenuhi persyaratan performa Anda. Perkirakan persyaratan bandwidth dan latensi untuk beban kerja hibrida Anda. Angka-angka ini akan mendorong persyaratan penyesuaian ukuran Anda.

Anti-pola umum:

- Anda hanya mengevaluasi VPN solusi untuk persyaratan enkripsi jaringan Anda.
- Anda tidak mengevaluasi opsi-opsi cadangan atau konektivitas redundan.
- Anda tidak mengidentifikasi semua persyaratan beban kerja (kebutuhan enkripsi, protokol, bandwidth, dan lalu lintas).

Manfaat menerapkan praktik terbaik ini: Memilih dan mengonfigurasi solusi konektivitas yang tepat akan meningkatkan keandalan beban kerja dan memaksimalkan performa. Dengan mengidentifikasi persyaratan beban kerja, merencanakan ke depan, dan mengevaluasi solusi hybrid, Anda dapat meminimalkan perubahan jaringan fisik yang mahal dan overhead operasional sambil meningkatkan biaya Anda. time-to-value

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kembangkan sebuah arsitektur jaringan hibrida berdasarkan kebutuhan bandwidth Anda. AWS Direct Connect akan memungkinkan Anda untuk menghubungkan jaringan on-premise Anda secara privat dengan AWS. Layanan ini ideal ketika Anda memerlukan bandwidth yang tinggi dan latensi yang rendah sekaligus mencapai performa yang konsisten. VPNKoneksi membuat koneksi aman melalui internet. VPN digunakan ketika yang diperlukan hanyalah sambungan sementara, ketika biaya menjadi pertimbangan, atau sebagai kontinjensi sambil menunggu terbentuknya konektivitas jaringan fisik yang kuat saat menggunakan AWS Direct Connect.

Jika persyaratan bandwidth Anda tinggi, Anda dapat mempertimbangkan beberapa AWS Direct Connect atau VPN layanan. Lalu lintas dapat menyeimbangkan beban di seluruh layanan, meskipun kami tidak merekomendasikan penyeimbangan beban antara AWS Direct Connect dan VPN karena perbedaan latensi dan bandwidth.

Langkah-langkah implementasi

- Perkirakan persyaratan-persyaratan bandwidth dan latensi aplikasi yang sudah Anda miliki.

  - Untuk beban kerja yang ada AWS, manfaatkan data dari sistem pemantauan jaringan internal Anda.
  - Untuk beban kerja baru atau lama yang data pemantauannya tidak Anda miliki, hubungi pemilik produk untuk menentukan metrik-metrik performa yang memadai dan memberikan pengalaman pengguna yang baik.
- Pilih koneksi khusus atau VPN sebagai opsi konektivitas Anda. Berdasarkan semua persyaratan beban kerja (enkripsi, bandwidth, dan kebutuhan lalu lintas), Anda dapat memilih AWS Direct Connect atau AWS VPN(atau keduanya). Diagram berikut dapat membantu Anda memilih jenis sambungan yang tepat.
  - AWS Direct Connect menyediakan konektivitas khusus ke lingkungan AWS , mulai dari 50 Mbps hingga 100 Gbps, menggunakan sambungan khusus atau sambungan yang di-host. Layanan ini memberikan Anda latensi yang terkelola dan terkontrol serta bandwidth yang tersedia agar beban kerja Anda dapat terhubung ke lingkungan-lingkungan lain secara efisien. Dengan menggunakan AWS Direct Connect mitra, Anda dapat memiliki end-to-end konektivitas dari berbagai lingkungan, menyediakan jaringan yang diperluas dengan kinerja yang konsisten. AWS menawarkan penskalaan bandwidth koneksi koneksi langsung menggunakan 100 Gbps asli, grup agregasi tautan (LAG), atau multipath () BGP dengan biaya sama. ECMP
  - AWS Site-to-Site VPNMenyediakan VPN layanan terkelola yang mendukung keamanan protokol internet (IPsec). Ketika VPN koneksi dibuat, setiap VPN koneksi mencakup dua terowongan untuk ketersediaan tinggi.
- Ikuti AWS dokumentasi untuk memilih opsi konektivitas yang sesuai:
  - Jika Anda memutuskan untuk menggunakan AWS Direct Connect, pilih bandwidth yang sesuai untuk konektivitas Anda.
  - Jika Anda menggunakan AWS Site-to-Site VPN di beberapa lokasi untuk terhubung ke Wilayah AWS, gunakan Site-to-SiteVPNkoneksi yang dipercepat untuk kesempatan meningkatkan kinerja jaringan.
  - Jika desain jaringan Anda terdiri dari IPSec VPN koneksi over AWS Direct Connect, pertimbangkan untuk menggunakan IP Pribadi VPN untuk meningkatkan keamanan dan mencapai segmentasi. AWS Site-to-Site IP pribadi VPN digunakan di atas antarmuka virtual transit (VIF).
  - AWS Direct Connect SiteLinkmemungkinkan membuat koneksi latensi rendah dan redundan antara pusat data Anda di seluruh dunia dengan mengirimkan data melalui jalur tercepat antar AWS Direct Connect lokasi, melewati. Wilayah AWS

- Lakukan validasi penyiapan konektivitas Anda sebelum deployment ke lingkungan produksi. Lakukan pengujian keamanan dan performa untuk memastikan persyaratan-persyaratan bandwidth, keandalan, latensi, dan kepatuhan Anda terpenuhi.
- Pantau performa dan penggunaan konektivitas Anda secara rutin dan optimalkan jika diperlukan.

Bagan alur performa penentu

Sumber daya

Dokumen terkait:

- Jaringan Produk dengan AWS
- AWS Transit Gateway
- VPCTitik akhir
- Membangun Infrastruktur Multi VPC AWS Jaringan yang Skalabel dan Aman
- Klien VPN

Video terkait:

- AWS re: invent 2023 - Membangun konektivitas jaringan hybrid dengan AWS
- AWS re: invent 2023 - Amankan konektivitas jarak jauh ke AWS
- AWS re:invent 2022 — Mengoptimalkan kinerja dengan Amazon CloudFront
- AWS re:invent 2019 - Konektivitas ke AWS dan arsitektur jaringan hybrid AWS
- AWS RE: invent 2020 - Connect AWS Transit Gateway

Contoh terkait:

- AWS Transit Gateway dan Solusi Keamanan yang Dapat Diskalakan
- AWS Lokakarya Jaringan

PERF04-BP04 Gunakan load balancing untuk mendistribusikan lalu lintas di berbagai sumber daya

Distribusikan lalu lintas di berbagai sumber daya atau layanan untuk memanfaatkan elastisitas yang ada di cloud untuk beban kerja Anda. Anda juga dapat menggunakan penyeimbang beban untuk memindahkan beban penghentian enkripsi guna meningkatkan performa dan keandalan, dan untuk mengelola serta merutekan lalu lintas secara efektif.

Anti-pola umum:

- Anda tidak mempertimbangkan persyaratan-persyaratan beban kerja Anda ketika memilih jenis penyeimbang beban.
- Anda tidak memanfaatkan fitur penyeimbang beban untuk mengoptimalkan performa.
- Beban kerja terpapar langsung ke internet tanpa penyeimbang beban.
- Anda merutekan semua lalu lintas internet melalui penyeimbang beban yang ada.
- Anda menggunakan TCP load balancing generik dan membuat setiap node komputasi menangani enkripsi. SSL

Manfaat menerapkan praktik terbaik ini: Penyeimbang beban menangani berbagai beban lalu lintas aplikasi Anda dalam satu atau beberapa Zona Ketersediaan dan menghadirkan ketersediaan yang tinggi, penskalaan otomatis, dan pemanfaatan yang lebih baik untuk beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Penyeimbang beban berfungsi sebagai titik masuk untuk beban kerja Anda, yakni titik asal penyeimbang beban mendistribusikan lalu lintas ke target backend Anda, seperti kontainer atau instans komputasi, untuk meningkatkan pemanfaatan.

Memilih jenis penyeimbang beban yang tepat adalah langkah pertama untuk mengoptimalkan arsitektur Anda. Mulailah dengan mencantumkan karakteristik beban kerja Anda, seperti protokol (sepertiTCP,,HTTP, atau WebSockets)TLS, jenis target (seperti instance, kontainer, atau tanpa server), persyaratan aplikasi (seperti koneksi yang berjalan lama, otentikasi pengguna, atau kekakuan), dan penempatan (seperti Wilayah, Zona Lokal, Pos Luar, atau isolasi zona).

AWS menyediakan beberapa model untuk aplikasi Anda untuk menggunakan load balancing. Application Load Balancer paling cocok untuk penyeimbangan beban HTTP dan HTTPS lalu lintas dan menyediakan routing permintaan lanjutan yang ditargetkan pada pengiriman arsitektur aplikasi modern, termasuk layanan mikro dan kontainer.

Network Load Balancer paling cocok untuk penyeimbangan beban TCP lalu lintas di mana kinerja ekstrem diperlukan. Penyeimbangan beban ini mampu menangani jutaan permintaan per detik sekaligus membuat latensi tetap rendah, serta dioptimalkan untuk menangani pola lalu lintas yang tidak stabil dan mendadak.

Elastic Load Balancing menyediakan manajemen sertifikat dan SSL TLS dekripsi terintegrasi, memungkinkan Anda fleksibilitas untuk mengelola SSL pengaturan penyeimbang beban secara terpusat dan melepaskan pekerjaan intensif dari beban kerja Anda. CPU

Setelah memilih penyeimbang beban yang tepat, Anda dapat mulai memanfaatkan fitur-fiturnya untuk mengurangi jumlah upaya yang harus dilakukan backend guna melayani lalu lintas.

Misalnya, menggunakan Application Load Balancer (ALB) dan Network Load Balancer NLB (), Anda dapat SSL melakukanTLS/encryption offloading, yang merupakan kesempatan untuk menghindari CPU jabat tangan TLS intensif diselesaikan oleh target Anda dan juga untuk meningkatkan manajemen sertifikat.

Ketika Anda SSL TLS mengonfigurasi/membongkar di penyeimbang beban Anda, itu menjadi bertanggung jawab atas enkripsi lalu lintas dari dan ke klien sambil mengirimkan lalu lintas yang tidak dienkripsi ke backend Anda, membebaskan sumber daya backend Anda dan meningkatkan waktu respons untuk klien.

Application Load Balancer juga dapat melayani HTTP /2 lalu lintas tanpa perlu mendukungnya pada target Anda. Keputusan sederhana ini dapat meningkatkan waktu respons aplikasi Anda, karena HTTP /2 menggunakan TCP koneksi lebih efisien.

Persyaratan latensi beban kerja Anda harus dipertimbangkan ketika menentukan arsitekturnya. Sebagai contoh, jika Anda memiliki aplikasi yang sensitif latensi, Anda dapat memutuskan untuk menggunakan Penyeimbang Beban Jaringan, yang menawarkan latensi yang sangat rendah. Alternatifnya, Anda dapat memutuskan untuk membawa beban kerja lebih dekat ke pelanggan dengan memanfaatkan Penyeimbang Beban Aplikasi di Zona Lokal AWS atau bahkan di AWS Outposts.

Pertimbangan lain untuk beban kerja yang sensitif latensi adalah penyeimbangan beban lintas zona. Dengan penyeimbangan beban lintas zona, setiap simpul penyeimbang beban mendistribusikan lalu lintas ke target terdaftar di semua Zona Ketersediaan yang diaktifkan.

Gunakan Auto Scaling (penskalaan otomatis) yang terintegrasi dengan penyeimbang beban Anda. Salah satu aspek penting dari sebuah sistem dengan performa yang efisien berkaitan dengan

penyesuaian ukuran sumber daya backend Anda. Untuk melakukannya, Anda dapat memanfaatkan integrasi penyeimbang beban untuk sumber daya target backend. Dengan menggunakan integrasi penyeimbang beban dengan grup Auto Scaling (penskalaan otomatis), target akan ditambahkan atau disingkirkan dari penyeimbang beban sebagaimana diperlukan untuk merespons lalu lintas masuk. Load balancer juga dapat diintegrasikan dengan Amazon dan ECSAmazon EKS untuk beban kerja kontainer.

- Amazon ECS - Penyeimbangan beban layanan
- Penyeimbangan beban aplikasi di Amazon EKS
- Penyeimbangan beban jaringan di Amazon EKS

Langkah-langkah implementasi

- Tentukan persyaratan-persyaratan penyeimbangan beban Anda, termasuk volume lalu lintas, ketersediaan, dan skalabilitas aplikasi.
- Pilih jenis penyeimbang beban yang tepat untuk aplikasi Anda.
  - Gunakan Application Load Balancer HTTP HTTPS untuk/beban kerja.
  - Gunakan Network Load Balancer untuk HTTP non-beban kerja yang berjalan pada atau. TCP UDP
  - Gunakan kombinasi keduanya (ALBsebagai target NLB) jika Anda ingin memanfaatkan fitur kedua produk. Misalnya, Anda dapat melakukan ini jika Anda ingin menggunakan statis IPs NLB bersama dengan perutean berbasis HTTP header dariALB, atau jika Anda ingin mengekspos HTTP beban kerja Anda ke file. AWS PrivateLink
  - Untuk perbandingan lengkap penyeimbang beban, lihat perbandingan ELBproduk.
- SSLTLSGunakan/pembongkaran jika memungkinkan.
  - KonfigurasikanHTTPS/TLSlistener dengan Application Load Balancer dan Network Load Balancer yang terintegrasi dengannya. AWS Certificate Manager
  - Perhatikan bahwa beberapa beban kerja mungkin memerlukan end-to-end enkripsi untuk alasan kepatuhan. Jika demikian, enkripsi wajib diaktifkan di target.
  - Untuk praktik terbaik keamanan, lihat SEC09-BP02 Menegakkan enkripsi saat transit.
- Pilih algoritma routing yang tepat (hanyaALB).
  - Algoritma perutean dapat membuat perbedaan tentang seberapa baik target backend Anda digunakan, oleh karena itu juga membuat perbedaan dalam dampaknya pada performa. Misalnya, ALB menyediakan dua opsi untuk algoritma routing:

  - Permintaan paling tidak menonjol: Gunakan untuk mendapatkan distribusi beban yang lebih baik ke target backend Anda untuk kasus ketika permintaan-permintaan untuk aplikasi Anda mempunayi tingkat kompleksitas yang berbeda-beda atau target Anda kemampuan pemrosesannya berbeda-beda.
  - Round robin: Gunakan ketika permintaan dan target serupa, atau jika Anda harus mendistribusikan permintaan secara sama rata di antara banyak target.
- Pertimbangkan isolasi zona atau lintas zona.
  - Gunakan penonaktifan lintas zona (isolasi zona) untuk meningkatkan latensi dan domain kegagalan zona. Ini dimatikan secara default di NLB dan di dalam ALBAnda dapat mematikannya per grup target.
  - Gunakan pengaktifan lintas zona untuk meningkatkan ketersediaan dan fleksibilitas. Secara default, lintas-zona dihidupkan ALB dan NLBAnda dapat menyalakannya per grup target.
- Aktifkan HTTP keep-alives untuk beban HTTP kerja Anda (hanya). ALB Dengan fitur ini, penyeimbang beban dapat menggunakan kembali koneksi backend hingga batas waktu keep-alive berakhir, meningkatkan HTTP permintaan dan waktu respons Anda dan juga mengurangi pemanfaatan sumber daya pada target backend Anda. Untuk detail tentang cara melakukan ini untuk Apache dan Nginx, lihat Untuk apa pengaturan optimal untuk menggunakan Apache atau NGINX sebagai server backend? ELB
- Aktifkan pemantauan untuk penyeimbang beban Anda.
  - Aktifkan log akses untuk Penyeimbang Beban Aplikasi dan Penyeimbang Beban Jaringan Anda.
  - Bidang utama yang perlu dipertimbangkan ALB adalah`request_processing_time`,`request_processing_time`, dan`response_processing_time`.
  - Bidang utama yang perlu dipertimbangkan NLB adalah `connection_time` dan`tls_handshake_time`.
  - Bersiaplah untuk melakukan kueri log ketika Anda memerlukannya. Anda dapat menggunakan Amazon Athena untuk menanyakan ALBlog dan NLB log.
  - Buat alarm untuk metrik terkait kinerja seperti `TargetResponseTime` untuk. ALB

Sumber daya

Dokumen terkait:

- ELBperbandingan produk

- AWS Infrastruktur Global
- Meningkatkan Performa dan Mengurangi Biaya Menggunakan Afinitas Zona Ketersediaan
- Langkah demi langkah untuk Analisis Log dengan Amazon Athena
- Kueri Log Penyeimbang Beban Aplikasi
- Memantau Penyeimbang Beban Aplikasi Anda
- Memantau Penyeimbang Beban Jaringan Anda
- Gunakan Penyeimbangan Beban Elastis Untuk mendistribusikan lalu lintas di seluruh instans dalam grup Auto Scaling Anda

Video terkait:

- AWS Re:invent 2023: Apa yang dapat dilakukan jaringan untuk aplikasi Anda?
- AWS RE: Inforce 20: Cara menggunakan Elastic Load Balancing untuk meningkatkan postur keamanan Anda dalam skala besar
- AWS RE: Invent 2018: Elastic Load Balancing: Deep Dive dan Praktik Terbaik
- AWS Re:invent 2021 - Bagaimana memilih penyeimbang beban yang tepat untuk beban kerja Anda AWS
- AWS Re: Invent 2019: Dapatkan hasil maksimal dari Elastic Load Balancing untuk beban kerja yang berbeda

Contoh terkait:

- Penyeimbang Beban Gateway
- CDKdan AWS CloudFormation sampel untuk Analisis Log dengan Amazon Athena

PERF04-BP05 Pilih protokol jaringan untuk meningkatkan kinerja

Buatlah keputusan terkait protokol untuk komunikasi antara sistem dan jaringan berdasarkan dampaknya terhadap kinerja beban kerja.

Ada hubungan antara latensi dan bandwidth untuk mencapai throughput. Jika transfer file Anda menggunakan Transmission Control Protocol (TCP), latensi yang lebih tinggi kemungkinan besar akan mengurangi keseluruhan throughput. Ada pendekatan untuk memperbaikinya dengan TCP tuning dan protokol transfer yang dioptimalkan, tetapi salah satu solusinya adalah dengan menggunakan User Datagram Protocol (). UDP

Anti-pola umum:

- Anda gunakan TCP untuk semua beban kerja terlepas dari persyaratan kinerja.

Manfaat menerapkan praktik terbaik ini: Memverifikasi bahwa sebuah protokol yang tepat telah digunakan untuk komunikasi antara pengguna dan bahwa komponen beban kerja akan membantu Anda dalam meningkatkan pengalaman pengguna secara keseluruhan untuk aplikasi Anda. Misalnya, tanpa koneksi UDP memungkinkan kecepatan tinggi, tetapi tidak menawarkan transmisi ulang atau keandalan tinggi. TCPadalah protokol berfitur lengkap, tetapi membutuhkan overhead yang lebih besar untuk memproses paket.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jika Anda memiliki kemampuan untuk memilih protokol yang berbeda-beda untuk aplikasi Anda dan Anda memiliki keahlian di bidang ini, optimalkan aplikasi dan pengalaman pengguna akhir Anda dengan menggunakan protokol yang berbeda-beda. Perlu diingat bahwa pendekatan ini memiliki tingkat kesulitan yang tinggi dan hanya boleh dicoba jika Anda telah mengoptimalkan aplikasi Anda dengan cara lain terlebih dahulu.

Pertimbangan utama dalam meningkatkan performa beban kerja Anda adalah pemahaman Anda terhadap persyaratan latensi dan throughput, dan kemudian pemilihan protokol jaringan yang mengoptimalkan performa.

Kapan harus mempertimbangkan untuk menggunakan TCP

TCPmenyediakan pengiriman data yang andal, dan dapat digunakan untuk komunikasi antara komponen beban kerja di mana keandalan dan pengiriman data yang terjamin penting. Banyak aplikasi berbasis web mengandalkan protokol TCP berbasis, seperti HTTP danHTTPS, untuk membuka TCP soket untuk komunikasi antar komponen aplikasi. Transfer data email dan file adalah aplikasi umum yang juga digunakanTCP, karena ini adalah mekanisme transfer yang sederhana dan andal antara komponen aplikasi. Menggunakan TLS dengan TCP dapat menambahkan beberapa overhead ke komunikasi, yang dapat mengakibatkan peningkatan latensi dan pengurangan throughput, tetapi dilengkapi dengan keuntungan keamanan. Overhead ini terutama berasal dari penambahan overhead untuk proses handshake, yang dapat memerlukan beberapa perjalanan pulang pergi agar selesai. Setelah handshake selesai, overhead enkripsi dan dekripsi data relatif kecil.

Kapan harus mempertimbangkan untuk menggunakan UDP

UDPadalah connection-less-oriented protokol dan oleh karena itu cocok untuk aplikasi yang membutuhkan transmisi cepat dan efisien, seperti log, pemantauan, dan data VoIP. Juga, pertimbangkan untuk menggunakan UDP jika Anda memiliki komponen beban kerja yang merespons kueri kecil dari sejumlah besar klien untuk memastikan kinerja beban kerja yang optimal. Datagram Transport Layer Security (DTLS) UDP setara dengan Transport Layer Security (TLS). Saat menggunakan DTLS withUDP, overhead berasal dari mengenkripsi dan mendekripsi data, karena proses jabat tangan disederhanakan. DTLSjuga menambahkan sejumlah kecil overhead ke UDP paket, karena mencakup bidang tambahan untuk menunjukkan parameter keamanan dan untuk mendeteksi gangguan.

Kapan harus mempertimbangkan untuk menggunakan SRD

Datagram andal yang dapat diskalakan (SRD) adalah protokol transportasi jaringan yang dioptimalkan untuk beban kerja throughput tinggi karena kemampuannya untuk memuat lalu lintas penyeimbang di beberapa jalur dan dengan cepat pulih dari penurunan paket atau kegagalan tautan. SRDOleh karena itu paling baik digunakan untuk beban kerja komputasi (HPC) kinerja tinggi yang memerlukan throughput tinggi dan komunikasi latensi rendah antara node komputasi. Hal ini dapat mencakup tugas pemrosesan paralel seperti simulasi, pemodelan, dan analisis data yang melibatkan banyak transfer data antara simpul.

Langkah-langkah implementasi

- Gunakan layanan AWS Global Accelerator dan AWS Transfer Family untuk memperbaiki throughput aplikasi transfer file online Anda. AWS Global Accelerator Layanan ini membantu Anda mencapai latensi yang lebih rendah antara perangkat klien Anda dan beban kerja Anda. AWS Dengan AWS Transfer Family, Anda dapat menggunakan protokol TCP berbasis seperti Secure Shell File Transfer Protocol (SFTP) dan File Transfer Protocol over SSL (FTPS) untuk menskalakan dan mengelola transfer file Anda ke AWS layanan penyimpanan dengan aman.
- Gunakan latensi jaringan untuk menentukan TCP apakah sesuai untuk komunikasi antar komponen beban kerja. Jika latensi jaringan antara aplikasi klien dan server Anda tinggi, maka jabat tangan TCP tiga arah dapat memakan waktu, sehingga berdampak pada respons aplikasi Anda. Metrik seperti time to first byte (TTFB) dan round-trip time (RTT) dapat digunakan untuk mengukur latensi jaringan. Jika beban kerja Anda menyajikan konten dinamis kepada pengguna, pertimbangkan untuk menggunakan Amazon CloudFront, yang membuat koneksi persisten ke setiap asal untuk konten dinamis guna menghapus waktu penyiapan koneksi yang akan memperlambat setiap permintaan klien.
- Menggunakan TLS dengan TCP atau UDP dapat mengakibatkan peningkatan latensi dan pengurangan throughput untuk beban kerja Anda karena dampak enkripsi dan dekripsi. Untuk

beban kerja seperti itu, pertimbangkanSSL/TLSpembongkaran pada Elastic Load Balancing untuk meningkatkan kinerja beban kerja dengan memungkinkan penyeimbang beban SSL menanganiTLS/proses enkripsi dan dekripsi alih-alih meminta instance backend melakukannya. Ini dapat membantu mengurangi CPU pemanfaatan pada instance backend, yang dapat meningkatkan kinerja dan meningkatkan kapasitas.

- Gunakan Network Load Balancer (NLB) untuk menyebarkan layanan yang bergantung pada UDP protokol, seperti otentikasi dan otorisasi, logging, DNS IoT, dan media streaming, untuk meningkatkan kinerja dan keandalan beban kerja Anda. Ini NLB mendistribusikan UDP lalu lintas masuk di beberapa target, memungkinkan Anda untuk menskalakan beban kerja Anda secara horizontal, meningkatkan kapasitas, dan mengurangi overhead satu target.
- Untuk beban kerja High Performance Computing (HPC) Anda, pertimbangkan untuk menggunakan fungsionalitas Elastic Network Adapter (ENA) Express yang menggunakan SRD protokol untuk meningkatkan kinerja jaringan dengan menyediakan bandwidth aliran tunggal yang lebih tinggi (25 Gbps) dan latensi ekor yang lebih rendah (99,9 persentil) untuk lalu lintas jaringan antar instance. EC2
- Gunakan Application Load Balancer (ALB) untuk merutekan dan memuat keseimbangan lalu lintas g RPC (Remote Procedure Calls) antara komponen beban kerja atau antara RPC klien g dan layanan. g RPC menggunakan protokol HTTP /2 TCP berbasis untuk transportasi dan memberikan manfaat kinerja seperti jejak jaringan yang lebih ringan, kompresi, serialisasi biner yang efisien, dukungan untuk berbagai bahasa, dan streaming dua arah.

Sumber daya

Dokumen terkait:

- Cara merutekan UDP lalu lintas ke Kubernetes
- Penyeimbang Beban Aplikasi
- EC2Jaringan yang disempurnakan di Linux
- EC2Jaringan yang Ditingkatkan di Windows
- EC2Grup Penempatan
- Mengaktifkan Jaringan yang Ditingkatkan dengan Adaptor Jaringan Elastis (ENA) di Instans Linux
- Penyeimbang Beban Jaringan
- Produk Networking dengan AWS
- Transisi ke Latensi Berbasis Perutean di Amazon Route 53

- VPCTitik akhir

Video terkait:

- AWS re:invent 2022 — Menskalakan kinerja jaringan pada instans Amazon Elastic Compute Cloud generasi berikutnya
- AWS Re: invent 2022 - Yayasan jaringan aplikasi

Contoh terkait:

- AWS Transit Gateway dan Solusi Keamanan yang Dapat Diskalakan
- Lokakarya Jaringan AWS

PERF04-BP06 Pilih lokasi beban kerja Anda berdasarkan persyaratan jaringan

Lakukan evaluasi terhada opsi-opsi untuk penempatan sumber daya guna mengurangi latensi jaringan dan meningkatkan throughput, yang akan memberikan pengalaman pengguna optimal dengan mengurangi beban halaman dan waktu transfer data.

Anti-pola umum:

- Anda menggabungkan semua sumber daya beban kerja ke dalam satu lokasi geografis.
- Anda memilih Wilayah terdekat dengan lokasi Anda tetapi tidak dekat dengan pengguna akhir beban kerja.

Manfaat menerapkan praktik terbaik ini: Pengalaman pengguna sangat mereka sangat terpengaruh oleh latensi yang ada antara pengguna dan aplikasi Anda. Dengan menggunakan jaringan global yang sesuai Wilayah AWS dan AWS pribadi, Anda dapat mengurangi latensi dan memberikan pengalaman yang lebih baik kepada pengguna jarak jauh.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sumber daya, seperti EC2 instans Amazon, ditempatkan ke Availability Zone di dalam Wilayah AWS, AWS Local Zones AWS Outposts, atau AWS Wavelengthzona. Pemilihan lokasi ini akan memengaruhi latensi jaringan dan throughput dari lokasi pengguna tertentu. Layanan Edge seperti Amazon CloudFront dan juga AWS Global Acceleratordapat digunakan untuk meningkatkan kinerja

jaringan dengan menyimpan konten di lokasi tepi atau memberi pengguna jalur optimal ke beban kerja melalui jaringan AWS global.

Amazon EC2 menyediakan grup penempatan untuk jaringan. Grup penempatan adalah sebuah pengelompokan logis instans untuk mengurangi latensi. Menggunakan grup penempatan dengan tipe instans yang didukung dan Elastic Network Adapter (ENA) memungkinkan beban kerja untuk berpartisipasi dalam jaringan jitter 25 Gbps dengan latensi rendah dan dikurangi. Grup penempatan direkomendasikan untuk beban kerja yang memanfaatkan latensi jaringan yang rendah, throughput jaringan yang tinggi, atau keduanya.

Layanan yang sensitif terhadap latensi dikirimkan di lokasi tepi menggunakan jaringan AWS global, seperti Amazon. CloudFront Lokasi tepi ini biasanya menyediakan layanan seperti jaringan pengiriman konten (CDN) dan sistem nama domain (DNS). Dengan memiliki layanan ini di tepi, beban kerja dapat merespons dengan latensi rendah untuk permintaan konten atau DNS resolusi. Layanan-layanan ini juga menyediakan layanan geografis seperti penargetan geografis konten (menyediakan konten yang berbeda berdasarkan lokasi pengguna akhir) atau perutean berbasis latensi untuk mengarahkan para pengguna akhir ke Wilayah terdekat (latensi minimum).

Gunakan layanan-layanan edge untuk mengurangi latensi dan memungkinkan caching konten. Konfigurasikan kontrol cache dengan benar untuk keduanya DNS danHTTP/HTTPSuntuk mendapatkan manfaat maksimal dari pendekatan ini.

Langkah-langkah implementasi

- Rekam informasi tentang lalu lintas IP ke dan dari antarmuka jaringan.
  - Mencatat lalu lintas IP menggunakan VPC Flow Logs
  - Bagaimana alamat IP klien dipertahankan di AWS Global Accelerator
- Lakukan analisis terhadap pola akses jaringan di beban kerja Anda untuk mengidentifikasi cara pengguna menggunakan aplikasi Anda.
  - Gunakan alat pemantauan, seperti Amazon CloudWatch dan AWS CloudTrail, untuk mengumpulkan data tentang aktivitas jaringan.
  - Analisis data untuk mengidentifikasi pola akses jaringan.
- Pilih Wilayah untuk deployment beban kerja Anda berdasarkan elemen-elemen utama berikut:
  - Dimana lokasi data Anda: Untuk aplikasi-aplikasi dengan banyak data (seperti big data dan machine learning), kode aplikasi harus dijalankan sedekat mungkin dengan data.
  - Dimana lokasi pengguna Anda: Untuk aplikasi-aplikasi yang ditampilkan kepada pengguna, pilihlah sebuah Wilayah (Wilayah-wilayah) yang dekat dengan para pengguna beban kerja Anda.

  - Kendala lainnya: Pertimbangkan kendala-kendala seperti biaya dan kepatuhan sebagaimana yang dijelaskan dalam Hal-Hal yang Perlu Dipertimbangkan saat Memilih Wilayah untuk Beban Kerja Anda.
- Gunakan Zona Lokal AWS untuk menjalankan beban kerja seperti rendering video. Zona Lokal memungkinkan Anda untuk mendapatkan semua manfaat dari komputasi dan sumber daya penyimpanan yang lebih dekat dengan para pengguna akhir.
- Gunakan AWS Outposts untuk beban kerja yang harus tetap berada on-premise dan di tempat Anda ingin beban kerja tersebut berfungsi dengan lancar bersama dengan beban kerja Anda yang lain yang ada di AWS.
- Aplikasi seperti streaming video langsung resolusi tinggi, audio kesetiaan tinggi, dan augmented reality atau virtual reality (AR/VR) memerlukan perangkat 5G. ultra-low-latency Untuk aplikasi semacam itu, pertimbangkan AWS Wavelength. AWS Wavelength AWS menyematkan layanan komputasi dan penyimpanan dalam jaringan 5G, menyediakan infrastruktur komputasi tepi seluler untuk mengembangkan, menyebarkan, dan menskalakan aplikasi. ultra-low-latency
- Gunakan caching lokal atau Solusi Penerapan Cache AWS untuk aset-aset yang sering digunakan untuk meningkatkan performa, mengurangi perpindahan data, dan mengurangi dampak pada lingkungan.

| Layanan | Kapan harus digunakan |
| --- | --- |
| Amazon CloudFront | Gunakan untuk menyimpan konten statis seperti gambar, skrip, dan video, serta konten dinamis seperti API respons atau aplikasi web. |
| Amazon ElastiCache | Gunakan untuk meng-cache konten bagi aplikasi web. |
| DynamoDB Accelerator | Gunakan untuk menambahkan percepatan dalam memori ke tabel DynamoDB Anda. |

- Gunakan layanan-layanan yang dapat membantu Anda menjalankan kode lebih dekat dengan pengguna beban kerja Anda seperti berikut:

| Layanan | Kapan harus digunakan |
| --- | --- |
| Lambda@Edge | Gunakan untuk operasi-operasi yang memiliki banyak komputasi yang dimulai saat objek tidak ada dalam cache. |
| CloudFront Fungsi Amazon | Gunakan untuk kasus penggunaan sederhana seperti HTTP permintaan atau manipulasi respons yang dapat dimulai oleh fungsi berumur pendek. |
| AWS IoT Greengrass | Gunakan untuk menjalankan komputasi lokal, olahpesan, dan caching data untuk perangkat yang terhubung. |

- Beberapa aplikasi memerlukan titik masuk tetap atau performa yang lebih tinggi dengan mengurangi jitter dan latensi bita pertama, dan meningkatkan throughput. Aplikasi ini dapat mengambil manfaat dari layanan jaringan yang menyediakan alamat IP anycast statis dan TCP penghentian di lokasi tepi. AWS Global Acceleratordapat meningkatkan kinerja untuk aplikasi Anda hingga 60% dan memberikan failover cepat untuk arsitektur multi-wilayah. AWS Global Accelerator memberi Anda alamat IP anycast statis yang berfungsi sebagai titik masuk tetap untuk aplikasi Anda yang dihosting dalam satu atau lebih Wilayah AWS. Alamat IP ini memungkinkan lalu lintas masuk ke jaringan AWS global sedekat mungkin dengan pengguna Anda. AWS Global Accelerator mengurangi waktu penyiapan koneksi awal dengan membuat TCP koneksi antara klien dan lokasi AWS tepi yang paling dekat dengan klien. Tinjau penggunaan AWS Global Accelerator untuk meningkatkan kinerja TCP UDP /beban kerja Anda dan menyediakan failover cepat untuk arsitektur Multi-region.

Sumber daya

Praktik-praktik terbaik terkait:

- COST07-BP02 Melaksanakan Daerah berdasarkan biaya
- COST08-BP03 Menerapkan layanan untuk mengurangi biaya transfer data
- REL10-BP01 Menyebarkan beban kerja ke beberapa lokasi
- REL10-BP02 Pilih lokasi yang sesuai untuk penyebaran multi-lokasi Anda

- SUS01-BP01 Pilih Wilayah berdasarkan persyaratan bisnis dan tujuan keberlanjutan
- SUS02-BP04 Optimalkan penempatan geografis beban kerja berdasarkan persyaratan jaringan mereka
- SUS04-BP07 Meminimalkan pergerakan data di seluruh jaringan

Dokumen terkait:

- AWS Infrastruktur Global
- AWS Local Zones dan AWS Outposts, memilih teknologi yang tepat untuk beban kerja edge Anda
- Grup penempatan
- AWS Local Zones
- AWS Outposts
- AWS Wavelength
- Amazon CloudFront
- AWS Global Accelerator
- AWS Direct Connect
- AWS Site-to-Site VPN
- Amazon Route 53

Video terkait:

- AWS Video Penjelasan Local Zones
- AWS Outposts: Ikhtisar dan Cara Kerjanya
- AWS re:invent 2023 - Strategi migrasi untuk beban kerja edge dan lokal
- AWS Re:invent 2021 - AWS Outposts: Membawa pengalaman di tempat AWS
- AWS re:invent 2020: AWS Wavelength: Jalankan aplikasi dengan latensi ultra-rendah di tepi 5G
- AWS re:invent 2022 - AWS Local Zones: Membangun aplikasi untuk tepi terdistribusi
- AWS re:invent 2021 - Membangun situs web latensi rendah dengan Amazon CloudFront
- AWS re:invent 2022 - Tingkatkan kinerja dan ketersediaan dengan AWS Global Accelerator
- AWS re:invent 2022 - Bangun jaringan area luas global Anda menggunakan AWS
- AWS re: invent 2020: Manajemen lalu lintas global dengan Amazon Route 53

Contoh terkait:

- AWS Global Accelerator Lokakarya Perutean Kustom
- Menangani Penulisan Ulang dan Pengarahan Ulang dengan menggunakan Fungsi Edge

PERF04-BP07 Optimalkan konfigurasi jaringan berdasarkan metrik

Gunakan data yang telah terkumpul dan dianalisis untuk mengambil keputusan yang tepat terkait pengoptimalan konfigurasi jaringan Anda.

Anti-pola umum:

- Anda beranggapan bahwa semua masalah yang berkaitan dengan kinerja disebabkan oleh aplikasi.
- Anda hanya menguji performa jaringan dari sebuah lokasi yang dekat dari tempat deployment beban kerja.
- Anda menggunakan konfigurasi default untuk semua layanan jaringan.
- Anda menyediakan terlalu banyak sumber daya jaringan untuk memberikan kapasitas yang memadai.

Manfaat menerapkan praktik terbaik ini: Dengan mengumpulkan metrik jaringan AWS yang diperlukan dan mengimplementasikan alat pemantauan jaringan, Anda dapat memahami performa jaringan dan mengoptimalkan konfigurasi jaringan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Memantau lalu lintas ke dan dariVPCs, subnet, atau antarmuka jaringan sangat penting untuk memahami cara memanfaatkan sumber daya AWS jaringan dan mengoptimalkan konfigurasi jaringan. Dengan menggunakan alat AWS jaringan berikut, Anda dapat memeriksa lebih lanjut informasi tentang penggunaan lalu lintas, akses jaringan, dan log.

Langkah-langkah implementasi

- Identifikasi metrik kinerja utama seperti latensi atau kehilangan paket untuk dikumpulkan. AWS menyediakan beberapa alat yang dapat membantu Anda mengumpulkan metrik ini. Dengan menggunakan alat-alat berikut, Anda dapat melakukan memeriksa informasi lebih lanjut tentang penggunaan lalu lintas, akses jaringan, dan log:

| AWS alat | Harus digunakan di mana |
| --- | --- |
| Manajer Alamat VPC IP Amazon. | Gunakan IPAM untuk merencanakan, melacak, dan memantau alamat IP untuk beban kerja Anda AWS dan lokal. Ini adalah praktik terbaik yang bisa digunakan untuk mengoptimalkan alokasi dan penggunaan alamat IP. |
| VPCLog aliran | Gunakan VPC Flow Logs untuk menangkap informasi terperinci tentang lalu lintas ke dan dari antarmuka jaringan di AndaVPCs. Dengan VPC Flow Logs, Anda dapat mendiagnosis aturan grup keamanan yang terlalu ketat atau permisif dan menentuka n arah lalu lintas ke dan dari antarmuka jaringan. |
| AWS Transit Gateway Log Aliran | Gunakan AWS Transit Gateway Flow Logs untuk menangkap informasi tentang lalu lintas IP yang menuju dan dari gateway transit Anda. |
| DNSpencatatan kueri | Informasi log tentang DNS pertanyaan publik atau pribadi yang diterima Route 53. Dengan DNS log, Anda dapat mengoptimalkan DNS konfigurasi dengan memahami domain atau subdomain yang diminta atau EDGE lokasi Route 53 yang merespons kueri. DNS |

| AWS alat | Harus digunakan di mana |
| --- | --- |
| Reachability Analyzer | Reachability Analyzer akan membantu Anda menganalisis dan men-debug jangkauan jaringan. Reachability Analyzer adalah alat analisis konfigurasi yang memungkinkan Anda melakukan pengujian konektivitas antara sumber daya sumber dan sumber daya tujuan di sumber daya Anda. VPCs Alat ini dapat membantu Anda memverifikasi bahwa konfigurasi jaringan Anda sesuai dengan konektivitas yang ditarget. |
| Penganalisis Akses Jaringan | Penganalisis Akses Jaringan akan membantu Anda memahami akses jaringan ke sumber daya Anda. Anda dapat menggunak an Penganalisis Akses Jaringan untuk menentukan persyaratan-persyaratan akses jaringan Anda serta mengidentifikasi jalur jaringan yang berpotensi tidak memenuhi persyaratan yang Anda tentukan. Dengan mengoptimalkan konfigurasi jaringan Anda yang bersangkutan, Anda dapat memahami dan memverifikasi status jaringan Anda dan menunjukkan apakah jaringan Anda yang ada di AWS memenuhi persyaratan kepatuhan Anda. |

| AWS alat | Harus digunakan di mana |
| --- | --- |
| Amazon CloudWatch | Gunakan Amazon CloudWatch dan aktifkan metrik yang sesuai untuk opsi jaringan. Pastikan Anda memilih metrik jaringan yang tepat untuk beban kerja Anda. Misalnya, Anda dapat mengaktifkan metrik untuk Penggunaan Alamat VPC Jaringan, VPC NAT Gateway, VPN terowongan, AWS Transit Gateway, Elastic Load Balancing AWS Network Firewall, dan. AWS Direct Connect Melakukan pemantauan metrik secara terus-menerus merupakan praktik yang bagus untuk mengamati dan memahami penggunaan dan status jaringan Anda, yang membantu Anda mengoptimalkan konfigurasi jaringan berdasarkan pengamatan Anda. |
| AWS Network Manager | Dengan menggunakan AWS Network Manager, Anda dapat memantau kinerja real-time dan historis Jaringan AWS Global untuk tujuan operasional dan perencana an. Network Manager menyediakan latensi jaringan agregat antara Wilayah AWS dan Availability Zones dan dalam setiap Availabil ity Zone, memungkinkan Anda untuk lebih memahami bagaimana kinerja aplikasi Anda berhubungan dengan kinerja jaringan yang mendasarinya. AWS |
| Amazon CloudWatch RUM | Gunakan Amazon CloudWatch RUM untuk mengumpulkan metrik yang memberi Anda wawasan yang membantu Anda mengident ifikasi, memahami, dan meningkatkan pengalaman pengguna. |

- Identifikasi pembicara teratas dan pola lalu lintas aplikasi menggunakan VPC dan AWS Transit Gateway Flow Logs.
- Menilai dan mengoptimalkan arsitektur jaringan Anda saat ini termasukVPCs, subnet, dan routing. Sebagai contoh, Anda dapat mengevaluasi seberapa berbeda VPC peering atau AWS Transit Gateway dapat membantu Anda meningkatkan jaringan dalam arsitektur Anda.
- Nilai jalur perutean di jaringan Anda untuk memastikan digunakannya jalur terpendek antartujuan. Penganalisis Akses Jaringan dapat membantu Anda melakukan ini.

Sumber daya

Dokumen terkait:

- Pencatatan DNS kueri publik
- Apa ituIPAM?
- Apa itu Reachability Analyzer?
- Apa itu Penganalisis Akses Jaringan?
- CloudWatchmetrik untuk Anda VPCs
- Optimalkan kinerja dan kurangi biaya untuk analitik jaringan dengan VPC Flow Logs dalam format Apache Parquet
- Memantau jaringan global dan inti Anda dengan CloudWatch metrik Amazon
- Memantau sumber daya dan lalu lintas jaringan terus-menerus

Video terkait:

- AWS re:Invent 2023 - Panduan pengembang untuk jaringan cloud
- AWS Re: invent 2023 - Siap untuk apa selanjutnya? Merancang jaringan untuk pertumbuhan dan fleksibilitas
- AWS RE: invent 2023 - Desain canggih VPC dan kemampuan baru
- AWS re:invent 2022 — Menyelam jauh pada infrastruktur jaringan AWS
- AWS RE: Invent 2020 - Melakukan praktik dan kiat terbaik jaringan dengan Well-Architected Framework AWS
- AWS Re:invent 2020 - Pemantauan dan pemecahan masalah lalu lintas jaringan

Contoh terkait:

- Lokakarya Jaringan AWS
- Pemantauan Jaringan AWS
- Mengamati dan mendiagnosis jaringan Anda AWS
- Menemukan dan menangani kesalahan konfigurasi jaringan pada AWS

## Proses dan budaya

Pertanyaan

- PERF5. Bagaimana praktik dan budaya organisasi Anda berkontribusi pada efisiensi performa dalam beban kerja Anda?

### PERF5. Bagaimana praktik dan budaya organisasi Anda berkontribusi pada efisiensi performa dalam beban kerja Anda?

Saat merancang beban kerja, ada prinsip dan praktik yang dapat Anda adopsi untuk membantu Anda menjalankan beban kerja cloud berkinerja tinggi yang efisien dengan lebih baik. Untuk mengadopsi budaya yang mendorong efisiensi kinerja beban kerja cloud, pertimbangkan prinsip dan praktik utama ini:

Praktik terbaik

- PERF05-BP01 Menetapkan indikator kinerja utama (KPIs) untuk mengukur kesehatan dan kinerja beban kerja
- PERF05-BP02 Gunakan solusi pemantauan untuk memahami area di mana kinerja paling penting
- PERF05-BP03 Menentukan proses untuk meningkatkan kinerja beban kerja
- PERF05-BP04 Uji beban kerja Anda
- PERF05-BP05 Gunakan otomatisasi untuk secara proaktif memulihkan masalah terkait kinerja
- PERF05-BP06 Pertahankan beban kerja dan layanan Anda up-to-date
- PERF05-BP07 Tinjau metrik secara berkala

#### PERF05-BP01 Menetapkan indikator kinerja utama (KPIs) untuk mengukur kesehatan dan kinerja beban kerja

Identifikasi KPIs yang mengukur kinerja beban kerja secara kuantitatif dan kualitatif. KPIsmembantu Anda mengukur kesehatan dan kinerja beban kerja yang terkait dengan tujuan bisnis.

Anti-pola umum:

- Anda hanya memantau metrik tingkat sistem untuk memperoleh wawasan tentang beban kerja Anda dan tidak memahami dampak bisnis yang diakibatkan pada metrik-metrik tersebut.
- Anda berasumsi bahwa Anda sudah KPIs dipublikasikan dan dibagikan sebagai data metrik standar.
- Anda tidak mendefinisikan kuantitatif, terukurKPI.
- Anda tidak selaras KPIs dengan tujuan atau strategi bisnis.

Manfaat membangun praktik terbaik ini: Mengidentifikasi spesifik KPIs yang mewakili kesehatan dan kinerja beban kerja membantu menyelaraskan tim pada prioritas mereka dan menentukan hasil bisnis yang sukses. Ketika metrik-metrik tersebut dibagikan kepada semua departemen, akan ada visibilitas dan kesepakatan tentang ambang batas, harapan, dan dampak bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

KPIsmemungkinkan tim bisnis dan teknik untuk menyelaraskan pengukuran tujuan dan strategi dan bagaimana faktor-faktor ini bergabung untuk menghasilkan hasil bisnis. Misalnya, beban kerja situs web mungkin menggunakan waktu muat halaman sebagai sebuah indikasi kinerja secara keseluruhan. Metrik ini adalah salah satu dari beberapa poin data yang mengukur pengalaman pengguna. Selain mengidentifikasi ambang batas waktu muat halaman, Anda juga harus mendokumentasikan hasil yang diharapkan atau risiko bisnis yang diperkirakan jika kinerja ideal tidak dipenuhi. Waktu muat halaman yang lama memengaruhi pengguna akhir Anda secara langsung, mengurangi tingkat pengalaman pengguna mereka, dan dapat menyebabkan hilangnya pelanggan. Saat Anda menentukan KPI ambang batas Anda, gabungkan tolok ukur industri dan harapan pengguna akhir Anda. Misalnya, jika tolok ukur industri saat ini adalah pemuatan halaman web dalam periode waktu dua detik, tetapi pengguna akhir Anda mengharapkan halaman web dimuat dalam periode waktu satu detik, maka Anda harus mempertimbangkan kedua titik data ini saat menetapkan. KPI

Tim Anda harus mengevaluasi beban kerja Anda KPIs menggunakan data granular real-time dan data historis untuk referensi dan membuat dasbor yang melakukan matematika metrik pada KPI data Anda untuk memperoleh wawasan operasional dan pemanfaatan. KPIsharus didokumentasikan dan mencakup ambang batas yang mendukung tujuan dan strategi bisnis, dan harus dipetakan ke metrik yang dipantau. KPIsharus ditinjau kembali ketika tujuan bisnis, strategi, atau persyaratan pengguna akhir berubah.

Langkah-langkah implementasi

- Identifikasi pemangku kepentingan: Mengidentifikasi dan mendokumentasikan para pemangku kepentingan bisnis utama, termasuk tim pengembangan dan operasi.
- Tentukan tujuan: Bekerjalah dengan para pemangku kepentingan ini untuk menentukan dan mendokumentasikan tujuan beban kerja Anda. Pertimbangkan aspek-aspek kinerja penting dari beban kerja Anda, seperti throughput, waktu respons, dan biaya, serta tujuan bisnis, seperti kepuasan pengguna.
- Tinjau praktik terbaik industri: Tinjau praktik terbaik industri untuk mengidentifikasi relevan yang KPIs selaras dengan tujuan beban kerja Anda.
- Identifikasi metrik: Identifikasi metrik-metrik yang selaras dengan sasaran beban kerja Anda dan dapat membantu Anda mengukur kinerja dan tujuan-tujuan bisnis. Tetapkan KPIs berdasarkan metrik ini. Contoh metrik adalah pengukuran seperti waktu respons rata-rata atau jumlah pengguna serentak.
- Tentukan dan dokumentasikanKPIs: Gunakan praktik terbaik industri dan sasaran beban kerja Anda untuk menetapkan target beban kerja KPI Anda. Gunakan informasi ini untuk menetapkan KPI ambang batas tingkat keparahan atau alarm. Mengidentifikasi dan mendokumentasikan risiko dan dampak yang KPI tidak terpenuhi.
- Menerapkan pemantauan: Gunakan alat pemantauan seperti Amazon CloudWatch atau AWS Configuntuk mengumpulkan metrik dan pengukuranKPIs.
- Berkomunikasi secara visualKPIs: Gunakan alat dasbor seperti Amazon QuickSight untuk memvisualisasikan dan berkomunikasi KPIs dengan pemangku kepentingan.
- Analisis dan optimalkan: Tinjau dan analisis secara teratur KPIs untuk mengidentifikasi area beban kerja Anda yang perlu ditingkatkan. Bekerjalah dengan para pemangku kepentingan untuk mengimplementasikan perbaikan-perbaikan tersebut.
- Kunjungi kembali dan perbaiki: Tinjau metrik secara teratur dan KPIs untuk menilai efektivitasnya, terutama ketika tujuan bisnis atau kinerja beban kerja berubah.

Sumber daya

Dokumen terkait:

- CloudWatchdokumentasi
- Pemantauan, Pencatatan, dan Kinerja AWS Partner s
- AWS alat observabilitas

- Pentingnya Indikator Kinerja Utama (KPIs) untuk Migrasi Cloud Skala Besar
- Cara melacak pengoptimalan biaya Anda KPIs dengan KPI Dasbor
- Dokumentasi X-Ray
- Menggunakan CloudWatch dasbor Amazon
- Amazon QuickSight KPIs

Video terkait:

- AWS re: invent 2023 - Optimalkan biaya dan kinerja dan lacak kemajuan menuju mitigasi
- AWS re:invent 2023 - Kelola acara siklus hidup sumber daya dalam skala besar dengan AWS Health
- AWS re:invent 2023 - Kinerja & efisiensi di Pinterest: Mengoptimalkan instans terbaru
- AWS re:invent 2022 - AWS optimasi: Langkah-langkah yang dapat ditindaklanjuti untuk hasil langsung
- AWS re:invent 2023 - Membangun strategi observabilitas yang efektif
- AWS Summit SF 2022 - Observabilitas tumpukan penuh dan pemantauan aplikasi dengan AWS
- AWS Re:invent 2023 - Menskalakan AWS untuk 10 juta pengguna pertama
- AWS re:invent 2022 - Bagaimana Amazon menggunakan metrik yang lebih baik untuk meningkatkan kinerja situs web
- Membuat Strategi Metrik yang Efektif untuk Bisnis Anda | Acara AWS

Contoh terkait:

- Membuat dasbor dengan Amazon QuickSight

PERF05-BP02 Gunakan solusi pemantauan untuk memahami area di mana kinerja paling penting

Pahami dan identifikasi area di mana peningkatan kinerja beban kerja akan memiliki dampak positif pada efisiensi atau pengalaman pelanggan. Contohnya, situs web yang memiliki banyak interaksi pelanggan dapat diuntungkan oleh penggunaan layanan edge untuk membuat penyampaian konten ke pelanggan menjadi lebih dekat.

Anti-pola umum:

- Anda berasumsi bahwa metrik komputasi standar seperti CPU pemanfaatan atau tekanan memori sudah cukup untuk menangkap masalah kinerja.
- Anda hanya menggunakan metrik-metrik default yang dicatat oleh perangkat lunak pemantauan Anda yang dipilih.
- Anda hanya meninjau metrik-metrik tersebut ketika terdapat masalah.

Manfaat membangun praktik terbaik ini: Memahami area kinerja yang kritis membantu pemilik beban kerja memantau KPIs dan memprioritaskan peningkatan berdampak tinggi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Siapkan end-to-end penelusuran untuk mengidentifikasi pola lalu lintas, latensi, dan area kinerja kritis. Pantau pola akses data Anda untuk mencari kueri yang lambat atau data dengan fragmentasi dan partisi yang buruk. Identifikasi area-area beban kerja terbatas dengan menggunakan pengujian atau pemantauan beban.

Tingkatkan efisiensi kinerja dengan memahami arsitektur, pola lalu lintas, dan pola akses data Anda, serta lakukan identifikasi latensi dan waktu pemrosesan Anda. Lakukan juga identifikasi terhadap potensi hambatan yang bisa memengaruhi pengalaman pelanggan selama beban kerja berkembang. Setelah menginvestigasi area-area tersebut, lihat solusi mana yang dapat Anda deploy untuk menghilangkan masalah-masalah kinerja tersebut.

Langkah-langkah implementasi

- Siapkan end-to-end pemantauan untuk menangkap semua komponen dan metrik beban kerja. Berikut adalah contoh solusi pemantauan pada AWS.

| Layanan | Harus digunakan di mana |
| --- | --- |
| Pemantauan CloudWatch Pengguna Nyata Amazon () RUM | Untuk merekam metrik-metrik performa aplikasi dari sesi sisi klien dan frontend pengguna nyata. |
| AWS X-Ray | Untuk melacak lalu lintas melalui lapisan-l apisan aplikasi dan mengidentifikasi latensi yang ada antara komponen dan dependens i. Gunakan peta layanan X-Ray untuk melihat |

| Layanan | Harus digunakan di mana |
| --- | --- |
| | hubungan dan latensi yang ada antara komponen beban kerja. |
| Wawasan Performa Layanan Basis Data Relasional Amazon | Untuk melihat metrik-metrik kinerja basis data dan mengidentifikasi peningkatan kinerja. |
| Pemantauan Amazon RDS yang Ditingkatkan | Untuk melihat metrik-metrik kinerja OS basis data. |
| DevOpsGuru Amazon | Untuk mendeteksi pola operasi yang tidak normal sehingga Anda dapat mengidentifikasi setiap masalah operasional sebelum masalah tersebut berdampak pada para pelanggan Anda. |

- Lakukan pengujian untuk membuat metrik, mengidentifikasi pola lalu lintas, hambatan, dan mengidentifikasi area-area kinerja kritis. Berikut adalah beberapa contoh cara melakukan pengujian:
  - Siapkan Canaries CloudWatch Sintetis untuk meniru aktivitas pengguna berbasis browser secara terprogram menggunakan pekerjaan cron Linux atau ekspresi tingkat untuk menghasilkan metrik yang konsisten dari waktu ke waktu.
  - Gunakan solusi Pengujian Beban Terdistribusi AWS untuk menghasilkan lalu lintas puncak atau menguji beban kerja pada tingkat pertumbuhan yang diharapkan.
- Evaluasi metrik dan telemetri untuk mengidentifikasi area-area kinerja kritis Anda. Tinjau area-area ini bersama dengan tim Anda untuk mendiskusikan pemantauan dan solusi untuk menghindari hambatan.
- Lakukan eksperimen dengan peningkatan kinerja serta ukur perubahannya dengan data. Sebagai contoh, Anda dapat menggunakan CloudWatchEvidently untuk menguji peningkatan baru dan dampak kinerja terhadap beban kerja Anda.

Sumber daya

Dokumen terkait:

- Apa yang baru di AWS Observability at re:Invent 2023
- Amazon Builders' Library

- Dokumentasi X-Ray
- Amazon CloudWatch RUM
- DevOpsGuru Amazon

Video terkait:

- AWS RE: invent 2023 - [LAUNCH] Pemantauan aplikasi untuk beban kerja modern
- AWS re: invent 2023 - Menerapkan observabilitas aplikasi
- AWS re:invent 2023 - Membangun strategi observabilitas yang efektif
- AWS Summit SF 2022 - Observabilitas tumpukan penuh dan pemantauan aplikasi dengan AWS
- AWS re:invent 2022 - AWS optimasi: Langkah-langkah yang dapat ditindaklanjuti untuk hasil langsung
- AWS re:invent 2022 - Perpustakaan Amazon Builders: 25 tahun keunggulan operasional Amazon
- AWS re:invent 2022 - Bagaimana Amazon menggunakan metrik yang lebih baik untuk meningkatkan kinerja situs web
- Pemantauan Visual Aplikasi dengan Amazon CloudWatch Synthetics

Contoh terkait:

- Ukur waktu buka halaman dengan Amazon CloudWatch Synthetics
- Klien CloudWatch RUM Web Amazon
- X-Ray SDK untuk Python
- Pengujian Beban Terdistribusi pada AWS

PERF05-BP03 Menentukan proses untuk meningkatkan kinerja beban kerja

Menetapkan sebuah proses untuk mengevaluasi layanan, pola desain, tipe sumber daya, dan konfigurasi baru saat sudah tersedia. Misalnya, jalankan pengujian kinerja yang sudah ada pada penawaran instans baru untuk menentukan potensinya untuk beban kerja Anda.

Anti-pola umum:

- Anda berasumsi bahwa arsitektur Anda saat ini statis dan tidak akan diperbarui dari waktu ke waktu.
- Anda memperkenalkan metrik arsitektur seiring waktu tanpa justifikasi metrik.

Manfaat menerapkan praktik terbaik ini: Setelah proses untuk membuat perubahan arsitektur ditetapkan, Anda dapat menggunakan data yang dikumpulkan untuk memengaruhi desain beban kerja Anda seiring waktu.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Kinerja beban kerja Anda memiliki beberapa kendala utama. Dokumentasikan kendala-kendala tersebut untuk mengetahui jenis inovasi apa saja yang mungkin dapat meningkatkan kinerja beban kerja Anda. Gunakan informasi ini ketika mempelajari layanan atau teknologi baru ketika sudah tersedia untuk mengidentifikasi cara-cara yang bisa digunakan untuk menghilangkan kendala atau bottleneck.

Identifikasi kendala kinerja utama untuk beban kerja Anda. Dokumentasikan kendala-kendala performa beban kerja Anda sehingga Anda tahu jenis-jenis inovasi apa yang dapat meningkatkan performa beban kerja Anda.

Langkah-langkah implementasi

- IdentifikasiKPIs: Identifikasi kinerja beban kerja Anda KPIs sebagaimana diuraikan PERF05-BP01 Menetapkan indikator kinerja utama (KPIs) untuk mengukur kesehatan dan kinerja beban kerja untuk menjadi dasar beban kerja Anda.
- Terapkan pemantauan: Gunakan alat AWS observabilitas untuk mengumpulkan metrik dan pengukuran kinerja. KPIs
- Lakukan analisis: Lakukan analisis mendalam untuk mengidentifikasi area-area (seperti konfigurasi dan kode aplikasi) di dalam beban kerja Anda yang berkinerja buruk seperti yang diuraikan dalam PERF05-BP02 Gunakan solusi pemantauan untuk memahami area di mana kinerja paling penting. Gunakan alat-alat analisis dan kinerja Anda untuk mengidentifikasi strategi perbaikan kinerja.
- Validasi perbaikan: Gunakan sandbox atau lingkungan pra-produksi untuk memvalidasi efektivitas strategi perbaikan.
- Terapkan perubahan: Implementasikan perubahan-perubahan dalam lingkungan produksi dan terus pantau kinerja beban kerja. Dokumentasikan perbaikan, dan komunikasikan perubahan-perubahan kepada para pemangku kepentingan.
- Pertahankan dan perbaiki: Tinjau proses peningkatan kinerja Anda secara rutin untuk mengidentifikasi area-area yang bisa ditingkatkan lagi.

Sumber daya

Dokumen terkait:

- Blog AWS
- Apa yang baru dengan AWS
- AWS Pembangun Keterampilan

Video terkait:

- AWS re:invent 2022 - Memberikan arsitektur yang berkelanjutan dan berkinerja tinggi
- AWS re:invent 2023 - Optimalkan biaya dan kinerja dan lacak kemajuan menuju mitigasi
- AWS re:invent 2022 - AWS optimasi: Langkah-langkah yang dapat ditindaklanjuti untuk hasil langsung
- AWS Re:invent 2022 - Optimalkan beban AWS kerja Anda dengan panduan praktik terbaik

Contoh terkait:

- AWS Github

PERF05-BP04 Uji beban kerja Anda

Uji beban untuk beban kerja Anda untuk memverifikasi bahwa beban kerja Anda dapat menangani beban produksi dan mengidentifikasi kemacetan kinerja apa pun.

Anti-pola umum:

- Anda melakukan uji beban bagian beban kerja secara terpisah-pisah, bukan seluruh beban kerja.
- Anda melakukan uji beban pada infrastruktur yang tidak sama dengan lingkungan produksi Anda.
- Anda hanya melakukan pengujian beban pada beban yang diharapkan, tidak lebih, untuk membantu Anda memperkirakan area-area yang mungkin akan bermasalah di masa depan.
- Anda melakukan pengujian beban tanpa berkonsultasi dengan Kebijakan EC2 Pengujian Amazon dan mengirimkan Formulir Pengiriman Acara Simulasi. Ini mengakibatkan pengujian Anda gagal dijalankan, karena terlihat seperti denial-of-service acara.

Manfaat menerapkan praktik terbaik ini: Mengukur kinerja Anda dalam sebuah uji beban akan menunjukkan di mana Anda akan terdampak saat terjadi peningkatan beban. Hal ini bisa memberi Anda kemampuan untuk mengantisipasi perubahan yang diperlukan sebelum perubahan tersebut berdampak pada beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Pengujian beban di cloud adalah sebuah proses untuk mengukur kinerja beban kerja cloud dalam kondisi realistis dengan beban pengguna yang diharapkan. Proses ini melibatkan penyediaan lingkungan cloud yang mirip lingkungan produksi, penggunaan alat-alat pengujian beban untuk menghasilkan beban, dan analisis metrik untuk menilai kemampuan penanganan beban kerja Anda yang realistis. Pengujian beban harus dijalankan menggunakan versi data produksi yang sintetis atau sudah dibersihkan (menghapus informasi sensitif atau pengidentifikasi). Lakukan pengujian beban secara otomatis sebagai bagian dari jalur pengiriman Anda, dan bandingkan hasilnya dengan yang telah ditentukan sebelumnya KPIs dan ambang batas. Proses ini akan membantu Anda untuk terus mencapai kinerja yang dibutuhkan.

Langkah-langkah implementasi

- Tentukan tujuan pengujian Anda: Identifikasi aspek-aspek kinerja beban kerja Anda yang ingin Anda evaluasi, misalnya throughput dan waktu respons.
- Pilih alat pengujian: Pilih dan konfigurasikan alat pengujian beban yang sesuai dengan beban kerja Anda.
- Siapkan lingkungan Anda: Siapkan lingkungan pengujian berdasarkan lingkungan produksi Anda. Anda dapat menggunakan AWS layanan untuk menjalankan lingkungan skala produksi untuk menguji arsitektur Anda.
- Terapkan pemantauan: Gunakan alat pemantauan seperti Amazon CloudWatch untuk mengumpulkan metrik di seluruh sumber daya dalam arsitektur Anda. Anda juga dapat mengumpulkan dan menerbitkan metrik-metrik kustom.
- Tentukan skenario: Tentukan skenario dan parameter pengujian beban (seperti durasi pengujian dan jumlah pengguna).
- Melakukan pengujian beban: Melakukan skenario pengujian dalam skala besar. Manfaatkan AWS Cloud untuk menguji beban kerja Anda untuk menemukan di mana ia gagal untuk menskalakan, atau jika diskalakan dengan cara non-linier. Misalnya, gunakan Instans Spot untuk menghasilkan beban dengan biaya rendah dan temukan hambatan sebelum hambatan tersebut dialami di lingkungan produksi.

- Analisis hasil: Analisis hasil untuk mengidentifikasi hambatan kinerja dan area untuk perbaikan.
- Dokumentasikan dan bagikan temuan: Buatlah dokumentasi dan laporan mengenai temuan dan rekomendasi. Bagikan informasi ini kepada para pemangku kepentingan untuk membantu mereka mengambil keputusan yang cerdas mengenai strategi optimalisasi kinerja.
- Ulangi terus-menerus: Pengujian beban harus dilakukan pada irama reguler, terutama setelah perubahan pembaruan sistem.

Sumber daya

Dokumen terkait:

- Amazon CloudWatch RUM
- Amazon CloudWatch Synthetics
- Pengujian Beban Terdistribusi pada AWS

Video terkait:

- AWS Summit ANZ 2023: Mempercepat dengan percaya diri melalui Pengujian Beban AWS Terdistribusi
- AWS re:invent 2022 - Menskalakan AWS untuk 10 juta pengguna pertama Anda
- Memecahkan dengan AWS Solusi: Pengujian Beban Terdistribusi

- AWS re:invent 2021 - Optimalkan aplikasi melalui wawasan pengguna akhir dengan Amazon CloudWatch RUM
- Demo dari Amazon CloudWatch Synthetics

Contoh terkait:

- Pengujian Beban Terdistribusi pada AWS

PERF05-BP05 Gunakan otomatisasi untuk secara proaktif memulihkan masalah terkait kinerja

Gunakan indikator kinerja utama (KPIs), dikombinasikan dengan sistem pemantauan dan peringatan, untuk secara proaktif mengatasi masalah terkait kinerja.

Anti-pola umum:

- Anda hanya membekali staf operasional dengan kemampuan untuk membuat perubahan-perubahan operasional pada beban kerja.
- Anda membiarkan semua alarm disaring ke tim operasi tanpa perbaikan proaktif.

Manfaat menerapkan praktik terbaik ini: Perbaikan tindakan alarm yang proaktif akan memungkinkan staf dukungan untuk berkonsentrasi pada item-item yang tidak dapat ditindaklanjuti secara otomatis. Hal ini akan membantu staf operasi dalam menangani semua alarm tanpa merasa kewalahan dan mereka hanya berkonsentrasi pada alarm yang kritis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Gunakan alarm untuk memicu tindakan-tindakan otomatis untuk memperbaiki masalah ketika memungkinkan. Teruskan eskalasi alarm ke personel yang mampu merespons jika respons otomatis tidak memungkinkan. Misalnya, Anda mungkin memiliki sistem yang dapat memprediksi nilai indikator kinerja kunci (KPI) yang diharapkan dan alarm ketika mereka melanggar ambang tertentu, atau alat yang dapat secara otomatis menghentikan atau memutar kembali penerapan jika KPIs berada di luar nilai yang diharapkan.

Implementasikan proses yang menyediakan visibilitas tentang kinerja saat beban kerja Anda berjalan. Bangun dasbor pemantauan dan buat norma acuan untuk harapan kinerja guna menentukan apakah beban kerja mempunyai performa yang optimal.

Langkah-langkah implementasi

- Identifikasi alur kerja perbaikan: Identifikasi dan pahami masalah kinerja yang dapat diperbaiki secara otomatis. Gunakan solusi AWS pemantauan seperti Amazon CloudWatch atau AWS X-Ray untuk membantu Anda lebih memahami akar penyebab masalah.
- Tentukan proses otomatisasi: Buat proses step-by-step remediasi yang dapat digunakan untuk memperbaiki masalah secara otomatis.
- Konfigurasikan peristiwa inisiasi: Konfigurasikan peristiwa untuk memulai proses remediasi secara otomatis. Misalnya, Anda dapat menentukan pemicu untuk memulai ulang instance secara otomatis ketika mencapai ambang batas CPU pemanfaatan tertentu.
- Otomatiskan remediasi: Gunakan AWS layanan dan teknologi untuk mengotomatiskan proses remediasi. Sebagai contoh, AWS Systems Manager Automation menyediakan cara yang aman dan dapat diskalakan untuk mengotomatiskan proses perbaikan. Pastikan menggunakan logika pemulihan mandiri untuk mengembalikan perubahan jika masalah tidak berhasil diselesaikan.

- Uji alur kerja: Uji proses perbaikan otomatis di lingkungan praproduksi.
- Terapkan alur kerja: Terapkan remediasi otomatis di lingkungan produksi.
- Kembangkan playbook: Kembangkan dan dokumentasikan playbook yang menguraikan langkah-langkah untuk rencana remediasi, termasuk peristiwa inisiasi, logika remediasi, dan tindakan yang diambil. Pastikan Anda melatih pemangku kepentingan untuk membantu mereka merespons peristiwa-peristiwa perbaikan otomatis secara efektif.
- Tinjau dan perbaiki: Secara teratur lakukan evaluasi terhadap efektivitas alur kerja remediasi otomatis. Sesuaikan peristiwa inisiasi dan logika perbaikan jika perlu.

Sumber daya

Dokumen terkait:

- CloudWatchDokumentasi
- AWS Partner Network Mitra Pemantauan, Pencatatan, dan Kinerja
- Dokumentasi X-Ray
- Menggunakan Alarm dan Tindakan Alarm di CloudWatch
- Membangun Praktik Otomasi Cloud untuk Keunggulan Operasional: Praktik Terbaik dari AWS Managed Services
- Otomatiskan penyetelan kinerja Amazon Redshift Anda dengan pengoptimalan tabel otomatis

Video terkait:

- AWS re: Invent 2023 - Strategi untuk penskalaan otomatis, remediasi, dan penyembuhan diri yang cerdas
- AWS RE: invent 2023 - [LAUNCH] Pemantauan aplikasi untuk beban kerja modern
- AWS re: invent 2023 - Menerapkan observabilitas aplikasi
- AWS Re:invent 2021 - Mengotomatiskan operasi cloud secara cerdas
- AWS re:invent 2022 - Menyiapkan kontrol dalam skala besar di lingkungan Anda AWS
- AWS re:invent 2022 - Mengotomatiskan manajemen patch dan kepatuhan menggunakan AWS
- AWS re:invent 2022 - Bagaimana Amazon menggunakan metrik yang lebih baik untuk meningkatkan kinerja situs web
- AWS re:invent 2023 - Matikan beban: Mendiagnosis & menyelesaikan masalah kinerja dengan Amazon RDS

- AWS re:invent 2021 - {Peluncuran Baru} Secara otomatis mendeteksi dan menyelesaikan masalah dengan Amazon Guru DevOps
- AWS Re:invent 2023 - Pusatkan operasi Anda

Contoh terkait:

- CloudWatch Log Kustomisasi Alarm

PERF05-BP06 Pertahankan beban kerja dan layanan Anda up-to-date

Tetap up-to-date gunakan layanan dan fitur cloud baru untuk mengadopsi fitur yang efisien, menghapus masalah, dan meningkatkan efisiensi kinerja keseluruhan beban kerja Anda.

Anti-pola umum:

- Anda berasumsi bahwa arsitektur Anda saat ini adalah arsitektur statis dan tidak akan diperbarui seiring waktu.
- Anda tidak memiliki sistem atau koordinasi rutin untuk mengevaluasi apakah perangkat lunak dan paket-paket yang diperbarui kompatibel dengan beban kerja Anda.

Manfaat membangun praktik terbaik ini: Dengan membangun proses untuk tetap menggunakan layanan dan up-to-date penawaran baru, Anda dapat mengadopsi fitur dan kemampuan baru, menyelesaikan masalah, dan meningkatkan kinerja beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Evaluasi cara-cara yang dilakukan untuk meningkatkan performa saat layanan, pola desain, dan fitur produk baru tersedia. Tentukan mana hal-hal yang dapat meningkatkan kinerja atau menambah efisiensi beban kerja melalui pelaksanaan evaluasi, diskusi internal, atau analisis eksternal. Tentukan sebuah proses untuk mengevaluasi pembaruan, fitur baru, dan layanan yang relevan dengan beban kerja Anda. Misalnya, bangunlah sebuah bukti konsep yang memanfaatkan teknologi baru atau berkonsultasi dengan grup internal. Saat Anda mencoba layanan atau ide baru, jalankan pengujian kinerja untuk mengukur pengaruhnya terhadap kinerja beban kerja.

Langkah-langkah implementasi

- Buat inventaris beban kerja: Buat inventaris perangkat lunak dan arsitektur beban kerja Anda dan identifikasi komponen yang perlu diperbarui.
- Identifikasi sumber pembaruan: Identifikasi sumber berita dan pembaruan yang terkait dengan komponen beban kerja Anda. Sebagai contoh, Anda dapat berlangganan AWS Blog What's New at untuk produk yang sesuai dengan komponen beban kerja Anda. Anda dapat berlangganan RSS feed atau mengelola langganan email Anda.
- Tentukan jadwal pembaruan: Tentukan jadwal untuk mengevaluasi layanan dan fitur baru untuk beban kerja Anda.
  - Anda dapat menggunakan AWS Systems Manager Inventaris untuk mengumpulkan sistem operasi (OS), aplikasi, dan metadata instans dari instans Amazon Anda dan dengan cepat memahami EC2 instans mana yang menjalankan perangkat lunak dan konfigurasi yang diperlukan oleh kebijakan perangkat lunak Anda dan instans mana yang perlu diperbarui.
- Nilai pembaruan baru: Pahami cara memperbarui komponen beban kerja Anda. Manfaatkan ketangkasan di cloud untuk melakukan uji cepat mengenai bagaimana fitur-fitur baru dapat meningkatkan beban kerja Anda untuk mendapatkan efisiensi performa.
- Gunakan otomatisasi: Gunakan otomatisasi untuk proses pembaruan guna mengurangi tingkat upaya dalam melakukan deployment fitur baru dan membatasi kesalahan yang disebabkan oleh proses manual.
  - Anda dapat menggunakan CI/CD untuk memperbaruiAMIs, gambar kontainer, dan artefak lain yang terkait dengan aplikasi cloud Anda secara otomatis.
  - Anda dapat menggunakan alat-alat seperti AWS Systems Manager Patch Manager untuk melakukan otomatisasi terhadap proses pembaruan sistem, dan menjadwalkan aktivitas dengan menggunakan AWS Systems Manager Windows Maintenance.
- Dokumentasikan proses: Dokumentasikan proses Anda untuk mengevaluasi pembaruan dan layanan baru. Bekali para pemilik Anda dengan waktu dan ruang yang dibutuhkan untuk meneliti, menguji, melakukan eksperimen, serta memvalidasi pembaruan dan layanan baru. Lihat kembali persyaratan bisnis yang terdokumentasi dan KPIs untuk membantu memprioritaskan pembaruan mana yang akan membuat dampak bisnis yang positif.

Sumber daya

Dokumen terkait:

- Blog AWS

- Apa yang baru dengan AWS
- Menerapkan up-to-date gambar dengan pipeline EC2 Image Builder otomatis

Video terkait:

- AWS Re: Inforce 2022 - Mengotomatiskan manajemen patch dan kepatuhan menggunakan AWS
- Semua Hal Patch: AWS Systems Manager | AWS Acara

Contoh terkait:

- Manajemen Inventaris dan Patch
- Lokakarya Satu Observabilitas

PERF05-BP07 Tinjau metrik secara berkala

Sebagai bagian pemeliharaan rutin, atau sebagai respons terhadap peristiwa atau insiden, tinjau metrik mana yang dikumpulkan. Gunakan tinjauan ini untuk mengidentifikasi metrik mana yang penting untuk menangani masalah dan metrik mana yang merupakan tambahan. Jika dilacak, metrik tersebut dapat memudahkan Anda mengidentifikasi, mengatasi, dan mencegah masalah.

Anti-pola umum:

- Anda mengizinkan metrik-metrik untuk tetap dalam status alarm selama periode waktu yang lebih lama.
- Anda memberikan alarm yang tidak dapat ditindaklanjuti oleh sistem otomatisasi.

Manfaat menerapkan praktik terbaik ini: Lakukan peninjauan secara terus-menerus terhadap metrik yang sedang dikumpulkan untuk memverifikasi bahwa metrik tersebut dapat mengidentifikasi, mengatasi, atau mencegah masalah. Metrik juga dapat mengalami kedaluwarsa jika Anda membiarkannya berada dalam status alarm untuk waktu yang lama.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Lakukan peningkatan pemantauan dan pengumpulan metrik secara konstan. Sebagai bagian dari tindakan merespons insiden atau peristiwa, evaluasikan mana metrik yang berguna untuk mengatasi masalah dan mana metrik yang dapat membantu tetapi saat ini tidak terdeteksi. Gunakan metode ini

untuk meningkatkan kualitas metrik yang Anda kumpulkan agar Anda dapat mencegah, atau agar Anda dapat menangani insiden pada masa mendatang dengan lebih cepat.

Sebagai bagian dari tindakan merespons insiden atau peristiwa, evaluasikan mana metrik yang berguna untuk mengatasi masalah dan mana metrik yang dapat membantu tetapi saat ini tidak terdeteksi. Gunakan ini untuk meningkatkan kualitas metrik yang Anda kumpulkan agar Anda dapat mencegah atau dapat mengatasi insiden di masa mendatang dengan lebih cepat.

Langkah-langkah implementasi

- Tentukan metrik: Tentukan metrik kinerja penting untuk memantau apakah mereka selaras dengan tujuan beban kerja Anda, termasuk metrik seperti waktu respons dan pemanfaatan sumber daya.
- Tetapkan garis acuan: Tetapkan garis acuan dan nilai yang diinginkan untuk setiap metrik. Garis acuan harus memberikan titik-titik referensi untuk mengidentifikasi penyimpangan atau anomali.
- Tetapkan frekuensi: Tetapkan frekuensi (seperti mingguan atau bulanan) untuk meninjau metrik penting.
- Identifikasi masalah performa: Dalam setiap tinjauan, nilai tren dan penyimpangan dari nilai garis acuan. Cari setiap anomali atau hambatan performa. Untuk masalah-masalah yang berhasil diidentifikasi, lakukan analisis akar penyebab secara mendalam untuk memahami alasan utama di balik masalah tersebut.
- Identifikasi tindakan-tindakan korektif: Gunakan analisis Anda untuk mengidentifikasi tindakan korektif. Tindakan tersebut antara lain penyesuaian parameter, perbaikan bug, dan penskalaan sumber daya.
- Dokumentasikan temuan: Dokumentasikan temuan Anda, termasuk masalah yang diidentifikasi, akar penyebab, dan tindakan korektif.
- Ulangi dan tingkatkan: Terus nilai dan tingkatkan proses peninjauan metrik. Gunakan pelajaran yang dipetik dari tinjauan sebelumnya untuk menyempurnakan proses dari waktu ke waktu.

Sumber daya

Dokumen terkait:

- CloudWatchDokumentasi
- Kumpulkan metrik dan log dari EC2 Instans Amazon dan server lokal dengan Agen CloudWatch
- Kueri metrik Anda dengan Wawasan CloudWatch Metrik
- AWS Partner Network Mitra Pemantauan, Pencatatan, dan Kinerja

- Dokumentasi X-Ray

Video terkait:

- AWS re:invent 2022 - Menyiapkan kontrol dalam skala besar di lingkungan Anda AWS
- AWS re:invent 2022 - Bagaimana Amazon menggunakan metrik yang lebih baik untuk meningkatkan kinerja situs web
- AWS re:invent 2023 - Membangun strategi observabilitas yang efektif
- AWS Summit SF 2022 - Observabilitas tumpukan penuh dan pemantauan aplikasi dengan AWS
- AWS re:invent 2023 - Matikan beban: Mendiagnosis & menyelesaikan masalah kinerja dengan Amazon RDS

Contoh terkait:

- Membuat dasbor dengan Amazon QuickSight
- CloudWatch Dasbor

# Optimalisasi Biaya

Pilar Optimalisasi Biaya mencakup kemampuan untuk menjalankan sistem guna menghadirkan nilai bisnis dengan harga yang paling rendah. Anda dapat menemukan panduan preskriptif tentang implementasi di Dokumen Whitepaper Pilar Optimalisasi Biaya.

Area praktik terbaik

- Mempraktikkan Manajemen Keuangan Cloud
- Kesadaran akan penggunaan dan pengeluaran
- Sumber daya hemat biaya
- Kelola sumber daya pasokan dan permintaan
- Pengoptimalan dari waktu ke waktu

## Mempraktikkan Manajemen Keuangan Cloud

Pertanyaan

- COST1. Bagaimana cara mengimplementasikan manajemen keuangan cloud?

## COST1. Bagaimana cara mengimplementasikan manajemen keuangan cloud?

Menerapkan Cloud Financial Management membantu organisasi mewujudkan nilai bisnis dan kesuksesan finansial saat mereka mengoptimalkan biaya dan penggunaan serta skalanya AWS.

Praktik terbaik

- COST01-BP01 Menetapkan kepemilikan optimalisasi biaya
- COST01-BP02 Menjalin kemitraan antara keuangan dan teknologi
- COST01-BP03 Menetapkan anggaran dan prakiraan cloud
- COST01-BP04 Menerapkan kesadaran biaya dalam proses organisasi Anda
- COST01-BP05 Laporkan dan beri tahu tentang optimalisasi biaya
- COST01-BP06 Monitor biaya proaktif
- COST01-BP07 Tetap up-to-date dengan rilis layanan baru
- COST01-BP08 Ciptakan budaya sadar biaya
- COST01-BP09 Mengukur nilai bisnis dari optimalisasi biaya

### COST01-BP01 Menetapkan kepemilikan optimalisasi biaya

Buat tim (Cloud Business Office, Cloud Center of Excellence, atau FinOps tim) yang bertanggung jawab untuk membangun dan memelihara kesadaran biaya di seluruh organisasi Anda. Pemilik optimasi biaya dapat berupa individu atau tim (membutuhkan orang-orang dari tim keuangan, teknologi, dan bisnis) yang memahami seluruh organisasi dan keuangan cloud.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Ini adalah pengenalan fungsi atau tim Cloud Business Office (CBO) atau Cloud Center of Excellence (CCOE) yang bertanggung jawab untuk membangun dan memelihara budaya kesadaran biaya dalam komputasi awan. Fungsi ini dapat terdiri dari individu yang sudah ada sekarang, tim di dalam organisasi Anda, atau tim baru yang terdiri dari pemangku kepentingan utama organisasi, keuangan, dan teknologi dari seluruh organisasi.

Fungsi ini (individu atau tim) memprioritaskan dan menggunakan sebagian besar waktunya untuk aktivitas manajemen dan pengoptimalan biaya. Untuk organisasi kecil, fungsi ini mungkin memerlukan persentase waktu yang lebih sedikit dibandingkan dengan fungsi purnawaktu di korporasi yang lebih besar.

Fungsi ini memerlukan pendekatan multidisiplin, dengan kemampuan di bidang manajemen proyek, ilmu data, analisis keuangan, dan pengembangan perangkat lunak atau infrastruktur. Fungsi ini dapat meningkatkan efisiensi beban kerja dengan menjalankan pengoptimalan biaya dalam tiga kepemilikan yang berbeda:

- Terpusat: Melalui tim yang ditunjuk seperti tim, FinOps tim Cloud Financial Management (CFM), Cloud Business Office (CBO), atau Cloud Center of Excellence (CCoE), pelanggan dapat merancang dan menerapkan mekanisme tata kelola dan mendorong praktik terbaik di seluruh perusahaan.
- Terdesentralisasi: Memengaruhi tim-tim teknologi untuk menjalankan optimasi biaya.
- Hibrida: Gabungan dari tim tersentralisasi dan terdesentralisasi dapat bekerja sama untuk menjalankan optimasi biaya.

Fungsi ini dapat diukur dari kemampuannya menjalankan dan menyampaikan tujuan pengoptimalan biaya (misalnya, metrik efisiensi beban kerja).

Anda harus mendapatkan sponsor eksekutif untuk fungsi ini, yang merupakan faktor keberhasilan utama. Sponsor ini dianggap sebagai penyokong penggunaan cloud hemat biaya, serta memberikan dukungan peningkatan kepada tim guna memastikan bahwa aktivitas pengoptimalan biaya ditangani menurut tingkat prioritas yang ditentukan oleh organisasi. Jika tidak, panduan dapat diabaikan dan peluang penghematan biaya tidak akan diprioritaskan. Bersama-sama, sponsor dan tim membantu organisasi Anda menggunakan cloud secara efisien dan menghadirkan nilai bisnis.

Jika Anda memiliki paket dukungan Bisnis, Enterprise-On-Ramp atau Perusahaan dan memerlukan bantuan untuk membangun tim atau fungsi ini, hubungi pakar Cloud Financial Management (CFM) Anda melalui tim akun Anda.

Langkah-langkah implementasi

- Tentukan para anggota inti: Semua bagian yang relevan dari organisasi Anda harus berkontribusi dan berminat pada manajemen biaya. Tim umum dalam organisasi biasanya meliputi: keuangan, aplikasi atau pemilik produk, manajemen, dan tim teknis (DevOps). Beberapa dari mereka terlibat secara purnawaktu (keuangan atau teknis), sementara yang lain dilibatkan secara berkala sesuai kebutuhan. Individu atau tim yang berkinerja CFM membutuhkan seperangkat keterampilan berikut:
  - Keahlian pengembangan perangkat lunak: jika skrip dan otomatisasi sedang dibuat.
  - Rekayasa infrastruktur: untuk men-deploy skrip, mengotomatisasi proses, dan memahami bagaimana layanan atau sumber daya disediakan.

  - Ketajaman operasi: CFM adalah tentang beroperasi di cloud secara efisien dengan mengukur, memantau, memodifikasi, merencanakan, dan menskalakan penggunaan cloud yang efisien.
- Tentukan tujuan dan metrik: Fungsi harus memberikan nilai kepada organisasi dengan cara yang berbeda. Tujuan-tujuan tersebut ditetapkan dan terus berkembang seiring dengan perkembangan organisasi. Aktivitas umum mencakup: membuat dan menjalankan program edukasi tentang pengoptimalan biaya di seluruh organisasi, mengembangkan standar di seluruh organisasi, seperti pemantauan dan pelaporan untuk pengoptimalan biaya, dan menetapkan sasaran beban kerja dalam pengoptimalan. Fungsi juga harus melaporkan kemampuan pengoptimalan biaya mereka secara rutin kepada organisasi.

  Anda dapat menentukan indikator kinerja kunci berbasis nilai atau biaya ()KPIs. Ketika Anda menentukanKPIs, Anda dapat menghitung biaya yang diharapkan dalam hal efisiensi dan hasil bisnis yang diharapkan. Berbasis nilai KPIs mengikat metrik biaya dan penggunaan dengan pendorong nilai bisnis dan membantu merasionalisasi perubahan dalam pengeluaran. AWS Langkah pertama untuk mendapatkan berbasis nilai KPIs adalah bekerja sama, lintas organisasi, untuk memilih dan menyetujui seperangkat standar. KPIs
- Tetapkan jadwal pertemuan rutin: Departemen (keuangan, teknologi, dan tim bisnis) harus melakukan rapat secara teratur untuk meninjau sasaran dan metrik mereka. Koordinasi ini biasanya membahas peninjauan status organisasi, program apa pun yang sedang berlangsung, serta metrik pengoptimalan dan keuangan secara keseluruhan. Selanjutnya, beban kerja utama dilaporkan dengan lebih mendetail.

  Selama tinjauan rutin ini, Anda dapat meninjau efisiensi beban kerja (biaya) dan hasil bisnis. Misalnya, kenaikan biaya 20% untuk beban kerja dapat diselaraskan dengan peningkatan penggunaan pelanggan. Dalam kasus ini, kenaikan biaya 20% dapat diartikan sebagai investasi. Panggilan irama reguler ini dapat membantu tim mengidentifikasi nilai KPIs yang memberi makna bagi seluruh organisasi.

Sumber daya

Dokumen terkait:

- Blog AWS CCOE
- Membuat Kantor Bisnis Cloud
- CCOE- Pusat Keunggulan Cloud

Video terkait:

- Kisah Sukses Vanguard CCOE

Contoh terkait:

- Menggunakan Cloud Center of Excellence (CCOE) untuk Mengubah Seluruh Perusahaan
- Membangun CCOE untuk mengubah seluruh perusahaan
- 7 Jebakan yang Harus Dihindari Saat Membangun CCOE

COST01-BP02 Menjalin kemitraan antara keuangan dan teknologi

Libatkan tim keuangan dan teknologi dalam diskusi biaya dan penggunaan di semua tahap perjalanan cloud Anda. Tim secara teratur bertemu dan membahas topik-topik seperti target dan tujuan organisasi, kondisi biaya dan penggunaan saat ini, serta praktik keuangan dan akuntansi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Tim teknologi berinovasi lebih cepat di cloud karena siklus deployment infrastruktur, pengadaan, dan persetujuan yang lebih pendek. Ini dapat membutuhkan penyesuaian untuk organisasi keuangan yang sebelumnya terbiasa menjalankan proses yang memerlukan waktu lama dan banyak sumber daya untuk mendapatkan dan melakukan deployment modal di lingkungan on-premise dan pusat data, dan alokasi biaya hanya berdasarkan persetujuan proyek.

Dari perspektif organisasi keuangan dan pengadaan, proses penganggaran modal, permintaan modal, persetujuan, pengadaan, dan pemasangan infrastruktur fisik adalah proses yang telah dipelajari dan distandardisasi selama beberapa dekade:

- Tim rekayasa atau IT biasanya adalah pemohon
- Berbagai tim keuangan bertindak sebagai pemberi persetujuan dan pengadaan
- Tim operasi merakit, menumpuk, dan menyerahkan ready-to-use infrastruktur


Dengan penerapan cloud, pengadaan dan penggunaan infrastruktur tidak lagi terikat pada rantai dependensi. Dalam model cloud, tim teknologi dan produk tidak lagi hanya builder, tetapi operator dan pemilik produk mereka, bertanggung jawab atas sebagian besar aktivitas yang secara historis terkait dengan tim keuangan dan operasi, termasuk pengadaan dan deployment.

Hal yang diperlukan untuk menyediakan sumber daya cloud adalah sebuah akun, dan serangkaian izin yang tepat. Ini juga yang mengurangi risiko TI dan keuangan; yang berarti tim selalu hanya dengan beberapa klik atau API panggilan dari penghentian sumber daya cloud yang menganggur atau tidak perlu. Hal ini juga yang memungkinkan tim teknologi berinovasi lebih cepat – ketangkasan dan kemampuan untuk menjalankan dan kemudian menghentikan eksperimen. Sementara sifat variabel penggunaan cloud dapat memengaruhi prediktabilitas dari perspektif penganggaran dan

prakiraan modal, cloud memberi organisasi kemampuan untuk mengurangi biaya penyediaan yang berlebihan, serta mengurangi biaya peluang yang terkait dengan kekurangan penyediaan yang konservatif.


Buat kemitraan antara pemangku kepentingan penting dalam bidang keuangan dan teknologi untuk menghasilkan pemahaman bersama akan tujuan organisasi dan kembangkan mekanisme agar berhasil secara finansial dalam model pengeluaran variabel komputasi cloud. Tim yang relevan dalam organisasi Anda harus dilibatkan dalam diskusi biaya dan penggunaan di semua tahap perjalanan cloud Anda, termasuk:

- Prospek keuangan:CFOs, pengontrol keuangan, perencana keuangan, analis bisnis, pengadaan, sumber, dan hutang dagang harus memahami model konsumsi cloud, opsi pembelian, dan proses faktur bulanan. Tim keuangan perlu bermitra dengan tim teknologi untuk membuat dan menyosialisasikan kisah nilai IT, sehingga membantu tim bisnis memahami bagaimana pengeluaran teknologi terkait dengan hasil bisnis. Dengan cara ini, pengeluaran teknologi tidak

dipandang sebagai biaya, melainkan sebagai investasi. Karena perbedaan mendasar antara cloud (seperti laju perubahan dalam penggunaan, penetapan harga bayar sesuai pemakaian, harga berjenjang, model harga, dan informasi penggunaan serta penagihan mendetail) dibandingkan dengan operasi on-premise, maka penting organisasi keuangan memahami bagaimana penggunaan cloud dapat memengaruhi aspek bisnis, termasuk proses pengadaan, pelacakan insentif, alokasi biaya, dan laporan keuangan.

- Kepala bagian teknologi: Kepala bagian teknologi (termasuk pemilik aplikasi dan produk) harus sadar akan persyaratan keuangan (misalnya, batas anggaran) serta persyaratan bisnis (misalnya, perjanjian tingkat layanan). Ini memungkinkan beban kerja diimplementasikan untuk mencapai tujuan yang diinginkan organisasi.

Kemitraan antara bagian keuangan dan teknologi memberikan manfaat berikut:

- Tim keuangan dan teknologi memiliki visibilitas hampir dalam waktu nyata ke biaya dan penggunaan.
- Tim keuangan dan teknologi menetapkan prosedur pengoperasian standar untuk menangani varian pengeluaran cloud.
- Pemangku kepentingan keuangan bertindak sebagai penasihat strategis sehubungan dengan bagaimana modal digunakan untuk membeli diskon komitmen (misalnya, Instans Cadangan atau AWS Savings Plans), dan bagaimana cloud digunakan untuk menumbuhkan organisasi.
- Proses pengadaan dan utang usaha yang ada digunakan dengan cloud.
- Tim keuangan dan teknologi berkolaborasi dalam memperkirakan AWS biaya dan penggunaan masa depan untuk menyelaraskan dan membangun anggaran organisasi.
- Komunikasi lintas organisasi yang lebih baik melalui bahasa bersama, dan pemahaman bersama akan konsep keuangan.

Pemangku kepentingan tambahan di dalam organisasi Anda yang harus dilibatkan dalam diskusi biaya dan penggunaan termasuk:

- Pemilik unit bisnis: Pemilik unit bisnis harus memahami model bisnis cloud sehingga mereka dapat memberikan pengarahan kepada unit bisnis dan seluruh perusahaan. Pengetahuan cloud ini sangat penting ketika ada kebutuhan untuk memprediksi pertumbuhan dan penggunaan beban kerja, dan ketika mengevaluasi opsi-opsi pembelian jangka lebih panjang, seperti Instans Terpesan atau Savings Plans.

- Tim teknik: Membangun kemitraan antara tim keuangan dan teknologi sangat penting untuk membangun budaya sadar biaya yang mendorong para insinyur untuk mengambil tindakan pada Cloud Financial Management (). CFM Salah satu masalah umum dari CFM praktisi operasi keuangan dan tim keuangan adalah membuat para insinyur memahami seluruh bisnis di cloud, mengikuti praktik terbaik, dan mengambil tindakan yang direkomendasikan.
- Pihak ketiga: Jika organisasi Anda menggunakan pihak ketiga (misalnya, konsultan atau alat), pastikan bahwa mereka selaras dengan tujuan keuangan Anda dan dapat menunjukkan keselarasan melalui model keterlibatan mereka dan laba atas investasi ()ROI. Umumnya, pihak ketiga akan berkontribusi dalam pelaporan dan analisis beban kerja apa pun yang mereka kelola, dan mereka akan memberikan analisis biaya beban kerja apa pun yang mereka desain.

Menerapkan CFM dan mencapai kesuksesan membutuhkan kolaborasi di seluruh keuangan, teknologi, dan tim bisnis, dan pergeseran dalam bagaimana belanja cloud dikomunikasikan dan dievaluasi di seluruh organisasi. Ikut sertakan tim rekayasa sehingga mereka dapat menjadi bagian dari diskusi biaya dan penggunaan ini di semua tahap, dan dorong mereka untuk mengikuti praktik terbaik dan mengambil tindakan yang disepakati.

Langkah-langkah implementasi

- Tentukan anggota utama: Verifikasi bahwa semua anggota yang relevan dari tim keuangan dan teknologi Anda ambil bagian dalam kemitraan. Anggota yang relevan dari tim keuangan adalah mereka yang memiliki interaksi dengan tagihan cloud. Ini biasanya, pengontrol keuanganCFOs, perencana keuangan, analis bisnis, pengadaan, dan sumber. Umumnya, anggota bagian teknologi adalah pemilik aplikasi dan produk, manajer teknis, dan perwakilan dari semua tim yang membangun di cloud. Anggota-anggota lainnya dapat mencakup pemilik unit bisnis, misalnya bagian pemasaran, yang akan memengaruhi penggunaan produk, dan pihak ketiga seperti konsultan, untuk mencapai keselarasan dengan tujuan dan mekanisme Anda, dan untuk membantu dalam pelaporan.
- Tentukan topik yang akan dibahas: Tentukan topik yang bersifat umum lintas tim, atau yang akan membutuhkan pemahaman bersama. Ikuti biaya dari waktu biaya dibuat, sampai tagihan dibayar. Catat setiap anggota yang terlibat, dan proses organisasi yang harus diterapkan. Pahami setiap langkah atau proses yang dilewatinya dan informasi terkait, seperti model harga yang tersedia, harga berjenjang, model diskon, penetapan anggaran, dan persyaratan keuangan.
- Tetapkan jadwal pertemuan rutin:  Untuk menciptakan kemitraan keuangan dan teknologi, buatlah koordinasi komunikasi yang teratur untuk menciptakan dan mempertahankan keselarasan. Grup tersebut harus mengadakan pertemuan koordinasi secara rutin untuk membahas tujuan dan

metrik. Koordinasi ini biasanya membahas peninjauan status organisasi, program apa pun yang sedang berlangsung, serta metrik pengoptimalan dan keuangan secara keseluruhan. Kemudian beban kerja utama dilaporkan dengan lebih mendetail.

Sumber daya

Dokumen terkait:

- AWS Blog Berita

COST01-BP03 Menetapkan anggaran dan prakiraan cloud

Sesuaikan proses pemrakiraan dan penganggaran yang ada di organisasi agar kompatibel dengan biaya dan penggunaan cloud yang sangat bervariasi. Prosesnya harus dinamis menggunakan algoritma berbasis pendorong bisnis atau berbasis tren, atau kombinasi dari keduanya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Dalam pengaturan IT on-premise konvensional, pelanggan sering menghadapi tantangan perencanaan biaya tetap yang hanya berubah sesekali, biasanya dengan pembelian perangkat keras dan layanan IT baru untuk memenuhi permintaan puncak. Sebaliknya, AWS Cloud mengadopsi pendekatan yang berbeda, di mana pelanggan membayar sumber daya yang mereka gunakan sebagaimana ditentukan oleh kebutuhan TI dan bisnis mereka yang sebenarnya. Di lingkungan cloud, permintaan dapat mengalami fluktuasi setiap bulan, setiap hari, atau bahkan setiap jam.

Dengan menggunakan cloud, pelanggan memperoleh efisiensi, kecepatan, dan ketangkasan, yang menghasilkan pola biaya dan penggunaan yang sangat bervariasi. Biaya dapat berkurang atau terkadang bertambah dikarenakan efisiensi beban kerja yang lebih besar atau deployment beban kerja dan fitur baru. Seiring meningkatnya skala beban kerja untuk melayani basis pelanggan yang berkembang, penggunaan dan biaya cloud juga meningkat disebabkan peningkatan aksesibilitas sumber daya. Fleksibilitas dalam layanan cloud ini meluas hingga ke biaya dan prakiraan, yang menciptakan elastisitas pada tingkat tertentu.

Sangat penting melakukan penyelarasan dengan perubahan kebutuhan bisnis dan faktor pendorong permintaan ini, dan menargetkan perencanaan yang seakurat mungkin. Proses anggaran organisasi konvensional perlu beradaptasi untuk mengakomodasi variabilitas ini.

Pertimbangkan pemodelan biaya saat Anda memprakirakan biaya untuk beban kerja baru. Pemodelan biaya menciptakan pemahaman dasar tentang biaya cloud yang diharapkan, yang membantu Anda melakukan total biaya kepemilikan (TCO), laba atas investasi (ROI), dan analisis keuangan lainnya, menetapkan target dan harapan dengan pemangku kepentingan, dan mengidentifikasi peluang untuk optimalisasi biaya.

Organisasi Anda harus memahami penetapan biaya dan pengelompokan yang diterima. Tingkat detail prakiraan Anda dapat bervariasi tergantung struktur organisasi dan alur kerja internal Anda. Pilih tingkat detail yang sesuai dengan kebutuhan khusus dan pengaturan organisasi Anda. Penting untuk memahami pada tingkat apa prakiraan dilakukan:

- Akun manajemen atau tingkat AWS Organizations : Akun manajemen adalah akun yang Anda gunakan untuk membuat AWS Organizations. Organisasi akan memiliki satu akun manajemen secara default.
- Akun tertaut atau anggota: Akun di Organizations adalah standar Akun AWS yang berisi AWS sumber daya Anda dan identitas yang dapat mengakses sumber daya tersebut.
- Lingkungan: Lingkungan adalah kumpulan sumber AWS daya yang menjalankan versi aplikasi. Lingkungan dapat dibuat dengan beberapa akun tertaut atau anggota.
- Proyek: Sebuah proyek adalah kombinasi dari sekumpulan tujuan atau tugas yang harus diselesaikan dalam satu periode yang sudah ditentukan. Penting untuk mempertimbangkan siklus hidup proyek selama prakiraan Anda.
- AWS Layanan: Grup atau kategori seperti layanan komputasi atau penyimpanan tempat Anda dapat mengelompokkan AWS layanan untuk perkiraan Anda.
- Pengelompokan yang menyesuaikan kebutuhan: Anda dapat membuat pengelompokan yang menyesuaikan kebutuhan organisasi Anda, seperti unit bisnis, pusat biaya, tim, tag alokasi biaya, kategori biaya, akun tertaut, atau kombinasi dari semuanya.

Identifikasi pendorong bisnis yang dapat berdampak pada biaya penggunaan Anda, dan lakukan prakiraan untuk masing-masing secara terpisah untuk menghitung perkiraan penggunaan di awal. Beberapa pendorong dapat terkait dengan tim IT dan produk di dalam organisasi. Pendorong bisnis lainnya, seperti acara pemasaran, promosi, ekspansi geografis, merger, dan akuisisi, dikenal oleh pemimpin penjualan, pemasaran, dan bisnis Anda, dan penting juga untuk berkolaborasi dan memperhitungkan semua pendorong permintaan tersebut.

Anda dapat menggunakan AWS Cost Explorer untuk peramalan berbasis tren dalam rentang waktu masa depan yang ditentukan berdasarkan pengeluaran masa lalu Anda. AWS Cost Explorer mesin

peramalan mengelompokkan data historis Anda berdasarkan jenis pengisian daya (misalnya, Instans Cadangan) dan menggunakan kombinasi pembelajaran mesin dan model berbasis aturan untuk memprediksi pengeluaran di semua jenis pengisian daya secara individual.

Setelah Anda menetapkan proses perkiraan dan model yang dibuat, Anda dapat menggunakan AWS Budgetsuntuk mengatur anggaran khusus pada tingkat granular dengan menentukan periode waktu, pengulangan, atau jumlah (tetap atau variabel) dan menambahkan filter seperti layanan,, Wilayah AWS dan tag. Anggaran biasanya disiapkan untuk satu tahun dan tidak berubah-ubah, yang membutuhkan kepatuhan yang ketat dari semua orang yang terlibat. Sebaliknya, prakiraan lebih fleksibel, yang memungkinkan penyesuaian ulang sepanjang tahun dan memberikan proyeksi dinamis selama periode satu, dua, atau tiga tahun. Baik anggaran maupun prakiraan memainkan peran penting ketika Anda membangun ekspektasi keuangan di kalangan berbagai pemangku kepentingan teknologi dan bisnis. Prakiraan dan implementasi yang akurat juga menghadirkan akuntabilitas bagi para pemangku kepentingan yang bertanggung jawab langsung atas penyediaan biaya sejak awal, serta dapat meningkatkan kesadaran biaya mereka secara keseluruhan.

Agar terus dapat memantau kinerja anggaran yang ada, Anda dapat membuat dan menjadwalkan laporan AWS Budgets untuk dikirim melalui email kepada Anda dan pemangku kepentingan Anda secara rutin. Anda juga dapat membuat peringatan AWS Budgets berdasarkan biaya aktual, yang bersifat reaktif, atau biaya yang diprakirakan, yang menyediakan waktu untuk menerapkan mitigasi terhadap potensi kelebihan biaya. Anda dapat menerima peringatan saat biaya atau penggunaan benar-benar melampaui level tertentu atau jika diprakirakan akan melampaui jumlah yang dianggarkan.

Sesuaikan proses prakiraan dan anggaran yang ada agar menjadi lebih dinamis menggunakan algoritma berbasis tren (dengan biaya historis sebagai masukan), dan menggunakan algoritma berbasis pendorong bisnis (misalnya, peluncuran produk baru atau ekspansi Regional), yang ideal untuk lingkungan penganggaran dinamis dan bervariasi. Setelah Anda menentukan perkiraan berbasis tren menggunakan Cost Explorer atau alat lainnya, gunakan AWS Pricing Calculatoruntuk memperkirakan kasus penggunaan dan biaya future berdasarkan AWS penggunaan yang diharapkan (lalu lintas requests-per-second, atau EC2 instans Amazon yang diperlukan).

Lacak keakuratan prakiraan tersebut, karena anggaran harus ditetapkan berdasarkan perhitungan dan estimasi prakiraan ini. Pantau keakuratan dan efektivitas prakiraan biaya cloud terintegrasi. Tinjau secara rutin pengeluaran aktual dibandingkan dengan prakiraan Anda, dan sesuaikan dengan kebutuhan untuk meningkatkan presisi prakiraan. Lacak selisih prakiraan, dan lakukan analisis akar masalah pada selisih yang dilaporkan untuk bertindak dan menyesuaikan prakiraan.

Sebagaimana disebutkan dalam COST01-BP02 Menjalin kemitraan antara keuangan dan teknologi, penting untuk membina kemitraan dan koordinasi antara IT, keuangan, dan pemangku kepentingan lainnya untuk memverifikasi bahwa mereka semua menggunakan alat atau proses yang sama untuk konsistensi. Dalam kasus yang mungkin memerlukan perubahan anggaran, tingkatkan frekuensi rapat singkat koordinasi untuk bereaksi terhadap perubahan tersebut secara lebih cepat.

Langkah-langkah implementasi

- Tentukan bahasa biaya dalam organisasi: Buat bahasa AWS biaya umum dalam Organisasi dengan berbagai dimensi dan pengelompokan. Pastikan pemangku kepentingan memahami tingkat detail prakiraan, model harga, dan tingkat prakiraan biaya Anda.
- Analisis prakiraan berbasis tren: Gunakan alat prakiraan berbasis tren seperti AWS Cost Explorer dan Amazon Forecast. Analisis biaya penggunaan Anda pada beberapa dimensi-dimensi seperti kategori layanan, akun, tag, dan biaya. Jika prakiraan lanjutan diperlukan, impor data AWS Biaya dan Penggunaan (CUR) Anda ke Amazon Forecast (yang menerapkan regresi linier sebagai bentuk pembelajaran mesin untuk memperkirakan).
- Analisis prakiraan berbasis faktor pendorong Tentukan dampak faktor-faktor bisnis pada penggunaan cloud Anda, dan prediksi setiap faktor bisnis satu-persatu untuk menghitung biaya yang dapat terkira di kemudian hari. Bekerjasamalah dengan pemilik unit bisnis dan pemangku kepentingan untuk memahami dampak pada pendorong baru, dan menghitung perkiraan perubahan biaya untuk menentukan anggaran yang akurat.
- Perbarui prakiraan yang sudah ada dan proses penganggaran: Berdasarkan metode prakiraan yang telah digunakan seperti prakiraan berbasis tren, berbasis pendorong bisnis, atau kombinasi dari kedua metode prakiraan tersebut, tentukan prakiraan Anda dan proses penganggarannya. Anggaran harus terhitung, realistis, dan didasarkan pada proses prakiraan Anda.
- Konfigurasikan peringatan dan notifikasi: Gunakan AWS Budgets peringatan dan deteksi anomali biaya untuk mendapatkan peringatan dan pemberitahuan.
- Lakukan peninjauan rutin bersama para pembangku jabatan inti: Misalnya, menyamakan persepsi tentang perubahan arah bisnis dan penggunaannya bersama para pihak penting di bagian IT, keuangan, platform, dan bidang bisnis lainnya.

Sumber daya

Dokumen terkait:

- AWS Cost Explorer
- AWS Cost and Usage Report

- Melakukan Prakiraan dengan Cost Explorer
- QuickSight Peramalan Amazon
- Amazon Forecast
- AWS Budgets

Video terkait:

- Bagaimana saya bisa menggunakan AWS Budgets untuk melacak pengeluaran dan penggunaan saya
- AWS Seri Optimalisasi Biaya: AWS Budgets

Contoh terkait:

- Memahami dan membangun prakiraan berbasis pendorong
- Cara membangun dan mendorong budaya prakiraan
- Cara meningkatkan prakiraan biaya cloud Anda
- Menggunakan alat yang tepat untuk prakiraan biaya cloud Anda

COST01-BP04 Menerapkan kesadaran biaya dalam proses organisasi Anda

Implementasikan kesadaran biaya, transparansi, dan akuntabilitas biaya ke dalam proses baru atau yang sudah ada yang memengaruhi penggunaan, serta manfaatkan proses yang sudah ada untuk kesadaran biaya. Implementasikan kesadaran biaya ke dalam pelatihan karyawan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kesadaran biaya harus diimplementasikan di dalam proses-proses baru dan yang sudah ada. Ini adalah salah satu kemampuan dasar dan prasyarat untuk praktik terbaik lainnya. Disarankan untuk menggunakan ulang dan memodifikasi proses yang sudah ada jika memungkinkan—hal ini meminimalkan dampak terhadap ketangkasan dan kecepatan. Laporkan biaya cloud kepada tim teknologi dan pengambil keputusan dalam tim bisnis dan keuangan untuk meningkatkan kesadaran biaya, dan menetapkan indikator kinerja utama efisiensi (KPIs) untuk pemangku kepentingan keuangan dan bisnis. Saran berikut ini akan membantu mengimplementasikan kesadaran biaya di dalam beban kerja Anda:

- Pastikan manajemen perubahan mencakup pengukuran biaya untuk menghitung dampak keuangan dari perubahan Anda. Hal ini membantu Anda secara proaktif menangani masalah terkait biaya dan menyoroti penghematan biaya.
- Pastikan pengoptimalan biaya adalah komponen inti kemampuan operasional Anda. Misalnya, Anda dapat memanfaatkan proses manajemen insiden yang sudah ada untuk menyelidiki dan mengidentifikasi akar masalah untuk anomali biaya dan penggunaan atau kelebihan biaya.
- Percepat penghematan biaya dan realisasi nilai bisnis melalui otomatisasi atau penggunaan alat. Ketika memikirkan biaya implementasi, bingkai percakapan untuk memasukkan komponen pengembalian investasi (ROI) untuk membenarkan investasi waktu atau uang.
- Alokasikan biaya cloud dengan menerapkan showback atau chargeback untuk pengeluaran cloud, termasuk pengeluaran untuk opsi pembelian berbasis komitmen, layanan bersama, dan pembelian marketplace untuk mendukung penggunaan cloud yang paling sadar biaya.
- Sertakan pelatihan tentang kesadaran biaya dalam program pelatihan dan pengembangan yang sudah ada di seluruh organisasi Anda. Disarankan untuk menyertakan pelatihan dan sertifikasi berkelanjutan di dalamnya. Hal ini akan menghasilkan sebuah organisasi yang mampu mengelola biaya dan penggunaan secara mandiri.
- Manfaatkan alat AWS asli gratis seperti AWS Cost Anomaly Detection, AWS Budgets, dan AWS Budgets Laporan.

Ketika organisasi secara konsisten mengadopsi praktik Cloud Financial Management (CFM), perilaku tersebut menjadi tertanam dalam cara kerja dan pengambilan keputusan. Hasilnya adalah budaya yang lebih sadar biaya, mulai dari pengembang yang merancang born-in-the-cloud aplikasi baru, hingga manajer keuangan yang menganalisis investasi cloud ROI baru ini.

Langkah-langkah implementasi

- Identifikasi proses-proses organisasi yang relevan: Setiap unit organisasi meninjau proses-proses mereka dan mengidentifikasi proses yang berdampak pada biaya dan penggunaan. Proses apa pun yang mengakibatkan pembuatan atau penghentian sumber daya perlu turut ditinjau. Cari proses yang dapat mendukung kesadaran biaya di dalam bisnis Anda, seperti manajemen insiden dan pelatihan insiden.
- Membangun budaya sadar biaya mandiri: Pastikan semua pemangku kepentingan yang relevan selaras dengan cause-of-change dan berdampak sebagai biaya sehingga mereka memahami biaya cloud. Ini akan memungkinkan organisasi Anda membangun budaya inovasi swadaya dan sadar biaya.

- Perbarui proses dengan kesadaran biaya: Setiap proses dimodifikasi agar menjadi sadar biaya. Proses mungkin memerlukan pemeriksaan awal tambahan, seperti penilaian dampak biaya, atau pemeriksaan akhir yang memvalidasi terjadinya perubahan yang diharapkan dalam hal biaya dan penggunaan. Proses pendukung seperti pelatihan dan manajemen insiden dapat dibuat agar menyertakan item-item untuk biaya dan penggunaan.

Untuk mendapatkan bantuan, hubungi CFM pakar melalui tim Akun Anda, atau jelajahi sumber daya dan dokumen terkait di bawah ini.

Sumber daya

Dokumen terkait:

- AWS Manajemen Keuangan Cloud

Contoh terkait:

- Strategi untuk Manajemen Penghematan Biaya Cloud
- Seri Blog Pengendalian Biaya #3: Cara Mengatasi Lonjakan Biaya
- Panduan Pemula untuk AWS Cost Management

COST01-BP05 Laporkan dan beri tahu tentang optimalisasi biaya

Siapkan anggaran cloud dan konfigurasikan mekanisme untuk mendeteksi anomali dalam penggunaan. Konfigurasikan alat-alat terkait untuk peringatan biaya dan penggunaan sesuai target yang telah ditentukan sebelumnya dan terima notifikasi ketika terdapat penggunaan yang melebihi target tersebut. Lakukan pertemuan rutin untuk menganalisis efektivitas biaya beban kerja Anda dan meningkatkan kesadaran biaya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Anda harus secara rutin melaporkan pengoptimalan biaya dan penggunaan di dalam organisasi Anda. Anda dapat mengimplementasikan sesi-sesi khusus untuk membahas kinerja biaya, atau sertakan pengoptimalan biaya di dalam siklus pelaporan operasional rutin untuk beban kerja Anda. Gunakan layanan dan alat untuk memantau kinerja biaya Anda secara teratur dan menerapkan peluang penghematan biaya.

Periksa biaya dan penggunaan Anda dengan berbagai filter dan informasi terperinci dengan menggunakan AWS Cost Explorer, yang menyediakan dasbor dan laporan seperti biaya berdasarkan layanan atau akun, biaya harian, atau biaya pasaran. Lacak kemajuan biaya dan penggunaan Anda dibandingkan dengan anggaran yang telah ditetapkan menggunakan Laporan AWS Budgets.

Gunakan AWS Budgetsuntuk mengatur anggaran khusus untuk melacak biaya dan penggunaan Anda dan merespons dengan cepat peringatan yang diterima dari email atau pemberitahuan Amazon Simple Notification Service SNS (Amazon) jika Anda melebihi ambang batas Anda. Tetapkan periode anggaran pilihan Anda menjadi harian, bulanan, triwulanan, atau tahunan, dan buat batasan anggaran khusus untuk tetap mendapat informasi tentang bagaimana biaya aktual atau perkiraan dan kemajuan penggunaan menuju ambang anggaran Anda. Anda juga dapat mengatur peringatan dan tindakan atas pemberitahuan tersebut agar berjalan secara otomatis atau melalui proses persetujuan saat target anggaran terlampaui.

Menerapkan pemberitahuan tentang biaya dan penggunaan untuk memastikan bahwa perubahan biaya dan penggunaan dapat ditindaklanjuti dengan cepat jika tidak terduga. AWS Cost Anomaly Detectionmemungkinkan Anda mengurangi kejutan biaya dan meningkatkan kontrol tanpa memperlambat inovasi. AWS Cost Anomaly Detection mengidentifikasi pengeluaran anomali dan akar penyebab, yang membantu mengurangi risiko kejutan penagihan. Dengan tiga langkah mudah, Anda dapat membuat pemantauan kontekstual Anda sendiri dan menerima peringatan saat ada anomali pengeluaran yang terdeteksi.

Anda juga dapat menggunakan Amazon QuickSight with AWS Cost and Usage Report (CUR) data, untuk menyediakan pelaporan yang sangat disesuaikan dengan data yang lebih terperinci. Amazon QuickSight memungkinkan Anda menjadwalkan laporan dan menerima email Laporan Biaya berkala untuk biaya historis dan penggunaan atau peluang penghematan biaya. Periksa solusi Cost Intelligence Dashboard (CID) kami yang dibangun di Amazon QuickSight, yang memberi Anda visibilitas tingkat lanjut.

Gunakan AWS Trusted Advisor, yang memberikan panduan untuk memverifikasi apakah sumber daya yang disediakan selaras dengan praktik AWS terbaik untuk pengoptimalan biaya.

Lihat rekomendasi Savings Plans Anda melalui grafik visual berdasarkan biaya dan penggunaan mendetail Anda. Grafik per jam menunjukkan pengeluaran Sesuai Permintaan beserta komitmen Savings Plans yang disarankan, sehingga memberikan wawasan tentang perkiraan penghematan, cakupan Savings Plans, dan pemanfaatan Savings Plans. Ini membantu organisasi untuk memahami bagaimana Savings Plans mereka berlaku untuk setiap jam pengeluaran tanpa harus menginvestasikan waktu dan sumber daya untuk membangun model-model analisis pengeluaran.

Buat laporan secara berkala yang berisi sorotan rekomendasi Savings Plans, Reserved Instances, dan Amazon EC2 rightsizing mulai mengurangi biaya yang terkait dengan AWS Cost Explorer beban kerja kondisi mapan, idle, dan sumber daya yang kurang dimanfaatkan. Identifikasi dan dapatkan kembali pengeluaran yang terkait dengan pemborosan cloud untuk sumber daya yang di-deploy. Pemborosan cloud terjadi saat sumber daya berukuran tidak tepat dibuat atau ditemukan pola penggunaan yang berbeda dari yang diharapkan. Ikuti praktik AWS terbaik untuk mengurangi pemborosan Anda atau minta tim akun dan mitra Anda untuk membantu Anda mengoptimalkan dan menghemat biaya cloud Anda.

Buat laporan secara teratur untuk opsi pembelian yang lebih baik bagi sumber daya Anda guna menurunkan biaya unit untuk beban kerja Anda. Opsi pembelian seperti Savings Plans, Reserved Instances, atau Amazon EC2 Spot Instances menawarkan penghematan biaya terdalam untuk beban kerja yang toleran terhadap kesalahan dan memungkinkan pemangku kepentingan (pemilik bisnis, keuangan, dan tim teknologi) untuk menjadi bagian dari diskusi komitmen ini.

Bagikan laporan yang berisi peluang atau pengumuman rilis baru yang dapat membantu Anda mengurangi total biaya kepemilikan (TCO) cloud. Terapkan layanan baru, Wilayah, fitur, solusi, atau cara baru untuk mencapai pengurangan biaya lebih lanjut.

Langkah-langkah implementasi

- Konfigurasi AWS Budgets: Konfigurasikan AWS Budgets pada semua akun untuk beban kerja Anda. Tetapkan anggaran untuk keseluruhan pengeluaran akun, serta anggaran untuk beban kerja menggunakan tag.
  - Lab Well-Architected: Penggunaan Biaya dan Tata Kelola
- Laporan mengenai Optimalisasi Biaya: Siapkan siklus rutin untuk membahas dan menganalisis efisiensi beban kerja. Menggunakan metrik yang telah dibangun, buat laporan tentang metrik-metrik yang dicapai serta biaya untuk mencapainya. Identifikasi dan perbaiki tren negatif apa pun, serta tren positif yang dapat Anda pupuk di seluruh organisasi Anda. Pelaporan harus melibatkan perwakilan dari tim dan pemilik aplikasi, keuangan, dan pengambil keputusan utama sehubungan dengan pengeluaran cloud.

Sumber daya

Dokumen terkait:

- AWS Cost Explorer
- AWS Trusted Advisor

- AWS Budgets
- AWS Cost and Usage Report
- AWS Budgets Praktik Terbaik
- Amazon Analitik S3

Contoh terkait:

- Lab Well-Architected: Penggunaan Biaya dan Tata Kelola
- Cara utama untuk mulai mengoptimalkan biaya AWS cloud Anda

COST01-BP06 Monitor biaya proaktif

Gunakan alat dan dasbor untuk memantau biaya beban kerja secara proaktif. Tinjau biaya secara teratur dengan alat yang dikonfigurasi atau alat siap pakai, jangan hanya memeriksa biaya dan kategori saat Anda menerima notifikasinya. Memantau dan menganalisis biaya secara proaktif akan membantu mengidentifikasi tren positif dan memungkinkan Anda mempromosikan tren tersebut ke seluruh organisasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sebaiknya pantau biaya dan penggunaan di organisasi Anda secara proaktif, tidak hanya saat terdapat pengecualian atau anomali. Dasbor yang dapat dilihat dari kantor atau lingkungan kerja Anda memastikan orang yang memiliki peran penting dapat mengakses informasi yang diperlukan, serta menunjukkan bahwa organisasi Anda fokus pada pengoptimalan biaya. Dengan dasbor yang terlihat ini, Anda dapat mendorong hasil secara aktif dan menerapkannya di seluruh organisasi Anda.

Buat rutinitas harian atau sering untuk digunakan AWS Cost Exploreratau dasbor lain seperti Amazon QuickSight untuk melihat biaya dan menganalisis secara proaktif. Analisis penggunaan dan biaya AWS layanan di tingkat AWS akun, tingkat beban kerja, atau tingkat AWS layanan tertentu dengan pengelompokan dan penyaringan, dan validasi apakah mereka diharapkan atau tidak. Gunakan perincian dan tag tingkat sumber daya dan per jam untuk memfilter dan mengidentifikasi biaya yang timbul untuk sumber daya teratas. Anda juga dapat membuat laporan Anda sendiri dengan Dasbor Kecerdasan Biaya, QuickSight solusi Amazon yang dibuat oleh AWS Solutions Architects, dan membandingkan anggaran Anda dengan biaya dan penggunaan aktual.

Langkah-langkah implementasi

- Laporan mengenai Optimalisasi Biaya: Siapkan siklus rutin untuk membahas dan menganalisis efisiensi beban kerja. Menggunakan metrik yang telah dibangun, buat laporan tentang metrik-metrik yang dicapai serta biaya untuk mencapainya. Identifikasikan dan perbaiki tren negatif apa pun, serta identifikasi tren positif yang dapat Anda pupuk di seluruh organisasi Anda. Pelaporan harus melibatkan perwakilan dari tim dan pemilik aplikasi, keuangan, dan manajemen.
- Buat dan aktifkan perincian harian untuk biaya dan penggunaan AWS Budgetsuntuk mengambil tindakan tepat waktu guna mencegah potensi kelebihan biaya: AWS Budgets memungkinkan Anda mengonfigurasi pemberitahuan peringatan, sehingga Anda tetap mendapat informasi jika ada jenis anggaran Anda yang keluar dari ambang batas yang telah dikonfigurasi sebelumnya. Cara terbaik untuk memanfaatkan AWS Budgets adalah dengan menetapkan biaya dan penggunaan yang Anda harapkan sebagai batas Anda, sehingga apa pun di atas anggaran Anda dapat dianggap terlalu banyak.
- Create AWS Cost Anomaly Detection for cost monitor: AWS Cost Anomaly Detectionmenggunakan teknologi Machine Learning canggih untuk mengidentifikasi pengeluaran anomali dan akar penyebab, sehingga Anda dapat dengan cepat mengambil tindakan. Hal ini akan memungkinkan Anda mengonfigurasi pemantauan biaya yang menentukan segmen pengeluaran yang ingin Anda evaluasi (misalnya, tiap-tiap layanan AWS , akun anggota, tag alokasi biaya, dan kategori biaya), dan memungkinkan Anda mengatur kapan, di mana, dan bagaimana Anda menerima notifikasi peringatan. Untuk setiap pemantauan, kaitkan beberapa langganan peringatan untuk pemilik bisnis dan tim teknologi, termasuk nama, ambang batas dampak biaya, dan frekuensi peringatan (tiap-tiap peringatan, ringkasan harian, ringkasan mingguan) untuk setiap langganan.
- Menggunakan AWS Cost Explorer atau mengintegrasikan data AWS Cost and Usage Report (CUR) Anda dengan QuickSight dasbor Amazon untuk memvisualisasikan biaya organisasi Anda: AWS Cost Explorer memiliki easy-to-use antarmuka yang memungkinkan Anda memvisualisasikan, memahami, dan mengelola AWS biaya dan penggunaan Anda dari waktu ke waktu. Dasbor Kecerdasan Biaya adalah dasbor yang dapat disesuaikan dan dapat diakses untuk membantu Anda membuat landasan manajemen biaya dan alat optimalisasi Anda sendiri.

Sumber daya

Dokumen terkait:

- AWS Budgets
- AWS Cost Explorer
- Anggaran Biaya dan Penggunaan Harian
- AWS Cost Anomaly Detection

Contoh terkait:

- Lab Well-Architected: Visualisasi
- Lab Well-Architected: Visualisasi Tingkat Lanjut
- Lab Well-Architected: Dasbor Kecerdasan Cloud
- Lab Well-Architected: Visualisasi Biaya
- AWS Cost Anomaly Detection Waspada dengan Slack

COST01-BP07 Tetap up-to-date dengan rilis layanan baru

Konsultasikan secara teratur dengan para ahli atau AWS Mitra untuk mempertimbangkan layanan dan fitur mana yang memberikan biaya lebih rendah. Tinjau AWS blog dan sumber informasi lainnya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS terus menambahkan kemampuan baru sehingga Anda dapat memanfaatkan teknologi terbaru untuk bereksperimen dan berinovasi lebih cepat. Anda mungkin dapat menerapkan AWS layanan dan fitur baru untuk meningkatkan efisiensi biaya dalam beban kerja Anda. Lakukan peninjauan secara teratur Manajemen Biaya AWS, Blog Berita AWS, Blog Manajemen Biaya AWS, dan Apa yang Baru di AWS untuk informasi tentang rilisan layanan dan fitur-fitur baru. What's New post memberikan gambaran singkat tentang semua AWS layanan, fitur, dan pengumuman perluasan Wilayah saat dirilis.

Langkah-langkah implementasi

- Berlangganan blog: Buka halaman AWS blog dan berlangganan Blog Apa yang Baru dan blog lain yang relevan. Anda dapat mendaftar di halaman preferensi komunikasi dengan alamat email Anda.
- Berlangganan AWS Berita: Tinjau Blog AWS Berita dan Apa yang Baru secara teratur AWS untuk informasi tentang layanan baru dan rilis fitur. Berlangganan RSS feed, atau dengan email Anda untuk mengikuti pengumuman dan rilis.
- Ikuti Pengurangan AWS Harga: Pemotongan harga reguler pada semua layanan kami telah menjadi cara standar AWS untuk meneruskan efisiensi ekonomi kepada pelanggan kami yang diperoleh dari skala kami. Per 20 September 2023, AWS telah menurunkan harga 134 kali sejak 2006. Jika Anda memiliki keputusan bisnis yang tertunda karena masalah harga, Anda dapat meninjaunya kembali setelah pengurangan harga dan integrasi layanan baru. Anda dapat

mempelajari tentang upaya pengurangan harga sebelumnya, termasuk instans Amazon Elastic Compute Cloud EC2 (Amazon), dalam kategori pengurangan harga di Blog Berita. AWS

- AWS acara dan pertemuan: Hadiri AWS pertemuan puncak lokal Anda, dan setiap pertemuan lokal dengan organisasi lain dari daerah Anda. Jika Anda tidak dapat hadir secara langsung, cobalah untuk menghadiri acara virtual untuk mendengar lebih banyak dari AWS para ahli dan kasus bisnis pelanggan lain.
- Temui tim akun Anda: Jadwalkan pertemuan rutin dengan tim akun Anda, adakan pertemuan lalu diskusikan tren industri dan layanan AWS . Bicarakan dengan manajer akun, Arsitek Solusi, dan tim dukungan Anda.

Sumber daya

Dokumen terkait:

- AWS Manajemen Biaya
- Apa yang baru dengan AWS
- AWS Blog Berita

Contoh terkait:

- Amazon EC2 — 15 Tahun Mengoptimalkan dan Menghemat Biaya TI Anda
- AWS Blog Berita - Pengurangan Harga

COST01-BP08 Ciptakan budaya sadar biaya

Implementasikan perubahan atau program untuk menciptakan budaya sadar biaya di seluruh organisasi. Sebaiknya mulai dari cakupan kecil. Kemudian, seiring meningkatnya kemampuan dan penggunaan cloud oleh organisasi Anda, implementasikan program dengan cakupan yang lebih luas dan besar.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Budaya sadar biaya akan memungkinkan Anda menskalakan pengoptimalan biaya dan Manajemen Finansial Cloud (tim operasi keuangan, pusat keunggulan cloud, operasi cloud, dan sebagainya) melalui praktik terbaik yang dilakukan secara organik dan terdesentralisasi di seluruh organisasi

Anda. Kesadaran biaya ini akan memungkinkan Anda untuk membuat kemampuan tingkat tinggi di seluruh organisasi dengan upaya yang minimal, dibandingkan dengan pendekatan tersentralisasi dari atas ke bawah yang kaku.

Menghadirkan kesadaran biaya dalam komputasi cloud, terutama untuk pendorong biaya utama dalam komputasi cloud, memungkinkan tim memahami hasil yang diharapkan untuk setiap perubahan dalam perspektif biaya. Tim yang mengakses lingkungan cloud harus mengetahui model harga dan perbedaan antara pusat data on-premise tradisional dan komputasi cloud.

Manfaat utama budaya sadar biaya adalah bahwa tim teknologi mengoptimalkan biaya secara proaktif dan secara kontinu (misalnya, biaya tersebut dianggap sebagai persyaratan nonfungsional saat merancang beban kerja baru atau membuat perubahan pada beban kerja yang ada), bukan melakukan pengoptimalan biaya reaktif seperlunya.

Perubahan kecil dalam budaya bisa berdampak besar terhadap efisiensi beban kerja saat ini dan masa mendatang. Contohnya adalah:

- Memberikan visibilitas dan memunculkan kesadaran dalam tim rekayasa untuk memahami fungsi dan dampak budaya tersebut dari segi biaya.
- Menerapkan mekanisme yang kompetitif pada biaya dan penggunaan di seluruh organisasi. Ini dapat dilakukan melalui dasbor yang terlihat publik, atau laporan yang membandingkan biaya dan penggunaan yang dinormalisasi di seluruh tim (misalnya, cost-per-workload dan). cost-per-transaction
- Penghargaan atas efisiensi biaya. Berikan penghargaan atas pencapaian optimasi biaya yang dilakukan dengan sukarela atau tanpa diminta baik secara publik maupun privat, dan belajar dari kesalahan agar tidak mengulanginya di masa depan.
- Membuat persyaratan organisasi dari atas ke bawah untuk beban kerja yang dijalankan dengan anggaran yang telah ditentukan sebelumnya.
- Mempertanyakan kebutuhan bisnis yang akan dipenuhi oleh perubahan, dan dampak biaya dari perubahan yang diminta pada infrastruktur arsitektur atau konfigurasi beban kerja untuk memastikan Anda hanya membayar sesuai kebutuhan.
- Memastikan perencana perubahan mengetahui perubahan yang diharapkan yang memiliki dampak biaya, dan bahwa perubahan tersebut dikonfirmasi oleh pemangku kepentingan untuk memberikan hasil bisnis dengan biaya yang efektif.

Langkah-langkah implementasi

- Laporkan biaya cloud ke tim teknologi: Untuk meningkatkan kesadaran biaya, dan membangun efisiensi KPIs bagi pemangku kepentingan keuangan dan bisnis.
- Beritahukan para pemangku jabatan atau para anggota tim tentang perubahan yang sudah terencana: Buat item agenda untuk membahas perubahan yang direncanakan dan dampak manfaat biaya pada beban kerja selama rapat perubahan mingguan.
- Temui tim akun Anda: Jadwalkan pertemuan rutin dengan tim akun Anda, lalu diskusikan tren industri dan layanan AWS . Bicarakan dengan manajer akun, arsitek, dan tim dukungan Anda.
- Bagikan kisah sukses: Bagikan kisah sukses tentang pengurangan biaya untuk beban kerja apa pun, Akun AWS, atau organisasi untuk menciptakan sikap dan dorongan positif seputar pengoptimalan biaya.
- Pelatihan: Pastikan tim teknis atau anggota tim dilatih untuk mengetahui biaya sumber daya AWS Cloud.
- AWS acara dan pertemuan: Hadiri AWS KTT lokal, dan setiap pertemuan lokal dengan organisasi lain dari daerah Anda.
- Berlangganan blog: Buka halaman AWS blog dan berlangganan Blog Baru Apa dan blog relevan lainnya untuk mengikuti rilis, implementasi, contoh, dan perubahan baru yang dibagikan oleh AWS.

Sumber daya

Dokumen terkait:

- AWS Blog
- AWS Manajemen Biaya
- AWS Blog Berita

Contoh terkait:

- AWS Manajemen Keuangan Cloud
- AWS Well-Architected Labs: Manajemen Keuangan Cloud

COST01-BP09 Mengukur nilai bisnis dari optimalisasi biaya

Penghitungan nilai bisnis dari optimalisasi biaya memungkinkan Anda memahami seluruh rangkaian keuntungan untuk organisasi Anda. Karena optimasi biaya adalah investasi yang diperlukan, penghitungan nilai bisnis memungkinkan Anda untuk menjelaskan laba atas investasi kepada

pemangku kepentingan. Penghitungan nilai bisnis dapat membantu Anda memperoleh lebih banyak dukungan dari pemangku kepentingan untuk investasi optimasi biaya mendatang, dan menyediakan kerangka kerja untuk mengukur hasil bagi aktivitas optimasi biaya organisasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Menghitung nilai bisnis berarti mengukur manfaat yang diperoleh bisnis dari tindakan dan keputusan yang mereka ambil. Nilai bisnis dapat berwujud (seperti berkurangnya biaya atau meningkatnya keuntungan) atau tidak berwujud (seperti meningkatnya reputasi merek atau meningkatnya kepuasan pelanggan).

Menghitung nilai bisnis dari pengoptimalan biaya berarti menentukan seberapa besar nilai atau manfaat yang Anda dapatkan dari upaya Anda mengefisiensikan pengeluaran. Misalnya, jika sebuah perusahaan menghabiskan $100.000 untuk menyebarkan beban kerja AWS dan kemudian mengoptimalkannya, biaya baru menjadi hanya $80.000 tanpa mengorbankan kualitas atau output. Dalam skenario ini, nilai bisnis yang terhitung dari pengoptimalan biaya adalah penghematan sebesar 20.000 USD. Namun selain penghematan, bisnis mungkin juga menghitung nilai dalam hal waktu pengiriman yang lebih cepat, kepuasan pelanggan yang meningkat, atau metrik lain yang dihasilkan dari upaya pengoptimalan biaya. Pemangku kepentingan perlu membuat keputusan tentang potensi nilai pengoptimalan biaya, biaya untuk mengoptimalkan beban kerja, dan nilai labanya.

Selain melaporkan penghematan dari optimasi biaya, Anda direkomendasikan untuk menghitung nilai tambah yang dihadirkan. Keuntungan optimasi biaya biasanya dihitung dari biaya terendah per hasil bisnis. Misalnya, Anda dapat mengukur penghematan biaya EC2 ( Amazon Elastic Compute Cloud Amazon) saat membeli Savings Plans, yang mengurangi biaya dan mempertahankan tingkat output beban kerja. Anda dapat mengukur pengurangan biaya dalam AWS pengeluaran saat EC2 instans Amazon yang tidak aktif dihapus, atau volume Amazon Elastic Block Store (Amazon) yang tidak terpasang dihapus. EBS

Namun, keuntungan dari optimasi biaya bukan sekadar pengurangan atau peniadaan biaya. Pertimbangkan pengambilan data tambahan untuk mengukur peningkatan efisiensi dan nilai bisnis.

Langkah-langkah implementasi

- Mengevaluasi manfaat bisnis: Ini adalah proses menganalisis dan menyesuaikan AWS Cloud biaya dengan cara yang memaksimalkan manfaat yang diterima dari setiap dolar yang dihabiskan. Alih-alih berfokus pada pengurangan biaya tanpa nilai bisnis, pertimbangkan manfaat bisnis dan laba atas investasi untuk pengoptimalan biaya, yang dapat membawa lebih banyak nilai dari uang yang

Anda keluarkan. Intinya adalah mengeluarkan uang dengan bijak dan melakukan investasi dan pengeluaran di area yang paling banyak menghasilkan laba.

- Analisis AWS biaya peramalan: Peramalan membantu membiayai pemangku kepentingan menetapkan harapan dengan pemangku kepentingan organisasi internal dan eksternal lainnya, dan dapat meningkatkan prediktabilitas keuangan organisasi Anda. AWS Cost Explorerdapat digunakan untuk melakukan peramalan untuk biaya dan penggunaan Anda.

Sumber daya

Dokumen terkait:

- AWS Cloud Ekonomi
- AWS Blog
- AWS Manajemen Biaya
- AWS Blog Berita
- Laporan resmi Pilar Keandalan Well-Architected
- AWS Cost Explorer

Video terkait:

- Buka Kunci Nilai Bisnis dengan Windows aktif AWS

Contoh terkait:

- Mengukur dan Memaksimalkan Nilai Bisnis Customer 360
- Nilai Bisnis Dengan Mengadopsi Database yang Dikelola Layanan Web Amazon
- Nilai Bisnis dari Layanan Web Amazon untuk Vendor Software Mandiri
- Nilai Bisnis dari Modernisasi Cloud
- Nilai Bisnis dari Migrasi ke Layanan Web Amazon

## Kesadaran akan penggunaan dan pengeluaran

Pertanyaan

- COST2. Bagaimana cara mengatur penggunaan?

- COST3. Bagaimana cara memantau biaya dan penggunaan Anda?
- COST4. Bagaimana cara melakukan penonaktifan sumber daya?

## COST2. Bagaimana cara mengatur penggunaan?

Tetapkan kebijakan dan mekanisme untuk memverifikasi bahwa biaya yang ditimbulkan memang diperlukan untuk mencapai tujuan. Dengan menggunakan checks-and-balances pendekatan, Anda dapat berinovasi tanpa pengeluaran berlebihan.

Praktik terbaik

- COST02-BP01 Mengembangkan kebijakan berdasarkan kebutuhan organisasi Anda
- COST02-BP02 Melaksanakan tujuan dan sasaran
- COST02-BP03 Menerapkan struktur akun
- COST02-BP04 Melaksanakan kelompok dan peran
- COST02-BP05 Menerapkan kontrol biaya
- COST02-BP06 Lacak siklus hidup proyek

COST02-BP01 Mengembangkan kebijakan berdasarkan kebutuhan organisasi Anda

Kembangkan kebijakan yang menentukan cara organisasi Anda mengelola sumber daya dan periksa sumber daya secara berkala. Kebijakan harus mencakup aspek biaya sumber daya dan beban kerja, termasuk pembuatan, pengubahan, dan penonaktifan selama masa pakai sumber daya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Memahami pendorong dan biaya organisasi Anda sangat penting untuk mengelola biaya dan penggunaan Anda secara efektif, serta untuk mengenali peluang penghematan biaya. Umumnya organisasi mengoperasikan beberapa beban kerja yang dijalankan oleh beberapa tim. Tim-tim ini dapat berada dalam unit organisasi yang berlainan, masing-masing dengan aliran pendapatannya sendiri. Kemampuan untuk mengaitkan biaya sumber daya dengan beban kerja, tiap-tiap organisasi, atau pemilik produk mendorong perilaku penggunaan yang efisien dan membantu mengurangi pemborosan. Pemantauan biaya dan penggunaan yang akurat membantu Anda memahami seberapa optimal suatu beban kerja, serta seberapa menguntungkan produk dan unit organisasi. Pengetahuan ini memungkinkan pengambilan keputusan yang lebih tepat tentang target

pengalokasian sumber daya di dalam organisasi Anda. Kesadaran Anda akan penggunaan di semua tingkatan dalam organisasi merupakan kunci yang akan mendorong perubahan, karena perubahan dalam penggunaan akan mendorong perubahan dalam biaya. Pertimbangkan untuk mengambil pendekatan dengan beberapa aspek untuk menyadari penggunaan dan pengeluaran Anda.

Langkah pertama dalam menjalankan tata kelola adalah dengan menggunakan keperluan organisasi Anda untuk mengembangkan kebijakan penggunaan cloud Anda. Kebijakan tersebut menentukan cara organisasi Anda menggunakan cloud dan cara mengelola sumber daya. Kebijakan harus mencakup semua aspek dalam sumber daya dan beban kerja yang berkaitan dengan biaya dan penggunaan, termasuk pembuatan, pengubahan, serta penonaktifan selama masa pakai sumber daya. Verifikasikan bahwa kebijakan dan prosedur diikuti dan diterapkan untuk setiap perubahan di lingkungan cloud. Selama rapat manajemen perubahan IT Anda, ajukan pertanyaan untuk mengetahui dampak biaya dari perubahan yang direncanakan apakah meningkat atau menurun, justifikasi bisnis, dan hasil yang diharapkan.

Kebijakan harus dibuat sederhana agar dapat dengan mudah dipahami dan diimplementasikan secara efektif di seluruh organisasi. Kebijakan juga harus mudah diikuti dan ditafsirkan (sehingga digunakan) serta spesifik (tidak ada salah tafsir antar tim). Selain itu, kebijakan perlu diperiksa secara berkala (seperti mekanisme kami) dan diperbarui seiring perubahan kondisi atau prioritas bisnis pelanggan yang dapat menjadikan kebijakan usang.

Mulailah dengan kebijakan tingkat tinggi yang sangat umum, seperti Wilayah geografis yang digunakan atau waktu dalam sehari saat sumber daya harus dijalankan. Sempurnakan kebijakan-kebijakan tersebut secara bertahap untuk unit dan beban kerja organisasi yang beragam. Kebijakan yang umum mencakup layanan dan fitur yang dapat digunakan (misalnya, penyimpanan yang beperforma lebih rendah di lingkungan pengujian dan pengembangan), jenis sumber daya yang dapat digunakan oleh grup yang berbeda-beda (misalnya, ukuran sumber daya terbesar dalam akun pengembangan adalah ukuran sedang), dan jangka waktu sumber daya tersebut akan digunakan (sementara, jangka pendek, atau untuk jangka waktu tertentu).

Contoh kebijakan

Berikut ini adalah contoh kebijakan yang dapat Anda kaji untuk membuat kebijakan tata kelola cloud Anda sendiri, yang berfokus pada pengoptimalan biaya. Pastikan Anda menyesuaikan kebijakan berdasarkan kebutuhan organisasi dan permintaan pemangku kepentingan Anda.

- Nama kebijakan: Tentukan nama kebijakan yang jelas, seperti Kebijakan Optimalisasi Sumber Daya dan Penurunan Biaya.

- Tujuan: Jelaskan alasan kebijakan ini harus digunakan serta hasil yang diharapkan. Tujuan dari kebijakan ini adalah untuk memverifikasi bahwa diperlukan biaya yang minim untuk men-deploy dan menjalankan beban kerja yang diinginkan untuk memenuhi persyaratan bisnis.
- Lingkup: Tentukan dengan jelas siapa yang harus menggunakan kebijakan ini dan kapan kebijakan tersebut harus digunakan, seperti DevOps X Team untuk menggunakan kebijakan ini di pelanggan AS Timur untuk lingkungan X (produksi atau non-produksi).

Pernyataan kebijakan

1. Pilih us-east-1 atau beberapa wilayah us-east berdasarkan lingkungan beban kerja dan kebutuhan bisnis Anda (pengembangan, pengujian penerimaan pengguna, praproduksi, atau produksi).
2. Jadwalkan RDS instans Amazon EC2 dan Amazon untuk berjalan antara pukul enam pagi dan delapan malam (Waktu Standar Timur (EST)).
3. Hentikan semua EC2 instans Amazon yang tidak digunakan setelah delapan jam dan instans RDS Amazon yang tidak digunakan setelah 24 jam tidak aktif.
4. Hentikan semua EC2 instans Amazon yang tidak digunakan setelah 24 jam tidak aktif di lingkungan non-produksi. Ingatkan pemilik EC2 instans Amazon (berdasarkan tag) untuk meninjau EC2 instans Amazon yang dihentikan dalam produksi dan beri tahu mereka bahwa EC2 instans Amazon mereka akan dihentikan dalam waktu 72 jam jika tidak digunakan.
5. Gunakan keluarga dan ukuran instance generik seperti m5.large dan kemudian ubah ukuran instance berdasarkan CPU dan pemanfaatan memori menggunakan. AWS Compute Optimizer
6. Prioritaskan menggunakan penskalaan otomatis untuk menyesuaikan jumlah instans yang berjalan secara dinamis berdasarkan lalu lintas.
7. Gunakan instans spot untuk beban kerja non-kritis.
8. Kaji persyaratan kapasitas untuk memberikan komitmen untuk paket penyimpanan (savings plans) atau instans terpesan untuk beban kerja yang dapat diprediksi dan beritahukan kepada Tim Manajemen Keuangan Cloud.
9. Gunakan kebijakan siklus hidup Amazon S3 untuk memindahkan data yang jarang diakses ke tingkat penyimpanan yang lebih murah. Jika tidak ada kebijakan penyimpanan yang ditentukan, gunakan Amazon S3 Intelligent-Tiering untuk memindahkan objek ke tingkat yang diarsipkan secara otomatis.
10. Pantau pemanfaatan sumber daya dan setel alarm untuk memicu peristiwa penskalaan menggunakan Amazon. CloudWatch

11. Untuk masing-masing Akun AWS, gunakan AWS Budgets untuk menetapkan anggaran biaya dan penggunaan untuk akun Anda berdasarkan pusat biaya dan unit bisnis.
12. Menggunakan AWS Budgets untuk menetapkan anggaran biaya dan penggunaan untuk akun Anda dapat membantu Anda tetap di atas pengeluaran Anda dan menghindari tagihan yang tidak terduga, memungkinkan Anda untuk lebih mengontrol biaya Anda.

Prosedur: Berikan prosedur mendetail untuk menerapkan kebijakan ini atau beri referensi ke dokumen lain yang menjelaskan cara menerapkan setiap pernyataan kebijakan. Bagian ini harus memberikan step-by-step instruksi untuk melaksanakan persyaratan kebijakan.

Untuk menerapkan kebijakan ini, Anda dapat menggunakan berbagai alat atau AWS Config aturan pihak ketiga untuk memeriksa kepatuhan terhadap pernyataan kebijakan dan memicu tindakan remediasi otomatis menggunakan AWS Lambda fungsi. Anda juga dapat menggunakan AWS Organizations untuk menegakkan kebijakan. Selain itu, Anda harus secara rutin meninjau penggunaan sumber daya Anda dan menyesuaikan kebijakan apabila diperlukan untuk memastikan kebijakan tersebut terus memenuhi kebutuhan bisnis Anda.

Langkah-langkah implementasi

- Bertemulah dengan pemangku kepentingan: Untuk mengembangkan kebijakan, mintalah pemangku kepentingan (kantor bisnis cloud, rekayasawan, atau pengambil keputusan fungsional untuk penegakan kebijakan) di dalam organisasi Anda untuk menentukan kebutuhan mereka dan mendokumentasikannya. Gunakan pendekatan berulang dengan memulai dari hal paling umum dan menyempurnakannya secara berkelanjutan hingga ke unit terkecil di setiap langkahnya. Anggota tim yang terlibat adalah mereka yang berkepentingan langsung dengan beban kerja, seperti unit organisasi atau pemilik aplikasi, serta grup pendukung, seperti tim keamanan dan keuangan.
- Dapatkan konfirmasi: Pastikan semua tim menyetujui kebijakan terkait siapa yang dapat mengakses dan melakukan deployment ke AWS Cloud. Pastikan mereka mengikuti kebijakan organisasi Anda serta konfirmasikan bahwa pembuatan sumber daya mereka sejalan dengan kebijakan dan prosedur yang disetujui.
- Buat sesi pelatihan onboarding: Minta semua anggota organisasi baru untuk menyelesaikan kursus pelatihan orientasi untuk menciptakan kesadaran biaya dan membentuk persyaratan organisasi. Mereka boleh mempertimbangkan berbagai kebijakan dari pengalaman mereka sebelumnya atau tidak mempertimbangkannya sama sekali.

- Tentukan lokasi untuk beban kerja Anda: Tentukan lokasi pengoperasian beban kerja Anda, termasuk negara dan wilayah di dalam negara tersebut. Informasi ini digunakan untuk pemetaan ke Wilayah AWS dan Availability Zones.
- Tentukan dan kelompokkan layanan dan sumber daya: Tentukan layanan yang dibutuhkan beban kerja. Untuk setiap layanan, tentukan jenis, ukuran, dan jumlah sumber daya yang diperlukan. Tentukan grup sumber daya berdasarkan fungsi, seperti server aplikasi atau penyimpanan basis data. Sumber daya dapat dimiliki oleh beberapa grup.
- Tentukan dan kelompokkan pengguna sesuai fungsi:  Tentukan pengguna yang berinteraksi dengan beban kerja, dengan berfokus pada apa yang mereka kerjakan dan cara mereka menggunakan beban kerja, bukan pada posisi atau jabatan mereka di organisasi. Kelompokkan pengguna atau fungsi yang serupa menjadi satu. Anda dapat menggunakan kebijakan AWS terkelola sebagai panduan.
- Tentukan tindakan: Dengan lokasi, sumber daya, dan pengguna yang telah diidentifikasi di awal, tentukan tindakan yang diperlukan oleh masing-masing untuk meraih hasil beban kerja pada sepanjang masa pakainya (pengembangan, operasi, dan dekomisioning). Identifikasi tindakannya berdasarkan grup di setiap lokasi, bukan masing-masing elemen dalam grup. Mulailah dari hal yang umum dengan membaca atau menulis, kemudian sesuaikan tindakan-tindakan tertentu untuk setiap layanan.
- Tentukan periode peninjauan: Beban kerja dan persyaratan organisasi dapat berubah-ubah seiring waktu. Tentukan jadwal peninjauan beban kerja untuk memastikannya tetap selaras dengan prioritas organisasi.
- Buat dokumentasi kebijakan: Lakukan verifikasi bahwa kebijakan yang telah ditentukan dapat diakses saat dibutuhkan oleh organisasi Anda. Kebijakan-kebijakan ini digunakan untuk mengimplementasikan, memelihara, dan mengaudit akses lingkungan Anda.

Sumber daya

Dokumen terkait:

- Manajemen Perubahan di Cloud
- AWS Kebijakan Terkelola untuk Fungsi Job
- AWS strategi penagihan beberapa akun
- Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan
- AWS Manajemen dan Tata Kelola
- Kontrol akses untuk Wilayah AWS menggunakan IAM kebijakan

- Wilayah Infrastruktur Global dan AZs

Video terkait:

- AWS Manajemen dan Tata Kelola pada Skala

Contoh terkait:

- VMware- Apa itu Kebijakan Cloud?

COST02-BP02 Melaksanakan tujuan dan sasaran

Implementasikan sasaran serta target biaya dan penggunaan untuk beban kerja Anda. Sasaran memberikan arah untuk organisasi Anda tentang hasil yang diharapkan, dan target memberikan hasil terukur spesifik yang harus dicapai untuk beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Kembangkan sasaran serta target biaya dan penggunaan untuk organisasi Anda. Sebagai organisasi yang sedang berkembang AWS, penting untuk menetapkan dan melacak tujuan untuk pengoptimalan biaya. Sasaran atau indikator kinerja utama ini (KPIs) dapat mencakup hal-hal seperti persentase pengeluaran sesuai permintaan atau adopsi layanan tertentu yang dioptimalkan seperti instance AWS Graviton atau jenis volume gp3. EBS Tetapkan sasaran yang jelas dan dapat dicapai untuk membantu Anda mengukur peningkatan efisiensi yang penting bagi operasional bisnis Anda. Sasaran memberikan panduan dan arah kepada organisasi Anda terkait hasil yang diharapkan.

Target memberikan hasil yang jelas dan spesifik yang harus dicapai. Singkatnya, tujuan adalah arah yang ingin Anda tuju, dan target adalah seberapa jauh ke arah itu dan kapan tujuan itu harus dicapai (gunakan panduan spesifik, terukur, dapat ditetapkan, realistis dan tepat waktu, atau). SMART Contoh sasaran adalah penggunaan platform harus meningkat secara signifikan, hanya dengan peningkatan biaya yang minim (nonlinier). Contoh target adalah peningkatan penggunaan platform 20%, dengan peningkatan biaya kurang dari lima persen. Contoh sasaran umum lainnya adalah beban kerja harus lebih efisien setiap enam bulan. Target yang menyertainya adalah metrik biaya per bisnis harus turun lima persen setiap enam bulan. Gunakan metrik yang tepat, dan atur perhitungan KPIs untuk organisasi Anda. Anda bisa mulai dengan dasar KPIs dan berkembang nanti berdasarkan kebutuhan bisnis.

Sasaran untuk optimalisasi biaya adalah meningkatkan efisiensi beban kerja, yang sejalan dengan penurunan biaya per hasil bisnis untuk beban kerja tersebut dari waktu ke waktu. Implementasikan sasaran ini ke semua beban kerja, dan tetapkan target seperti peningkatan efisiensi sebesar lima persen setiap enam bulan hingga satu tahun. Di cloud, Anda dapat mencapainya melalui pembentukan kemampuan dalam optimalisasi biaya, serta rilis layanan dan fitur baru.

Target adalah tolok ukur yang jelas yang ingin Anda capai untuk memenuhi sasaran dan tolok ukur membandingkan hasil aktual Anda dengan target. Tetapkan tolok ukur dengan KPIs biaya per unit layanan komputasi (seperti adopsi Spot, adopsi Graviton, jenis instans terbaru, dan cakupan Sesuai Permintaan), layanan penyimpanan (seperti EBS GP3 adopsi, snapshot usang, dan penyimpanan standar Amazon S3)EBS, atau penggunaan layanan basis data (RDSseperti mesin sumber terbuka, adopsi Graviton, dan cakupan sesuai permintaan). Tolok ukur ini dan KPIs dapat membantu Anda memverifikasi bahwa Anda menggunakan AWS layanan dengan cara yang paling hemat biaya.

Tabel berikut menyediakan daftar AWS metrik standar untuk referensi. Setiap organisasi dapat memiliki nilai target yang berbeda untuk iniKPIs.

| Kategori | KPI (%) | Deskripsi |
| --- | --- | --- |
| Hitung | EC2Cakupan penggunaan | EC2contoh (dalam biaya atau jam) menggunakan SP+RI +Spot dibandingkan dengan total (dalam biaya atau jam) instance EC2 |
| Hitung | Komputasikan pemanfaatan SP/RI | Jam pemanfaatan SP atau RI akan dibandingkan dengan total jam SP atau RI yang tersedia |
| Hitung | EC2/Biaya jam | EC2biaya dibagi dengan jumlah EC2 instans yang berjalan pada jam itu |
| Hitung | v CPU biaya | Biaya per v CPU untuk semua instans |

| Kategori | KPI (%) | Deskripsi |
| --- | --- | --- |
| Hitung | Generasi Instans Terbaru | Persentase instans pada Graviton (atau jenis instans generasi modern lainnya) |
| Basis Data | RDScakupan | RDScontoh (dalam biaya atau jam) menggunakan RI dibandingkan dengan total (dalam biaya atau jam) instance RDS |
| Basis Data | RDSpemanfaatan | Jam pemanfaatan RI akan dibandingkan dengan total jam RI yang tersedia |
| Basis Data | RDSuptime | RDSbiaya dibagi dengan jumlah RDS instans yang berjalan pada jam itu |
| Basis Data | Generasi Instans Terbaru | Persentase instans pada Graviton (atau tipe instans modern lainnya) |
| Penyimpanan | Pemanfaatan penyimpanan | Biaya penyimpanan yang dioptimalkan (misalnya Glacier, deep archive, atau Infrequent Access) dibagi dengan total biaya penyimpan an |

| Kategori | KPI (%) | Deskripsi |
| --- | --- | --- |
| Penandaan | Pelepasan tag sumber daya | Cost Explorer<br><br>1. Memfilter kredit, diskon, pajak, pengembalian dana, pasar, dan salinan biaya bulanan terbaru<br><br>2. Pilih Tampilkan hanya sumber daya yang tidak diberi tag di Cost Explorer<br><br>3. Bagi jumlah dalam sumber daya yang tidak mendapat tag dengan biaya bulanan Anda. |

Menggunakan tabel ini, sertakan nilai target atau tolok ukur, yang harus dihitung berdasarkan sasaran organisasi Anda. Anda perlu mengukur metrik tertentu untuk bisnis Anda dan memahami hasil bisnis agar beban kerja tersebut dapat didefinisikan secara akurat dan realistis. KPIs Saat Anda mengevaluasi metrik kinerja di dalam organisasi, bedakan antara berbagai jenis metrik dengan tujuan berbeda-beda. Metrik-metrik tersebut terutama mengukur kinerja dan efisiensi infrastruktur teknis alih-alih mengukur secara langsung keseluruhan dampak bisnis. Misalnya, metrik mungkin melacak waktu respons server, latensi jaringan, atau waktu aktif sistem. Metrik ini sangat penting untuk menilai kinerja infrastruktur dalam mendukung operasional teknis organisasi. Namun, metrik tersebut tidak memberikan wawasan langsung tentang sasaran bisnis yang lebih luas seperti kepuasan pelanggan, pertumbuhan pendapatan, atau pangsa pasar. Untuk mendapatkan pemahaman yang komprehensif tentang kinerja bisnis, lengkapi metrik efisiensi ini dengan metrik bisnis strategis yang berkorelasi langsung dengan hasil bisnis.

Tetapkan visibilitas mendekati waktu nyata atas peluang tabungan Anda KPIs dan terkait dan lacak kemajuan Anda dari waktu ke waktu. Untuk memulai dengan definisi dan pelacakan KPI sasaran, kami merekomendasikan KPI dasbor dari Cloud Intelligence Dashboards (CID). Berdasarkan data dari Laporan Biaya dan Penggunaan (CUR), KPI dasbor menyediakan serangkaian pengoptimalan biaya yang direkomendasikanKPIs, dengan kemampuan untuk menetapkan tujuan khusus dan melacak kemajuan dari waktu ke waktu.

Jika Anda memiliki solusi lain untuk menetapkan dan melacak KPI tujuan, pastikan metode ini diadopsi oleh semua pemangku kepentingan manajemen keuangan cloud di organisasi Anda.

Langkah-langkah implementasi

- Tentukan tingkat penggunaan yang diharapkan: Untuk memulai, fokuslah pada tingkat penggunaan. Berinteraksilah dengan pemilik aplikasi, pemasaran, dan tim bisnis yang lebih besar untuk memahami tingkat penggunaan yang diharapkan untuk beban kerja. Bagaimana permintaan pelanggan mungkin berubah seiring waktu, dan apa yang dapat berubah dikarenakan peningkatan musiman atau kampanye pemasaran?
- Tentukan sumber daya dan biaya untuk menjalankan beban kerja: Setelah tingkat penggunaan ditetapkan, hitung perubahan pada sumber daya beban kerja yang diperlukan untuk memenuhi tingkat penggunaan ini. Anda mungkin perlu meningkatkan ukuran atau jumlah sumber daya untuk suatu komponen beban kerja, meningkatkan transfer data, atau mengubah komponen beban kerja ke layanan lain pada tingkat tertentu. Tentukan biaya yang akan dikeluarkan pada setiap poin utama ini, dan perkirakan perubahan biaya ketika terdapat perubahan pada penggunaan.
- Tentukan tujuan bisnis: Dengan mengambil output dari perkiraan perubahan penggunaan dan biaya, gabungkan dengan perkiraan perubahan dalam teknologi, atau program apa pun yang sedang Anda jalankan, dan kembangkan sasaran untuk beban kerja. Sasaran harus mencakup penggunaan dan biaya, serta hubungan di antara keduanya. Sasaran harus sederhana, berada pada tingkat tinggi, dan membantu orang memahami apa yang diharapkan bisnis dalam hal hasil (seperti memastikan sumber daya yang tidak terpakai dipertahankan di bawah tingkat biaya tertentu). Anda tidak perlu menentukan sasaran untuk setiap jenis sumber daya yang tidak terpakai atau menentukan biaya yang dapat menyebabkan kerugian pada sasaran dan target. Pastikan terdapat program organisasi (misalnya, pengembangan kemampuan seperti pelatihan dan pendidikan) jika terdapat perubahan yang diperkirakan pada biaya tanpa perubahan pada penggunaan.
- Tentukan sasaran: Untuk setiap sasaran yang telah ditetapkan, tentukan target yang terukur. Jika sasarannya adalah untuk meningkatkan efisiensi beban kerja, targetnya seharusnya adalah menghitung jumlah peningkatan (biasanya pada output bisnis untuk setiap dolar yang dikeluarkan) dan kapan peningkatan tersebut seharusnya tercapai. Misalnya, Anda dapat menetapkan sasaran untuk meminimalkan pemborosan karena kelebihan pasokan. Dengan sasaran ini, target Anda bisa jadi adalah pemborosan akibat kelebihan pasokan komputasi di tingkat pertama beban kerja produksi tidak boleh melebihi sepuluh persen dari biaya komputasi tingkat. Selain itu, target kedua bisa jadi adalah pemborosan akibat kelebihan pasokan komputasi di tingkat kedua beban kerja produksi tidak boleh melebihi lima persen dari biaya komputasi tingkat.

Sumber daya

Dokumen terkait:

- Kebijakan terkelola AWS untuk fungsi pekerjaan
- Strategi penagihan beberapa akun AWS
- Kontrol akses untuk Wilayah AWS menggunakan IAM kebijakan
- Tujuan S.M.A.R.T.
- Cara melacak pengoptimalan biaya Anda KPIs dengan CID KPI Dasbor

Video terkait:

- Lab Well-Architected: Tujuan dan Target (Tingkat 100)

Contoh terkait:

- Apa itu metrik satuan?
- Memilih metrik satuan untuk mendukung bisnis Anda
- Metrik unit dalam praktik – pelajaran yang dipetik
- Bagaimana metrik satuan membantu menciptakan keselarasan antar fungsi bisnis
- Lab Well-Architected: Menonaktifkan sumber daya (Sasaran dan Target)
- Lab Well-Architected: Jenis, Ukuran, dan Jumlah Sumber Daya (Sasaran dan Target)

COST02-BP03 Menerapkan struktur akun

Implementasikan struktur akun yang dipetakan ke organisasi Anda. Hal ini dapat membantu alokasi dan pengelolaan biaya di organisasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS Organizations memungkinkan Anda membuat beberapa Akun AWS yang dapat membantu Anda mengatur lingkungan secara terpusat saat Anda meningkatkan beban kerja Anda. AWS Anda dapat memodelkan hierarki organisasi Anda dengan mengelompokkan Akun AWS dalam struktur unit

organisasi (OU) dan membuat beberapa Akun AWS di bawah setiap OU. Untuk membuat struktur akun, Anda perlu memutuskan mana dari Akun AWS Anda yang akan menjadi akun manajemen terlebih dahulu. Setelah itu, Anda dapat membuat akun baru Akun AWS atau memilih akun yang sudah ada sebagai akun anggota berdasarkan struktur akun yang dirancang dengan mengikuti praktik terbaik akun manajemen dan praktik terbaik akun anggota.

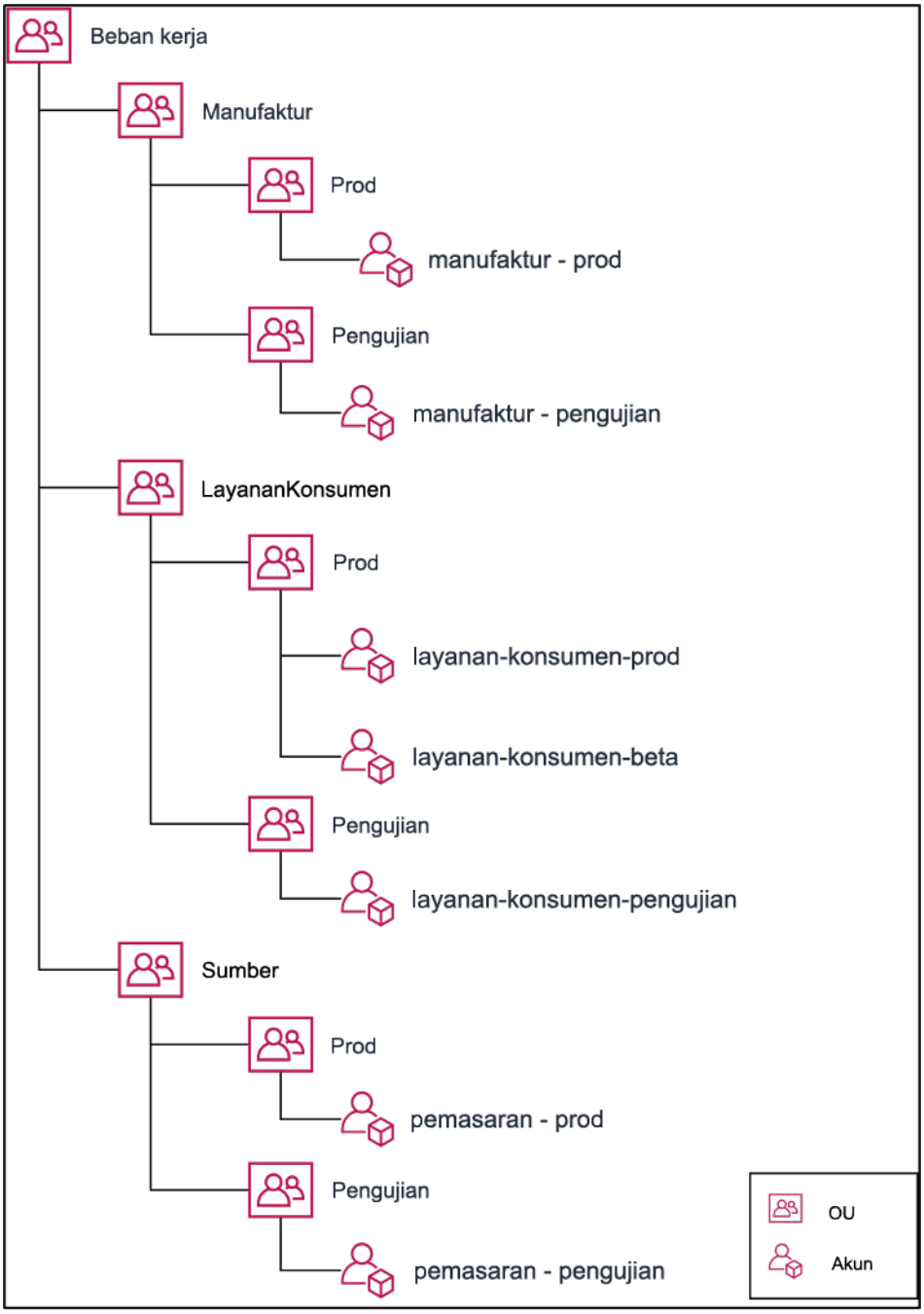
Disarankan untuk memiliki setidaknya satu akun manajemen dengan satu akun anggota yang terhubung ke sana, terlepas dari ukuran organisasi Anda dan penggunaannya. Semua sumber daya beban kerja hanya boleh berada di akun anggota dan tidak boleh ada sumber daya yang dibuat di dalam akun manajemen. Tidak ada satu ukuran yang cocok untuk semua jawaban untuk berapa banyak yang harus Akun AWS Anda miliki. Menilai model operasional dan biaya Anda saat ini dan masa depan untuk memastikan bahwa struktur Anda Akun AWS mencerminkan tujuan organisasi Anda. Beberapa perusahaan membuat beberapa Akun AWS untuk alasan bisnis, misalnya:

- Isolasi penagihan dan fiskal atau administratif diperlukan di antara unit organisasi, pusat biaya, atau beban kerja tertentu.
- AWS batas layanan ditetapkan khusus untuk beban kerja tertentu.
- Ada persyaratan untuk isolasi dan pemisahan antara beban kerja dan sumber daya.

Dalam AWS Organizations, penagihan terkonsolidasi menciptakan hubungan konsep antara satu atau beberapa akun anggota dan akun manajemen. Dengan akun anggota, Anda dapat mengisolasi serta membedakan biaya dan penggunaan berdasarkan grup. Praktik terbaik yang umum dilakukan adalah membuat akun anggota terpisah untuk setiap unit organisasi (seperti keuangan, pemasaran, dan penjualan), atau untuk setiap siklus hidup lingkungan (seperti pengembangan, pengujian, dan produksi), atau untuk setiap beban kerja (beban kerja a, b, dan c), kemudian menggabungkan akun yang terhubung ini menggunakan penagihan terkonsolidasi.

Penagihan terkonsolidasi akan membantu untuk menggabungkan pembayaran untuk beberapa Akun AWS anggota dalam satu akun manajemen, dan tetap memberikan visibilitas untuk setiap aktivitas akun yang dihubungkan. Karena biaya dan penggunaan digabungkan dalam akun manajemen, jadi Anda dapat memaksimalkan diskon volume layanan Anda, dan memaksimalkan penggunaan diskon komitmen Anda (Savings Plans dan Instans Terpesan) untuk mencapai diskon tertinggi.

Diagram berikut menunjukkan bagaimana Anda dapat menggunakan AWS Organizations dengan unit organisasi (OU) untuk mengelompokkan beberapa akun, dan menempatkan beberapa Akun AWS di bawah setiap OU. Disarankan untuk digunakan OUs untuk berbagai kasus penggunaan dan beban kerja yang menyediakan pola untuk mengatur akun.




Contoh pengelompokan beberapa Akun AWS di bawah unit organisasi.

AWS Control Towerdapat dengan cepat mengatur dan mengonfigurasi beberapa AWS akun, memastikan bahwa tata kelola selaras dengan persyaratan organisasi Anda.

Langkah-langkah implementasi

- Tentukan persyaratan pemisahan:  Persyaratan untuk pemisahan adalah kombinasi dari beberapa faktor, termasuk konsep keamanan, konsep keandalan, dan konsep keuangan. Telusuri setiap

faktor secara berurutan dan tentukan apakah beban kerja atau lingkungan beban kerja harus dipisahkan dari beban kerja lainnya. Keamanan meningkatkan perekatan pada persyaratan akses dan data. Keandalan mengelola batasan sehingga lingkungan dan beban kerja tidak memengaruhi hal lain. Pelajari pilar keamanan dan keandalan Kerangka Kerja Well-Architected secara berkala dan ikuti praktik-praktik terbaik yang disediakan. Konstruksi keuangan menciptakan pemisahan keuangan yang ketat (pusat biaya, kepemilikan beban kerja, dan akuntabilitas yang berbeda). Contoh umum pemisahan adalah menjalankan beban kerja pengujian dan produksi di akun terpisah, atau menggunakan akun terpisah agar data penagihan dan faktur dapat diberikan ke unit bisnis atau departemen individu di dalam organisasi atau pemangku kepentingan yang memiliki akun.

- Tentukan persyaratan pengelompokan: Persyaratan pengelompokan tidak menggantikan persyaratan pemisahan, tetapi digunakan untuk membantu manajemen. Kelompokkan menjadi satu berbagai lingkungan atau beban kerja serupa yang tidak perlu dipisahkan. Misalnya, kelompokkan beberapa lingkungan pengujian atau pengembangan dari satu atau beberapa beban kerja menjadi satu.
- Tentukan struktur akun:  Dengan menggunakan pemisahan dan pengelompokan ini, tentukan sebuah akun untuk setiap grup dan pastikan persyaratan pemisahan terpenuhi. Akun-akun ini adalah akun anggota atau akun tertaut Anda. Dengan mengelompokkan akun anggota ini ke dalam satu akun manajemen atau pembayar, Anda menggabungkan penggunaan untuk memperbesar volume diskon di semua akun, yang menyediakan satu tagihan untuk semua akun. Anda dapat memisahkan data penagihan dan menampilkannya secara terpisah untuk setiap akun anggota. Jika akun anggota tidak boleh memiliki data penggunaan atau penagihan yang terlihat oleh akun lain, atau jika tagihan terpisah dari AWS diperlukan, tentukan beberapa akun manajemen atau pembayar. Dalam hal ini, setiap akun anggota memiliki akun manajemen atau pembayarnya masing-masing. Sumber daya harus selalu disimpan di akun anggota atau terkait. Akun manajemen atau pembayar hanya digunakan untuk manajemen.

Sumber daya

Dokumen terkait:

- Menggunakan Tag Alokasi Biaya
- Kebijakan terkelola AWS untuk fungsi pekerjaan
- Strategi penagihan beberapa akun AWS
- Kontrol akses untuk Wilayah AWS menggunakan IAM kebijakan
- AWS Control Tower

- AWS Organizations
- Praktik terbaik untuk akun manajemen dan akun anggota
- Mengatur AWS Lingkungan Anda Menggunakan Beberapa Akun
- Mengaktifkan diskon instans terpesan dan Savings Plans bersama
- Penagihan terkonsolidasi
- Penagihan terkonsolidasi

Contoh terkait:

- Memisahkan CUR dan Berbagi Akses

Video terkait:

- Memperkenalkan AWS Organizations
- Menyiapkan AWS Lingkungan Multi-Akun yang Menggunakan Praktik Terbaik AWS Organizations

Contoh terkait:

- Well-Architected Labs: Buat Organisasi ( AWS Level 100)
- Memisahkan AWS Cost and Usage Report dan Berbagi Akses
- Mendefinisikan Strategi AWS Multi-Akun untuk perusahaan telekomunikasi
- Praktik Terbaik untuk Mengoptimalkan Akun AWS
- Praktik Terbaik untuk Unit Organisasi dengan AWS Organizations

COST02-BP04 Melaksanakan kelompok dan peran

Implementasikan grup dan peran yang selaras dengan kebijakan Anda serta kontrol siapa saja yang dapat membuat, mengubah, atau melakukan dekomisioning instans dan sumber daya di setiap grup. Misalnya, implementasikan grup pengembangan, pengujian, dan produksi. Ini berlaku untuk AWS layanan dan solusi pihak ketiga.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Peran dan grup pengguna merupakan blok pembangun dasar dalam desain dan implementasi sistem yang aman dan efisien. Peran dan grup membantu organisasi menyeimbangkan perlunya kontrol dengan kebutuhan akan fleksibilitas dan produktivitas, yang pada akhirnya mendukung tujuan organisasi dan kebutuhan pengguna. Seperti yang direkomendasikan di bagian Identitas dan manajemen akses dari AWS Pilar Keamanan Kerangka Kerja Well-Architected, Anda memerlukan manajemen identitas dan izin yang kuat untuk menyediakan akses ke sumber daya yang tepat bagi orang yang tepat dalam kondisi yang tepat. Pengguna hanya menerima akses yang diperlukan untuk menyelesaikan tugasnya. Ini meminimalkan risiko yang terkait dengan akses yang tidak sah atau penyalahgunaan.

Setelah mengembangkan kebijakan, Anda dapat membuat peran dan grup logis pengguna di organisasi Anda. Ini memungkinkan Anda untuk menetapkan izin, mengontrol penggunaan, dan membantu menerapkan mekanisme kontrol akses yang kuat, yang mencegah akses tidak sah ke informasi sensitif. Awali dengan pengelompokan orang tingkat tinggi. Hal ini biasanya selaras dengan unit organisasi dan peran pekerjaan (misalnya, administrator sistem di Departemen IT, pengontrol keuangan, atau analis bisnis). Grup tersebut mengategorikan orang yang memiliki tugas serupa dan memerlukan akses serupa. Peran menentukan aktivitas grup. Mengelola izin untuk grup dan peran lebih mudah dibandingkan untuk pengguna individu. Peran dan grup menetapkan izin secara konsisten dan sistematis bagi semua pengguna, sehingga mencegah kesalahan dan inkonsistensi.

Ketika peran pengguna berubah, administrator dapat menyesuaikan akses di tingkat peran atau grup, bukan mengonfigurasi ulang akun setiap pengguna. Misalnya, administrator sistem di TI memerlukan akses untuk membuat semua sumber daya, sedangkan tim analitik hanya perlu membuat sumber daya analitik.

Langkah-langkah implementasi

- Terapkan kelompok:  Dengan menggunakan grup pengguna yang ditentukan dalam kebijakan organisasi Anda, terapkan kelompok yang sesuai, jika perlu. Untuk praktik terbaik tentang pengguna, grup, dan otentikasi, lihat Pilar Keamanan Kerangka Kerja AWS Well-Architected.
- Terapkan peran dan kebijakan:  Dengan menggunakan tindakan yang ditentukan dalam kebijakan organisasi Anda, buatlah peran dan kebijakan akses yang diperlukan. Untuk praktik terbaik tentang peran dan kebijakan, lihat Pilar Keamanan Kerangka AWS Well-Architected.

Sumber daya

Dokumen terkait:

- Kebijakan terkelola AWS untuk fungsi pekerjaan
- Strategi penagihan beberapa akun AWS
- AWS Pilar Keamanan Kerangka Well-Architected
- AWS Identity and Access Management (IAM)
- AWS Identity and Access Management kebijakan

Video terkait:

- Mengapa menggunakan Manajemen Identitas dan Akses

Contoh terkait:

- Identitas dan Akses Dasar Lab Well-Architected
- Kontrol akses untuk Wilayah AWS menggunakan IAM kebijakan
- Memulai perjalanan Manajemen Keuangan Cloud Anda: Operasi biaya cloud

COST02-BP05 Menerapkan kontrol biaya

Implementasikan kontrol berdasarkan kebijakan organisasi serta grup dan peran yang ditetapkan. Ini semua memastikan bahwa biaya hanya dikenakan sesuai yang ditetapkan oleh persyaratan organisasi seperti kontrol akses ke wilayah atau tipe sumber daya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Langkah pertama yang umum dalam mengimplementasikan kontrol biaya adalah mengatur notifikasi ketika peristiwa biaya atau penggunaan terjadi di luar kebijakan. Anda dapat bertindak cepat dan memverifikasi apakah diperlukan tindakan perbaikan, tanpa membatasi atau mengganggu beban kerja atau aktivitas baru. Setelah Anda mengetahui beban kerja dan batasan lingkungan, Anda dapat menegakkan tata kelola. AWS Budgetsmemungkinkan Anda untuk mengatur notifikasi dan menentukan anggaran bulanan untuk AWS biaya, penggunaan, dan diskon komitmen Anda (Savings Plans and Reserved Instances). Anda dapat membuat anggaran pada tingkat biaya agregat (misalnya semua biaya), atau pada tingkat yang lebih mendetail yakni hanya menyertakan dimensi tertentu seperti akun yang dihubungkan, layanan, tag, atau Zona Ketersediaan.

Setelah Anda mengatur batas anggaran Anda AWS Budgets, gunakan AWS Cost Anomaly Detectionuntuk mengurangi biaya tak terduga Anda. AWS Cost Anomaly Detection adalah layanan manajemen biaya yang menggunakan pembelajaran mesin untuk terus memantau biaya dan penggunaan Anda untuk mendeteksi pengeluaran yang tidak biasa. Hal ini membantu Anda mengidentifikasi pengeluaran yang tidak wajar dan akar masalah, sehingga Anda dapat mengambil tindakan secara cepat. Pertama, buat monitor biaya AWS Cost Anomaly Detection, lalu pilih preferensi peringatan Anda dengan menyiapkan ambang dolar (seperti peringatan tentang anomali dengan dampak lebih dari $1.000). Setelah Anda menerima pemberitahuan, Anda dapat menganalisis akar masalah penyebab anomali dan dampaknya pada biaya Anda. Anda juga dapat memantau dan melakukan analisis anomali Anda sendiri di AWS Cost Explorer.

Menegakkan kebijakan tata kelola AWS melalui AWS Identity and Access Managementdan Kebijakan Kontrol AWS Organizations Layanan () SCP. IAMmemungkinkan Anda mengelola akses ke AWS layanan dan sumber daya dengan aman. Dengan menggunakanIAM, Anda dapat mengontrol siapa yang dapat membuat atau mengelola AWS sumber daya, jenis sumber daya yang dapat dibuat, dan di mana mereka dapat dibuat. Hal ini meminimalkan kemungkinan sumber daya dibuat di luar kebijakan yang ditetapkan. Gunakan peran dan grup yang dibuat sebelumnya dan tetapkan IAMkebijakan untuk menerapkan penggunaan yang benar. SCPmenawarkan kontrol pusat atas izin maksimum yang tersedia untuk semua akun di organisasi Anda, menjaga akun Anda tetap dalam pedoman kontrol akses Anda. SCPshanya tersedia di organisasi yang mengaktifkan semua fitur, dan Anda dapat mengonfigurasi SCPs untuk menolak atau mengizinkan tindakan untuk akun anggota secara default. Untuk detail selengkapnya tentang penerapan manajemen akses, lihat Laporan resmi Pilar Keamanan Well-Architected.

Tata kelola juga dapat diimplementasikan melalui manajemen Kuota Layanan (Service Quotas)AWS. Dengan memastikan kuota layanan (service quotas) diatur dengan biaya overhead minimum dan dipelihara secara akurat, Anda dapat meminimalkan pembuatan sumber daya di luar kebutuhan organisasi. Untuk meraih hal ini, Anda harus memahami seberapa cepat kebutuhan Anda dapat berubah, memahami proyek yang sedang berlangsung (baik pembuatan maupun penonaktifan sumber daya), dan mempertimbangkan seberapa cepat perubahan kuota dapat diimplementasikan. Kuota layanan (service quotas) dapat digunakan untuk meningkatkan kuota Anda saat diperlukan.

Langkah-langkah implementasi

- Terapkan pemberitahuan tentang pengeluaran: Untuk menggunakan kebijakan organisasi yang Anda tentukan, buatlah AWS Budgets untuk memberi tahu Anda ketika pengeluaran melonjak di luar kebijakan Anda. Konfigurasikan beberapa anggaran biaya, satu untuk masing-masing akun, yang memberi tahu Anda tentang keseluruhan pengeluaran akun. Konfigurasikan anggaran

biaya tambahan di dalam masing-masing akun untuk unit yang lebih kecil di dalam akun. Unit-unit tersebut berbeda-beda tergantung struktur akun Anda. Beberapa contoh umum adalah Wilayah AWS, beban kerja (menggunakan tag), atau AWS layanan. Konfigurasikan daftar distribusi email sebagai penerima notifikasi, bukan akun email individu. Anda dapat mengonfigurasi anggaran riil ketika jumlah terlampaui, atau gunakan prakiraan anggaran untuk memberitahukan prakiraan penggunaan. Anda juga dapat mengonfigurasi Tindakan AWS Anggaran yang dapat menerapkan kebijakan tertentu IAM atau SCP kebijakan, atau menghentikan instans Amazon atau EC2 Amazon RDS target. Tindakan Anggaran dapat dimulai secara otomatis atau memerlukan persetujuan alur kerja.

- Terapkan pemberitahuan tentang pengeluaran yang tidak biasa: Gunakan AWS Cost Anomaly Detection untuk mengurangi lonjakan biaya di organisasi Anda dan analisis akar penyebab potensi terjadinya pengeluaran tak biasa. Setelah Anda membuat monitor biaya untuk mengidentifikasi pengeluaran yang tidak biasa pada perincian yang Anda tentukan dan mengonfigurasi notifikasi AWS Cost Anomaly Detection, itu mengirimi Anda peringatan ketika pengeluaran yang tidak biasa terdeteksi. Hal ini memungkinkan Anda untuk menganalisis akar masalah penyebab anomali tersebut dan memahami dampaknya terhadap biaya Anda. Gunakan AWS Cost Categories saat mengonfigurasi AWS Cost Anomaly Detection untuk mengidentifikasi tim proyek atau tim unit bisnis mana yang dapat menganalisis akar penyebab biaya tak terduga dan mengambil tindakan yang diperlukan tepat waktu.
- Menerapkan kontrol pada penggunaan: Menggunakan kebijakan organisasi yang ditentukan, menerapkan IAM kebijakan dan peran untuk menentukan tindakan mana yang dapat dilakukan pengguna dan tindakan mana yang tidak dapat mereka lakukan. Beberapa kebijakan organisasi dapat dimasukkan dalam AWS kebijakan. Seperti saat Anda menetapkan kebijakan, mulailah secara umum lalu terapkan kontrol yang lebih mendetail di masing-masing langkah. Batas layanan juga merupakan kontrol penggunaan yang efektif. Implementasikan batas layanan yang tepat pada semua akun Anda.

Sumber daya

Dokumen terkait:

- Kebijakan terkelola AWS untuk fungsi pekerjaan
- Strategi penagihan beberapa akun AWS
- Kontrol akses untuk Wilayah AWS menggunakan IAM kebijakan
- AWS Budgets
- AWS Cost Anomaly Detection

- Kontrol AWS Biaya Anda

Video terkait:

- Bagaimana saya bisa menggunakan AWS Budgets untuk melacak pengeluaran dan penggunaan saya

Contoh terkait:

- Contoh kebijakan manajemen IAM akses
- Contoh kebijakan-kebijakan kontrol layanan
- AWS Anggaran Tindakan
- Membuat IAM Kebijakan untuk mengontrol akses ke EC2 sumber daya Amazon menggunakan Tag
- Batasi akses IAM Identitas ke sumber daya Amazon EC2 tertentu
- Membuat IAM Kebijakan untuk membatasi EC2 penggunaan Amazon menurut keluarga
- Lab Well-Architected: Tata Kelola Biaya dan Penggunaan (Tingkat 100)
- Lab Well-Architected: Tata Kelola Biaya dan Penggunaan (Tingkat 200)
- Integrasi kendur untuk Deteksi Anomali Biaya menggunakan AWS Chatbot

COST02-BP06 Lacak siklus hidup proyek

Lacak, ukur, dan audit siklus hidup proyek, tim, dan lingkungan untuk menghindari penggunaan dan pembayaran sumber daya yang tidak perlu.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Dengan melacak siklus hidup proyek secara efektif, organisasi dapat mencapai pengendalian biaya yang lebih baik melalui perencanaan, manajemen, dan optimalisasi sumber daya yang ditingkatkan. Wawasan yang diperoleh melalui pelacakan sangat berharga untuk membuat keputusan yang lebih tepat yang berperan dalam efektivitas biaya dan keberhasilan proyek secara keseluruhan.

Melacak seluruh siklus kerja beban kerja membantu Anda memahami kapan beban kerja atau komponen beban kerja tidak lagi diperlukan. Beban kerja dan komponen yang ada mungkin tampak sedang digunakan, tetapi ketika AWS merilis layanan atau fitur baru, mereka dapat dinonaktifkan

atau diadopsi. Periksa tahapan beban kerja sebelumnya. Setelah beban kerja diproduksi, lingkungan sebelumnya bisa dinonaktifkan atau jauh dikurangi kapasitasnya sampai diperlukan lagi.

Anda dapat memberikan tag pada sumber daya dengan jangka waktu atau pengingat untuk menyematkan waktu peninjauan beban kerja. Misalnya, jika lingkungan pengembangan terakhir kali ditinjau beberapa bulan lalu, sekarang mungkin adalah saat yang tepat untuk meninjaunya lagi untuk mempelajari apakah layanan baru dapat diadopsi atau apakah lingkungan sedang digunakan. Anda dapat mengelompokkan dan menandai aplikasi Anda AWS untuk mengelola dan melacak metadata seperti kekritisan, lingkungan, ulasan terakhir, dan pusat biaya. myApplications Anda dapat melacak siklus hidup beban kerja Anda dan memantau serta mengelola biaya, kesehatan, postur keamanan, dan kinerja aplikasi Anda.

AWS menyediakan berbagai layanan manajemen dan tata kelola yang dapat Anda gunakan untuk pelacakan siklus hidup entitas. Anda dapat menggunakan AWS Configatau AWS Systems Manager untuk menyediakan inventaris rinci AWS sumber daya dan konfigurasi Anda. Anda disarankan untuk mengintegrasikan proyek atau sistem manajemen aset yang sudah ada agar proyek dan produk aktif di dalam organisasi Anda tetap terlacak. Menggabungkan sistem Anda saat ini dengan serangkaian peristiwa dan metrik yang kaya yang disediakan oleh AWS memungkinkan Anda membangun tampilan peristiwa siklus hidup yang signifikan dan secara proaktif mengelola sumber daya untuk mengurangi biaya yang tidak perlu.

Mirip dengan Manajemen Siklus Hidup Aplikasi (ALM), melacak siklus hidup proyek harus melibatkan beberapa proses, alat, dan tim yang bekerja sama, seperti desain dan pengembangan, pengujian, produksi, dukungan, dan redundansi beban kerja.

Dengan memantau setiap fase siklus hidup proyek secara cermat, organisasi memperoleh wawasan penting dan kontrol yang lebih baik, sehingga dapat memfasilitasi perencanaan, implementasi, dan penyelesaian proyek dengan baik. Pengawasan yang cermat ini memverifikasi bahwa proyek tidak hanya memenuhi standar kualitas, tetapi juga disampaikan tepat waktu dan tidak melampaui anggaran, sehingga meningkatkan efisiensi biaya secara keseluruhan.

Untuk detail selengkapnya tentang penerapan pelacakan siklus hidup entitas, lihat Laporan resmi Pilar Keunggulan Operasional AWS Well-Architected.

Langkah-langkah implementasi

- Tetapkan proses pemantauan siklus hidup proyek: Tim Keunggulan Pusat Cloud harus menetapkan proses pemantauan siklus hidup proyek. Tetapkan pendekatan terstruktur dan sistematis untuk memantau beban kerja agar dapat meningkatkan kontrol, visibilitas, dan performa

proyek. Jadikan proses pemantauan transparan, kolaboratif, dan berfokus pada peningkatan berkelanjutan untuk memaksimalkan efektivitas dan nilainya.

- Lakukan peninjauan beban kerja: Seperti yang ditentukan oleh kebijakan organisasi Anda, siapkan jadwal rutin untuk mengaudit proyek yang ada dan lakukan peninjauan beban kerja. Besarnya upaya yang dilakukan untuk audit harus sebanding dengan perkiraan risiko, nilai, atau biaya bagi organisasi. Area utama yang disertakan dalam audit adalah risiko insiden atau pemadaman terhadap organisasi, nilai atau kontribusi terhadap organisasi (diukur dalam bentuk pendapatan atau reputasi merek), biaya beban kerja (diukur dalam bentuk total biaya sumber daya dan biaya operasional), dan penggunaan beban kerja (diukur dalam bentuk jumlah hasil organisasi per unit waktu). Jika area-area ini berubah selama siklus hidup, diperlukan penyesuaian beban kerja, seperti penonaktifan penuh atau sebagian.

Sumber daya

Dokumen terkait:

- Panduan untuk Menandai AWS
- Apa itu ALM (Manajemen Siklus Hidup Aplikasi)?
- Kebijakan terkelola AWS untuk fungsi pekerjaan

Contoh terkait:

- Kontrol akses untuk Wilayah AWS menggunakan IAM kebijakan

Alat Terkait:

- AWS Config
- AWS Systems Manager
- AWS Budgets
- AWS Organizations
- AWS CloudFormation

## COST3. Bagaimana cara memantau biaya dan penggunaan Anda?

Tetapkan kebijakan dan prosedur untuk memantau dan mengalokasikan biaya Anda dengan tepat. Hal ini memungkinkan Anda mengukur dan meningkatkan efisiensi biaya beban kerja ini.

Praktik terbaik

- COST03-BP01 Konfigurasikan sumber informasi terperinci
- COST03-BP02 Tambahkan informasi organisasi ke biaya dan penggunaan
- COST03-BP03 Identifikasi kategori atribusi biaya
- COST03-BP04 Menetapkan metrik organisasi
- COST03-BP05 Konfigurasikan alat penagihan dan manajemen biaya
- COST03-BP06 Mengalokasikan biaya berdasarkan metrik beban kerja

COST03-BP01 Konfigurasikan sumber informasi terperinci

Siapkan alat manajemen biaya dan pelaporan untuk meningkatkan analisis dan transparansi data biaya dan penggunaan. Konfigurasikan beban kerja Anda untuk membuat entri log yang memfasilitasi pelacakan dan segregasi biaya dan penggunaan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Dengan informasi penagihan yang mendetail seperti tingkat detail per jam di dalam alat manajemen biaya, organisasi dapat melacak pemakaian mereka dengan lebih mendetail dan mengidentifikasi beberapa penyebab kenaikan biaya. Sumber-sumber data ini menyediakan tampilan paling akurat untuk biaya dan penggunaan di seluruh organisasi Anda.

Anda dapat menggunakan Ekspor Data AWS untuk membuat ekspor dari AWS Cost and Usage Report (CUR) 2.0. Ini adalah cara baru dan direkomendasikan untuk menerima data biaya dan penggunaan terperinci Anda AWS. Ini menyediakan granularitas penggunaan harian atau per jam, tarif, biaya, dan atribut penggunaan untuk semua AWS layanan yang dikenakan biaya (informasi yang sama denganCUR), bersama dengan beberapa perbaikan. Semua dimensi yang mungkin ada di CUR seperti penandaan, lokasi, atribut sumber daya, dan akunIDs.

Ada tiga jenis ekspor berdasarkan jenis ekspor yang ingin Anda buat: ekspor data standar, ekspor ke dasbor biaya dan penggunaan dengan QuickSight integrasi Amazon, atau ekspor data lama.

- Ekspor data standar: Ekspor tabel yang sudah disesuaikan yang dikirimkan ke Amazon S3 secara berulang.
- Dasbor biaya dan penggunaan: Ekspor dan integrasi ke Amazon QuickSight untuk menerapkan dasbor biaya dan penggunaan yang sudah dibuat sebelumnya.

- Ekspor data lama: Ekspor legacy AWS Cost and Usage Report ()CUR.

Anda dapat membuat ekspor data dengan penyesuaian berikut:

- Sertakan sumber daya IDs
- Data alokasi biaya terpisah
- Tingkat detail per jam
- Penentuan Versi
- Jenis kompresi dan format file

Untuk beban kerja yang menjalankan container di Amazon ECS atau AmazonEKS, aktifkan data alokasi biaya terpisah sehingga Anda dapat mengalokasikan biaya kontainer ke unit bisnis dan tim individual, berdasarkan cara beban kerja container menggunakan sumber daya komputasi dan memori bersama. Data alokasi biaya terpisah memperkenalkan data biaya dan penggunaan untuk sumber daya tingkat kontainer baru. AWS Cost and Usage Report Data alokasi biaya terpisah dihitung dengan menghitung biaya ECS layanan individu dan tugas yang berjalan di cluster.

Dasbor biaya dan penggunaan mengekspor tabel dasbor biaya dan penggunaan ke bucket S3 secara berulang dan menyebarkan dasbor biaya dan penggunaan bawaan ke Amazon. QuickSight Gunakan opsi ini jika Anda ingin melakukan deployment dasbor data biaya dan penggunaan dengan cepat tanpa kemampuan untuk penyesuaian.

Jika diinginkan, Anda masih dapat mengekspor CUR dalam mode lama, di mana Anda dapat mengintegrasikan layanan pemrosesan lainnya seperti AWS Gluemenyiapkan data untuk analisis dan melakukan analisis data dengan Amazon Athena SQL menggunakan untuk menanyakan data.

Langkah-langkah implementasi

- Buat ekspor data: Buat ekspor yang disesuaikan dengan data yang Anda inginkan dan kendalikan skema ekspor Anda. Buat ekspor data penagihan dan manajemen biaya menggunakan dasarSQL, dan visualisasikan data penagihan dan manajemen biaya Anda dengan mengintegrasikan dengan Amazon. QuickSight Anda juga dapat mengekspor data Anda dalam mode standar untuk menganalisis data Anda dengan alat pemrosesan lain seperti Amazon Athena.
- Konfigurasikan laporan biaya dan penggunaan: Dengan menggunakan konsol penagihan, konfigurasi setidaknya satu laporan biaya dan penggunaan. Konfigurasikan laporan dengan perincian per jam yang mencakup semua pengidentifikasi dan sumber daya. IDs Anda juga dapat

membuat laporan lain dengan tingkat detail berbeda untuk menyediakan informasi rangkuman dengan tingkat lebih tinggi.

- Konfigurasikan info detail per jam di Cost Explorer: Untuk mengakses data biaya dan penggunaan dengan perincian per jam selama 14 hari terakhir, pertimbangkan untuk mengaktifkan data per jam dan data tingkat sumber daya di konsol penagihan.
- Konfigurasikan pencatatan log aktivitas dalam aplikasi: Pastikan bahwa aplikasi Anda mencatat setiap hasil bisnis yang dicapai sehingga bisa dilacak dan diukur. Pastikan tingkat detail data ini setidaknya per jam sehingga sesuai dengan data biaya dan penggunaan. Untuk detail lebih lanjut tentang pencatatan aktivitas dan pemantauan, lihat Pilar Keunggulan Operasional Well-Architected.

Sumber daya

Dokumen terkait:

- Ekspor Data AWS
- AWS Glue
- Amazon QuickSight
- Harga Manajemen Biaya AWS
- Pemberian tag pada sumber daya AWS
- Menganalisis biaya Anda dengan Cost Explorer
- Mengelola AWS Cost and Usage Report
- Pilar Keunggulan Operasional Well-Architected

Contoh terkait:

- Penyiapan Akun AWS
- Ekspor Data untuk AWS Billing and Cost Management
- AWS Cost Explorer Kasus Penggunaan Umum

COST03-BP02 Tambahkan informasi organisasi ke biaya dan penggunaan

Tentukan skema pemberian tag berdasarkan organisasi, atribut beban kerja, dan kategori alokasi biaya agar Anda dapat memfilter dan mencari sumber daya atau memantau biaya dan penggunaan di alat manajemen biaya. Implementasikan pemberian tag yang konsisten untuk semua sumber daya

jika memungkinkan berdasarkan tujuan, tim, lingkungan, atau kriteria lain yang relevan dengan bisnis Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Terapkan pemberian tag di AWS untuk menambahkan informasi organisasi ke sumber daya Anda, yang kemudian akan ditambahkan ke informasi biaya dan penggunaan Anda. Tag adalah pasangan kunci-nilai — kunci perlu ditentukan dan harus unik di seluruh organisasi, dan nilainya unik untuk grup sumber daya. Sebuah contoh dari suatu pasangan data kunci dan nilai adalah kunci `Environment`, dengan nilai `Production`. Semua sumber daya dalam lingkungan produksi pasti memiliki pasangan kunci-nilai ini. Dengan pemberian tag, Anda dapat melacak dan mengelola biaya dengan informasi organisasi yang relevan dan bermanfaat. Anda dapat menerapkan tag yang merepresentasikan kategori organisasi (seperti pusat biaya, nama aplikasi, proyek, atau pemilik), dan mengidentifikasi beban kerja serta karakteristik beban kerja (misalnya pengujian atau produksi) untuk mengaitkan biaya dan penggunaan di seluruh organisasi.

Saat Anda menerapkan tag ke AWS sumber daya Anda (seperti Amazon Elastic Compute Cloud instance atau Amazon Simple Storage Service bucket) dan mengaktifkan tag, AWS tambahkan informasi ini ke Laporan Biaya dan Penggunaan Anda. Anda dapat menjalankan laporan dan melakukan analisis pada sumber daya yang diberi tag dan tidak diberi tag untuk meningkatkan kepatuhan terhadap kebijakan manajemen biaya dan memastikan atribusi yang akurat.

Membuat dan menerapkan standar AWS penandaan di seluruh akun organisasi membantu Anda mengelola dan mengatur AWS lingkungan Anda secara konsisten dan seragam. Gunakan Kebijakan Tag AWS Organizations untuk menentukan aturan tentang bagaimana tag dapat digunakan pada AWS sumber daya di akun Anda AWS Organizations. Kebijakan Tag memungkinkan Anda untuk dengan mudah mengadopsi pendekatan standar untuk menandai sumber daya AWS

AWS Tag Editor memungkinkan Anda untuk menambahkan, menghapus, dan mengelola tag dari beberapa sumber daya. Degan Tag Editor, Anda dapat mencari sumber daya yang ingin Anda beri tag, lalu mengelola tag untuk sumber daya tersebut dalam hasil pencarian Anda.

AWS Cost Categories memungkinkan Anda untuk menetapkan arti organisasi untuk biaya Anda, tanpa memerlukan tag pada sumber daya. Anda dapat memetakan informasi biaya dan penggunaan ke struktur organisasi internal yang unik. Anda menentukan aturan kategori untuk memetakan dan mengategorikan biaya menggunakan dimensi penagihan, seperti akun dan tag. Selain penandaan, hal ini memberikan kemampuan manajemen pada tingkat yang berbeda. Anda juga dapat memetakan akun dan tag spesifik untuk beberapa proyek.

Langkah-langkah implementasi

- Tentukan skema pemberian tag: Kumpulkan semua pemangku kepentingan dari seluruh bisnis Anda untuk menentukan skema. Hal ini umumnya melibatkan orang-orang yang memiliki peran di bidang teknis, keuangan, dan manajemen. Kumpulkan daftar tag yang wajib dimiliki oleh semua sumber daya serta tag yang sebaiknya dimiliki oleh sumber daya. Verifikasikan bahwa nama dan nilai tag konsisten di seluruh organisasi.
- Tag sumber daya: Untuk menggunakan kategori atribusi biaya tempatkan tag di semua sumber daya di beban kerja Anda berdasarkan kategori. Gunakan alat sepertiCLI, Editor Tag, atau AWS Systems Manager untuk meningkatkan efisiensi.
- Implementasikan AWS Cost Categories: Anda dapat membuat Cost Categories tanpa menerapkan penandaan. Kategori biaya menggunakan dimensi biaya dan penggunaan yang sudah ada. Buat aturan kategori dari skema dan implementasikan ke kategori biaya.
- Pemberian tag otomatis: Untuk memastikan bahwa semua sumber daya memiliki tag yang lengkap dan konsisten, otomatiskan proses penambahan tag pada sumber daya saat sumber daya dibuat. Gunakan layanan seperti AWS CloudFormation untuk memverifikasi bahwa sumber daya mendapat tag saat dibuat. Anda juga dapat membuat solusi yang dapat disesuaikan untuk menambahkan tag secara otomatis menggunakan fungsi Lambda atau gunakan layanan mikro yang akan memeriksa beban kerja secara rutin dan menghapus sumber daya yang tidak memiliki tag, yang mana merupakan kondisi ideal untuk pengujian dan pengembangan lingkungan.
- Awasi dan laporkan pemberian tag: Untuk memverifikasi bahwa Anda memberi tag yang lengkap dan konsisten di seluruh organisasi, awasi dan laporkan tag di seluruh beban kerja Anda. Anda dapat menggunakan AWS Cost Explorer untuk melihat biaya dari sumber daya yang mendapat tag dan tidak mendapat tag, atau gunakan layanan seperti Editor Tag. Tinjau secara berkala jumlah sumber daya yang tidak diberi tag dan ambil tindakan untuk menambahkan tag hingga tingkat pemberian tag yang diinginkan tercapai.

Sumber daya

Dokumen terkait:

- Praktik Terbaik Pemberian Tag
- AWS CloudFormation Tag Sumber Daya
- AWS Cost Categories
- Sumber daya penandaan AWS
- Menganalisis biaya Anda dengan AWS Anggaran

- Menganalisis biaya Anda dengan Cost Explorer
- Mengelola Laporan Biaya dan Penggunaan AWS

Video terkait:

- Bagaimana saya bisa menandai AWS sumber daya saya untuk membagi tagihan saya dengan pusat biaya atau proyek
- Sumber Daya Penandaan AWS

COST03-BP03 Identifikasi kategori atribusi biaya

Identifikasi kategori organisasi seperti unit bisnis, departemen, atau proyek yang dapat digunakan untuk mengalokasikan biaya di dalam organisasi Anda ke entitas pengonsumsi internal. Gunakan kategori tersebut untuk menegakkan akuntabilitas pengeluaran, menciptakan kesadaran biaya, dan mendorong perilaku pemakaian yang efektif.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Proses kategorisasi biaya sangat penting dalam penganggaran, akuntansi, pelaporan keuangan, pengambilan keputusan, benchmarking, dan manajemen proyek. Dengan mengklasifikasikan dan membuat kategori pengeluaran, tim dapat lebih memahami jenis biaya yang mereka keluarkan selama perjalanan cloud mereka, sehingga membantu tim dalam mengambil keputusan yang tepat dan mengelola anggaran secara efektif.

Akuntabilitas pengeluaran cloud sangat bermanfaat untuk menghadirkan manajemen permintaan dan biaya yang disiplin. Hasilnya adalah penghematan biaya cloud yang jauh lebih besar untuk organisasi yang mengalokasikan sebagian besar pengeluaran cloud mereka untuk unit bisnis atau tim yang memakainya. Selain itu, pengalokasian pengeluaran cloud membantu organisasi dalam mengadopsi lebih banyak praktik terbaik tata kelola cloud yang tersentralisasi.

Bekerjasamalah dengan tim keuangan Anda atau pemangku kepentingan lain yang relevan untuk memahami persyaratan tentang bagaimana biaya harus dialokasikan di dalam organisasi Anda selama rapat koordinasi rutin Anda. Biaya beban kerja harus dialokasikan pada seluruh siklus hidup, termasuk pengembangan, pengujian, produksi, dan penonaktifan. Pahami bagaimana biaya dikenakan untuk pembelajaran, pengembangan staf, dan pencetusan ide yang berkaitan dengan organisasi. Dengan begitu, akun yang akan digunakan untuk tujuan ini dapat dialokasikan dengan tepat ke anggaran pelatihan dan pengembangan, bukan anggaran biaya IT umum.

Setelah mendefinisikan kategori atribusi biaya Anda dengan pemangku kepentingan di organisasi Anda, gunakan Cost AWS Categories untuk mengelompokkan informasi biaya dan penggunaan Anda ke dalam kategori yang bermakna di AWS Cloud, seperti biaya untuk proyek tertentu, atau Akun AWS untuk departemen atau unit bisnis. Anda dapat membuat kategori kustom dan memetakan informasi biaya dan penggunaan ke dalam kategori tersebut berdasarkan aturan yang Anda tentukan menggunakan berbagai dimensi seperti akun, tag, layanan, atau jenis biaya. Setelah kategori biaya disiapkan, Anda dapat melihat informasi biaya dan penggunaan berdasarkan kategori tersebut sehingga memungkinkan organisasi Anda membuat keputusan strategis dan pembelian yang lebih baik. Kategori-kategori ini terlihat di AWS Cost Explorer AWS Budgets,, dan AWS Cost and Usage Report juga.

Misalnya, buat kategori biaya untuk unit bisnis Anda (DevOps tim), dan di bawah setiap kategori buat beberapa aturan (aturan untuk setiap sub kategori) dengan beberapa dimensi (Akun AWS, tag alokasi biaya, layanan, atau jenis biaya) berdasarkan pengelompokan yang Anda tentukan. Dengan kategori biaya, Anda dapat mengatur biaya Anda menggunakan mesin berbasis aturan. Aturan yang Anda konfigurasikan akan mengatur biaya ke dalam kategori. Dalam aturan ini, Anda dapat memfilter dengan menggunakan beberapa dimensi untuk setiap kategori seperti spesifik Akun AWS, AWS layanan, atau jenis biaya. Anda kemudian dapat menggunakan kategori-kategori ini untuk berbagai produk di AWS Billing and Cost Management dan konsol Manajemen Biaya. Ini termasuk AWS Cost Explorer, AWS Budgets, AWS Cost and Usage Report, dan AWS Cost Anomaly Detection.

Misalnya, diagram berikut menunjukkan cara mengelompokkan biaya dan informasi penggunaan di organisasi Anda dengan memiliki beberapa tim (kategori biaya), beberapa lingkungan (aturan), dan setiap lingkungan yang memiliki beberapa sumber daya atau aset (dimensi).

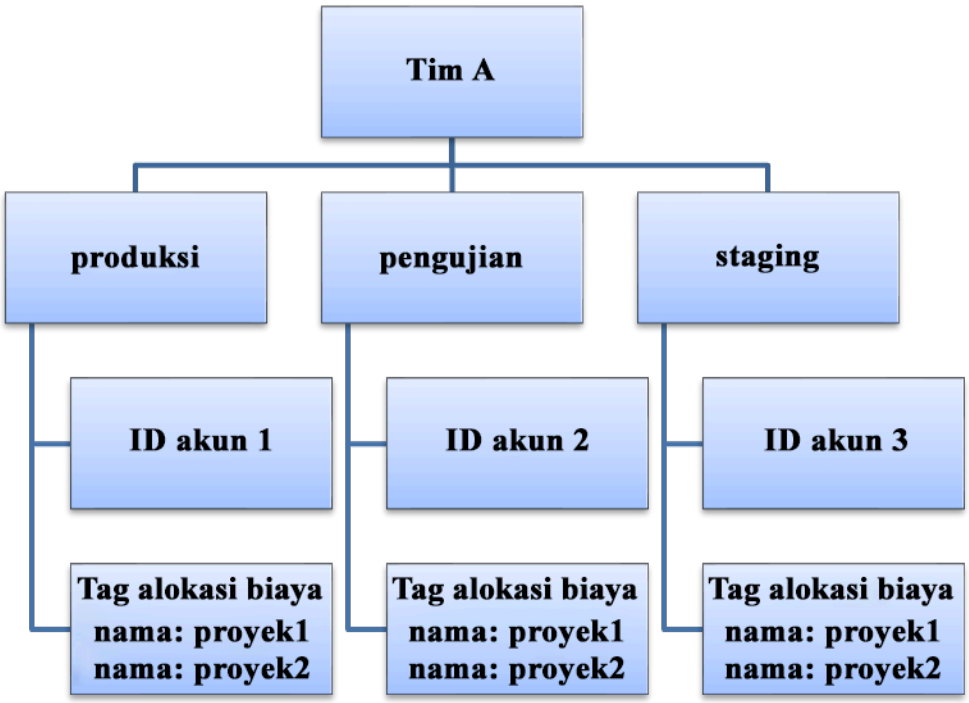



Diagram organisasi biaya dan penggunaan

Anda juga dapat membuat pengelompokan biaya menggunakan kategori biaya. Setelah Anda membuat kategori biaya (perlu waktu 24 jam setelah membuat kategori biaya agar catatan penggunaan Anda dapat diperbarui dengan nilai), kategori biaya tersebut akan muncul di AWS Cost Explorer, AWS Budgets, AWS Cost and Usage Report, dan AWS Cost Anomaly Detection. Di AWS Cost Explorer dan AWS Budgets, kategori biaya muncul sebagai dimensi penagihan tambahan. Anda dapat menggunakan ini untuk memfilter nilai kategori biaya tertentu, atau grup menurut kategori biaya.

Langkah-langkah implementasi

- Tentukan kategori organisasi Anda: Temui para pemangku kepentingan dan unit bisnis internal untuk menentukan kategori yang sesuai dengan struktur dan persyaratan organisasi Anda. Kategori ini akan secara langsung dipetakan ke struktur kategori keuangan yang ada, seperti unit bisnis, anggaran, pusat biaya, dan departemen. Lihat hasil yang diberikan cloud untuk bisnis Anda, seperti pelatihan dan edukasi, karena ini juga merupakan kategori organisasi.
- Tentukan kategori fungsional Anda: Temui para pemangku kepentingan dan unit bisnis internal untuk menentukan kategori yang sesuai dengan fungsi yang Anda miliki dalam bisnis Anda. Hal ini dapat berupa nama aplikasi atau beban kerja, serta jenis lingkungan, seperti produksi, pengujian, atau pengembangan.

- Tentukan AWS Cost Categories: Buat kategori biaya untuk mengatur informasi biaya dan penggunaan Anda dengan menggunakan AWS Cost Categories dan petakan AWS biaya dan penggunaan Anda ke dalam kategori yang bermakna. Beberapa kategori dapat ditetapkan untuk satu sumber daya, dan satu sumber daya dapat berada dalam beberapa kategori yang berbeda, jadi tentukan sebanyak kategori yang dibutuhkan sehingga Anda dapat mengelola biaya Anda ke dalam struktur yang yang dikategorikan menggunakan Kategori Biaya AWS .

Sumber daya

Dokumen terkait:

- Pemberian tag pada sumber daya AWS
- Menggunakan Tag Alokasi Biaya
- Menganalisis biaya Anda dengan AWS Budgets
- Menganalisis biaya Anda dengan Cost Explorer
- Mengelola AWS Cost and Usage Report
- Kategori Biaya AWS
- Mengelola biaya Anda dengan AWS Cost Categories
- Membuat kategori biaya
- Pemberian tag pada kategori biaya
- Membagi pembebanan biaya dalam kategori biaya
- Fitur-Fitur Kategori Biaya AWS

Contoh terkait:

- Atur data biaya dan penggunaan Anda dengan AWS Cost Categories
- Mengelola biaya Anda dengan AWS Cost Categories
- Lab Well-Architected: Visualisasi Biaya dan Penggunaan
- Lab Well-Architected: Kategori Biaya

### COST03-BP04 Menetapkan metrik organisasi

Tetapkan metrik-metrik organisasi yang diperlukan untuk beban kerja ini. Contoh metrik beban kerja adalah laporan pelanggan yang dibuat, atau halaman web yang disajikan untuk pelanggan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pahami bagaimana output beban kerja Anda diukur berdasarkan keberhasilan bisnis. Masing-masing beban kerja umumnya memiliki satu set kecil berisi output-output utama yang mengindikasikan kinerja. Jika Anda memiliki beban kerja yang kompleks dengan banyak komponen, Anda dapat memprioritaskan daftar, atau menetapkan dan melacak metrik untuk setiap komponen. Bekerjalah dengan tim Anda untuk memahami metrik mana yang akan digunakan. Unit ini akan digunakan untuk memahami efisiensi beban kerja, atau biaya untuk masing-masing output bisnis.

Langkah-langkah implementasi

- Tentukan hasil beban kerja: Lakukan pertemuan dengan pemangku kepentingan dalam bisnis dan tetapkan hasil untuk beban kerja. Ini adalah pengukur utama penggunaan pelanggan dan harus berupa metrik bisnis dan bukan metrik teknis. Harus ada sedikit metrik tingkat tinggi (kurang dari lima) per beban kerja. Jika beban kerja memunculkan beberapa hasil untuk kasus-kasus penggunaan yang berbeda, kelompokkan ke dalam satu metrik.
- Tentukan hasil komponen beban kerja: Opsi lainnya adalah jika Anda memiliki beban kerja besar dan kompleks, atau dapat dengan mudah mengurai beban kerja Anda ke dalam komponen (seperti layanan mikro) dengan input dan output yang ditetapkan dengan baik, tetapkan metrik untuk masing-masing komponen. Upaya harus mencerminkan nilai dan biaya komponen. Mulailah dengan komponen yang paling besar menuju komponen yang lebih kecil.

Sumber daya

Dokumen terkait:

- Sumber daya penandaan AWS
- Menganalisis biaya Anda dengan AWS Anggaran
- Menganalisis biaya Anda dengan Cost Explorer
- Mengelola Laporan Biaya dan Penggunaan AWS

COST03-BP05 Konfigurasikan alat penagihan dan manajemen biaya

Konfigurasikan alat manajemen biaya yang sesuai dengan kebijakan organisasi Anda untuk mengelola dan mengoptimalkan pengeluaran cloud. Hal ini mencakup layanan, alat, dan sumber

daya untuk mengatur dan melacak data biaya dan penggunaan, mengoptimalkan kontrol melalui penagihan terkonsolidasi dan izin akses, meningkatkan perencanaan melalui penganggaran dan prakiraan, menerima notifikasi atau peringatan, serta menurunkan biaya dengan sumber daya dan optimalisasi harga.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk membangun akuntabilitas yang kuat, pertimbangkan strategi akun Anda terlebih dahulu sebagai bagian dari strategi alokasi biaya Anda. Jika benar di tahap ini, Anda mungkin tidak perlu berupaya lebih jauh. Jika tidak, dapat muncul ketidaksadaran dan titik-titik masalah lebih lanjut.

Untuk mendorong akuntabilitas pengeluaran cloud, berikan akses kepada pengguna ke alat yang memberikan visibilitas tentang biaya dan penggunaan mereka. AWS menyarankan Anda untuk mengonfigurasi semua beban kerja dan tim untuk tujuan berikut:

- Atur: Tetapkan acuan dasar alokasi dan tata kelola biaya dengan strategi pemberian tag dan taksonomi Anda sendiri. Buat beberapa AWS Akun dengan alat seperti AWS Control Tower atau AWS Organisasi. Tandai AWS sumber daya yang didukung dan kategorikan secara bermakna berdasarkan struktur organisasi Anda (unit bisnis, departemen, atau proyek). Tandai nama akun untuk pusat biaya tertentu dan petakan dengan AWS Cost Categories untuk mengelompokkan akun untuk unit bisnis ke pusat biaya mereka sehingga pemilik unit bisnis dapat melihat konsumsi beberapa akun di satu tempat.
- Akses: Lacak informasi penagihan di seluruh organisasi dalam penagihan terkonsolidasi. Verifikasi bahwa pemangku kepentingan yang tepat dan pemilik bisnis memiliki akses.
- Kontrol: Bangun mekanisme tata kelola yang efektif dengan pagar pembatas yang tepat untuk mencegah skenario tak terduga saat menggunakan Kebijakan Kontrol Layanan (SCP), menandai kebijakan, IAM kebijakan, dan peringatan anggaran. Misalnya, Anda dapat mengizinkan tim untuk membuat sumber daya tertentu di wilayah yang dipilih hanya dengan menggunakan mekanisme kontrol yang efektif dan mencegah pembuatan sumber daya tanpa tag khusus (seperti pusat biaya).
- Kondisi saat ini: Konfigurasikan dasbor yang menampilkan tingkat biaya dan penggunaan saat ini. Dasbor harus tersedia di tempat yang mudah terlihat di dalam lingkungan kerja seperti dasbor operasi. Anda dapat mengekspor data dan menggunakan Dasbor Biaya dan Penggunaan dari Hub Optimisasi Biaya AWS atau produk apa pun yang didukung untuk membuat visibilitas ini. Anda mungkin perlu membuat dasbor yang berbeda untuk persona yang berbeda. Misalnya, dasbor manajer mungkin berbeda dari dasbor rekayasa.

- Pemberitahuan: Memberikan pemberitahuan ketika biaya atau penggunaan melebihi batas yang ditentukan dan anomali terjadi dengan AWS Anggaran atau Deteksi Anomali AWS Biaya.
- Laporan: Meringkas semua informasi biaya dan penggunaan Tingkatkan kesadaran dan akuntabilitas pengeluaran cloud Anda dengan data biaya yang mendetail dan dapat diatribusikan. Buat laporan yang relevan dengan tim yang memakai data tersebut dan berisi rekomendasi.
- Pelacakan: Tampilkan biaya dan penggunaan saat ini dan bandingkan dengan tujuan atau target yang telah ditentukan.
- Analisis: Beri izin anggota tim untuk melakukan analisis khusus dan mendalam hingga perincian per jam, per hari atau per bulan dengan filter yang berbeda-beda (sumber daya, akun, tag, dll.).
- Inspeksi: Ikuti terus perkembangan deployment sumber daya dan peluang untuk mengoptimalkan biaya Anda. Dapatkan notifikasi menggunakan Amazon CloudWatch, AmazonSNS, atau Amazon SES untuk penerapan sumber daya di tingkat organisasi. Tinjau rekomendasi pengoptimalan biaya dengan AWS Trusted Advisor atau AWS Compute Optimizer.
- Laporan tren: Menampilkan keberagaman biaya dan penggunaan selama periode waktu yang diinginkan, dengan detail yang diperlukan.
- Prakiraan: Tampilkan perkiraan biaya di masa depan, perkirakan penggunaan sumber daya Anda, dan lacak pengeluaran dengan dasbor perkiraan yang Anda buat.

Anda dapat menggunakan Hub Optimisasi Biaya AWS untuk memahami potensi penghematan biaya yang dikonsolidasikan dari lokasi terpusat dan membuat ekspor data untuk integrasi dengan Amazon Athena. Anda juga dapat menggunakan Hub Pengoptimalan AWS Biaya untuk menerapkan Dasbor Biaya dan Penggunaan, yang memanfaatkan Amazon QuickSight untuk analisis biaya interaktif dan berbagi wawasan biaya yang aman.

Jika Anda tidak memiliki keterampilan atau bandwidth penting dalam organisasi Anda, Anda dapat bekerja dengan AWS ProServ, AWS Managed Services (AMS), atau AWS Mitra. Anda juga dapat menggunakan alat pihak ketiga tetapi pastikan Anda memvalidasi proposisi nilainya.

Langkah-langkah implementasi

- Beri izin akses berbasis tim ke alat-alat: Konfigurasikan akun Anda dan buat kelompok yang memiliki akses ke laporan biaya dan penggunaan yang diperlukan untuk mereka pakai dan gunakan AWS Identity and Access Management untuk mengontrol akses ke alat-alat seperti AWS Cost Explorer. Grup tersebut harus menyertakan perwakilan dari semua tim yang memiliki atau mengelola sebuah aplikasi. Hal ini bertujuan untuk memastikan setiap tim dapat mengakses informasi biaya dan penggunaan untuk melacak konsumsi mereka.

- Kelola Tag dan Kategori Biaya: atur biaya Anda di seluruh tim, unit bisnis, aplikasi, lingkungan, dan proyek. Gunakan tag sumber daya untuk mengatur biaya, berdasarkan tag alokasi biaya. Buat Kategori Biaya berdasarkan dimensi dengan menggunakan tag, akun, layanan, dll. untuk memetakan biaya Anda.
- Konfigurasi AWS Anggaran: Konfigurasikan AWS Anggaran di semua akun untuk beban kerja Anda. Tetapkan anggaran untuk pengeluaran akun secara keseluruhan, serta anggaran untuk beban kerja menggunakan tag dan kategori biaya. Konfigurasikan pemberitahuan dalam AWS Anggaran untuk menerima peringatan ketika Anda melebihi jumlah yang dianggarkan, atau ketika perkiraan biaya melebihi anggaran Anda.
- Konfigurasikan Deteksi Anomali AWS AWS Biaya: Gunakan Deteksi Anomali Biaya untuk akun, layanan inti, atau kategori biaya yang Anda buat untuk memantau biaya dan penggunaan Anda serta mendeteksi pengeluaran yang tidak biasa. Anda dapat menerima peringatan secara individual dalam laporan agregat dan menerima peringatan dalam email atau SNS topik Amazon yang memungkinkan Anda menganalisis dan menentukan akar penyebab anomali dan mengidentifikasi faktor yang mendorong kenaikan biaya.
- Gunakan alat analisis biaya: Tentukan AWS Cost Explorer untuk beban kerja dan akun Anda guna memvisualisasikan data biaya untuk analisis lebih lanjut. Buat dasbor untuk beban kerja yang melacak pengeluaran secara keseluruhan, metrik penggunaan utama untuk beban kerja, dan perkiraan biaya mendatang berdasarkan data biaya historis Anda.
- Gunakan alat analisis hemat biaya: Gunakan Hub Pengoptimalan AWS Biaya untuk mengidentifikasi peluang penghematan dengan rekomendasi yang disesuaikan termasuk menghapus sumber daya yang tidak digunakan, hak, Rencana penghematan, reservasi, dan rekomendasi pengoptimal komputasi.
- Sesuaikan fitur-fitur lanjutan yang tersedia.: Anda dapat secara opsional membuat visualisasi untuk memudahkan analisis interaktif dan berbagi wawasan biaya. Dengan Data Exports on AWS Cost Optimization Hub, Anda dapat membuat dasbor biaya dan penggunaan yang didukung oleh Amazon QuickSight untuk organisasi Anda yang memberikan detail dan perincian tambahan. Anda juga dapat menerapkan kemampuan analisis lanjutan dengan menggunakan ekspor data di Amazon Athena untuk kueri lanjutan, dan membuat dasbor di Amazon. QuickSight Bekerja sama dengan Mitra AWS untuk mengadopsi solusi manajemen cloud untuk pemantauan dan pengoptimalan penagihan cloud terkonsolidasi.

Sumber daya

Dokumen terkait:

- Apa itu AWS Billing and Cost Management dan Manajemen Biaya?
- Membangun AWS lingkungan praktik terbaik Anda
- Praktik Terbaik untuk Menandai Sumber Daya AWS
- Menandai sumber daya Anda AWS
- AWS Cost Categories
- Menganalisis biaya Anda dengan AWS Anggaran
- Menganalisis biaya Anda dengan AWS Cost Explorer
- Apa itu Ekspor AWS Data?

Video terkait:

- Melakukan Deployment Dasbor Inteligensi Cloud
- Dapatkan Pemberitahuan tentang Metrik Pengoptimalan Biaya apa pun FinOps atau KPI

Contoh terkait:

- Dasbor Biaya dan Penggunaan didukung oleh Amazon QuickSight
- Lokakarya Tata Kelola Biaya dan Penggunaan AWS

COST03-BP06 Mengalokasikan biaya berdasarkan metrik beban kerja

Alokasikan biaya beban kerja berdasarkan metrik penggunaan atau hasil bisnis untuk mengukur efisiensi biaya beban kerja. Implementasikan proses untuk menganalisis data biaya dan penggunaan dengan layanan analitik, yang dapat menyediakan wawasan dan kemampuan charge back.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Optimisasi biaya adalah menghadirkan hasil bisnis pada titik harga terendah, yang hanya dapat dicapai dengan mengalokasikan biaya beban kerja berdasarkan metrik beban kerja (diukur berdasarkan efisiensi beban kerja). Pantau metrik beban kerja yang ditetapkan melalui file log atau pemantauan aplikasi lain. Kombinasikan data ini dengan biaya beban kerja, yang dapat diperoleh dengan melihat biaya dengan nilai tag khusus atau ID akun. Lakukan analisis ini pada level per jam. Efisiensi umumnya akan berubah jika Anda memiliki komponen biaya statis (misalnya basis data backend yang berjalan secara permanen) dengan beragam laju permintaan (misalnya puncak

penggunaan pada pukul sembilan pagi hingga lima sore, dengan sedikit permintaan pada malam hari). Memahami hubungan antara biaya statis dan variabel membantu Anda fokus pada aktivitas optimalisasi Anda.

Membuat metrik beban kerja untuk sumber daya bersama mungkin sulit dibandingkan dengan sumber daya seperti aplikasi kontainer di Amazon Elastic Container Service (Amazon) ECS dan Amazon Gateway. API Namun, ada beberapa cara tertentu bagi Anda untuk mengkategorikan penggunaan dan melacak biaya. Jika Anda perlu melacak Amazon ECS dan sumber daya AWS Batch bersama, Anda dapat mengaktifkan data alokasi biaya terpisah. AWS Cost Explorer Dengan data alokasi biaya terpisah, Anda dapat memahami dan mengoptimalkan biaya serta penggunaan aplikasi dalam kontainer dan mengalokasikan biaya aplikasi kembali ke entitas bisnis masing-masing berdasarkan cara pemakaian sumber daya komputasi dan memori bersama mereka.

Langkah-langkah implementasi

- Alokasikan metrik biaya per beban kerja: Dengan menggunakan metrik dan tag terkonfigurasi yang sudah ditetapkan, buatlah sebuah metrik yang mengombinasikan hasil akhir beban kerja dan biaya beban kerja. Gunakan layanan analitik seperti Amazon Athena dan Amazon QuickSight untuk membuat dasbor efisiensi untuk keseluruhan beban kerja dan komponen apa pun.

Sumber daya

Dokumen terkait:

- Sumber daya penandaan AWS
- Menganalisis biaya Anda dengan AWS Anggaran
- Menganalisis biaya Anda dengan Cost Explorer
- Mengelola Laporan Biaya dan Penggunaan AWS

Contoh terkait:

- Meningkatkan visibilitas biaya Amazon ECS dan AWS Batch dengan AWS Split Cost Allocation Data

## COST4. Bagaimana cara melakukan penonaktifan sumber daya?

Menerapkan kontrol perubahan dan manajemen sumber daya dari awal proyek hingga. end-of-life Hal ini memastikan Anda akan mematikan atau mengakhiri sumber daya yang tidak digunakan agar tidak boros.

Praktik terbaik

- COST04-BP01 Lacak sumber daya selama masa pakainya
- COST04-BP02 Menerapkan proses dekomisioning
- COST04-BP03 Sumber daya dekomisi
- COST04-BP04 Sumber daya penonaktifan secara otomatis
- COST04-BP05 Menegakkan kebijakan penyimpanan data

### COST04-BP01 Lacak sumber daya selama masa pakainya

Tentukan dan implementasikan metode untuk melacak sumber daya dan kaitannya dengan sistem sepanjang masa pakainya. Anda dapat menggunakan pemberian tag untuk mengidentifikasi beban kerja atau fungsi sumber daya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Nonaktifkan sumber daya beban kerja yang sudah tidak diperlukan. Contoh yang umum adalah sumber daya yang digunakan untuk pengujian: setelah pengujian selesai, sumber daya dapat dikeluarkan. Melacak sumber daya dengan tag (dan menjalankan laporan atas tag-tag tersebut) dapat membantu Anda mengidentifikasi aset untuk dinonaktifkan karena tidak digunakan atau yang lisensinya akan kedaluwarsa. Menggunakan tag merupakan cara efektif untuk melacak sumber daya, dengan memberi label sumber daya dengan fungsinya, atau tanggal kapan sumber daya dapat dinonaktifkan. Maka pelaporan dapat dijalankan atas tag ini. Contoh nilai untuk pemberian tag fitur adalah `feature-X testing` untuk mengidentifikasi tujuan sumber daya dalam hal siklus hidup beban kerja. Contoh lain adalah menggunakan `LifeSpan` atau `TTL` untuk sumber daya, seperti nama kunci to-be-deleted tag dan nilai untuk menentukan periode waktu atau waktu tertentu untuk penonaktifan.

Langkah-langkah implementasi

- Terapkan skema pemberian tag: Terapkan skema pemberian tag yang mengidentifikasi beban kerja tempat sumber daya tersebut berada, verifikasi bahwa semua sumber daya dalam beban kerja diberi tag yang sesuai. Pemberian tag akan membantu Anda membuat kategori sumber daya berdasarkan tujuan, tim, lingkungan, atau kriteria lain yang relevan dengan bisnis Anda. Untuk detail selengkapnya tentang kasus, strategi, dan teknik penggunaan pemberian tag, lihat Praktik Terbaik Pemberian Tag AWS.
- Menerapkan throughput beban kerja atau pemantauan output: Terapkan pemantauan atau peringatan throughput beban kerja, dimulai dari permintaan input atau penyelesaian output. Konfigurasikan untuk memberikan notifikasi ketika permintaan beban kerja atau output menurun hingga nol, yang menandakan sumber daya beban kerja sudah tidak digunakan. Sertakan faktor waktu jika beban kerja secara berkala menurun hingga nol dalam kondisi normal. Untuk detail lebih lanjut tentang sumber daya yang tidak digunakan atau kurang dimanfaatkan, lihat pemeriksaan Pengoptimalan Biaya AWS Trusted Advisor.
- AWS Sumber daya grup: Buat grup untuk AWS sumber daya. Anda dapat menggunakan AWS Resource Groupsuntuk mengatur dan mengelola AWS sumber daya Anda yang sama Wilayah AWS. Anda dapat menambahkan tag ke sebagian besar sumber daya Anda untuk membantu mengidentifikasi dan menyortir sumber daya Anda di dalam organisasi Anda. Gunakan Editor Tag untuk menambahkan tag ke sumber daya yang didukung secara massal. Pertimbangkan untuk menggunakan AWS Service Catalog untuk membuat, mengelola dan mendistribusikan portofolio produk-produk yang disetujui ke para pengguna akhir dan kelola siklus hidup produk.

Sumber daya

Dokumen terkait:

- AWS Auto Scaling
- AWS Trusted Advisor
- AWS Trusted Advisor Pemeriksaan Optimalisasi Biaya
- Sumber daya penandaan AWS
- Memublikasikan Metrik Kustom

Video terkait:

- Cara mengoptimalkan biaya menggunakan AWS Trusted Advisor

Contoh terkait:

- Mengatur AWS sumber daya
- Optimalkan biaya menggunakan AWS Trusted Advisor

COST04-BP02 Menerapkan proses dekomisioning

Implementasikan proses untuk mengidentifikasi dan menonaktifkan sumber daya tidak terpakai

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Implementasikan proses standar di organisasi Anda untuk mengidentifikasi dan menyingkirkan sumber daya yang tidak digunakan. Proses tersebut harus menetapkan frekuensi pelaksanaan pencarian, dan proses untuk menyingkirkan sumber daya guna memverifikasi terpenuhinya semua persyaratan organisasi.

Langkah-langkah implementasi

- Buat dan terapkan proses penonaktifan: Bekerjalah dengan para pengembang dan pemilik beban kerja untuk membangun proses penonaktifan untuk beban kerja dan sumber dayanya. Proses tersebut harus mencakup metode untuk memverifikasi apakah beban kerja sedang digunakan, begitu juga dengan setiap sumber daya beban kerja. Buat detail langkah-langkah yang diperlukan untuk menonaktifkan sumber daya, yakni menghapusnya dari layanan sambil memastikan kepatuhan terhadap semua persyaratan peraturan. Semua sumber daya terkait harus disertakan, seperti lisensi atau penyimpanan terlampir. Beri tahu para pemilik beban kerja bahwa proses penonaktifan telah dimulai.

  Gunakan langkah-langkah penonaktifan berikut ini untuk memandu Anda terkait hal-hal yang harus diperiksa sebagai bagian dari proses Anda:

  - Identifikasi sumber daya yang akan dinonaktifkan: Identifikasi sumber daya yang memenuhi syarat untuk dinonaktifkan di AWS Cloud Anda. Rekam semua informasi yang diperlukan dan jadwalkan penonaktifan. Dalam lini waktu Anda, pastikan untuk mempertimbangkan apakah (dan kapan) masalah yang tidak terduga muncul selama proses.
  - Koordinasi dan komunikasi: Bekerja sama dengan para pemilik beban kerja untuk mengonfirmasi sumber daya yang akan dinonaktifkan
  - Rekam metadata dan buat cadangan: Rekam metadata (seperti publikIPs, Wilayah, AZ,, SubnetVPC, dan Grup Keamanan) dan buat cadangan (seperti snapshot atau pengambilan

Amazon Elastic Block Store, ekspor kunciAMI, dan ekspor Sertifikat) jika diperlukan untuk sumber daya di lingkungan produksi atau jika itu adalah sumber daya penting.

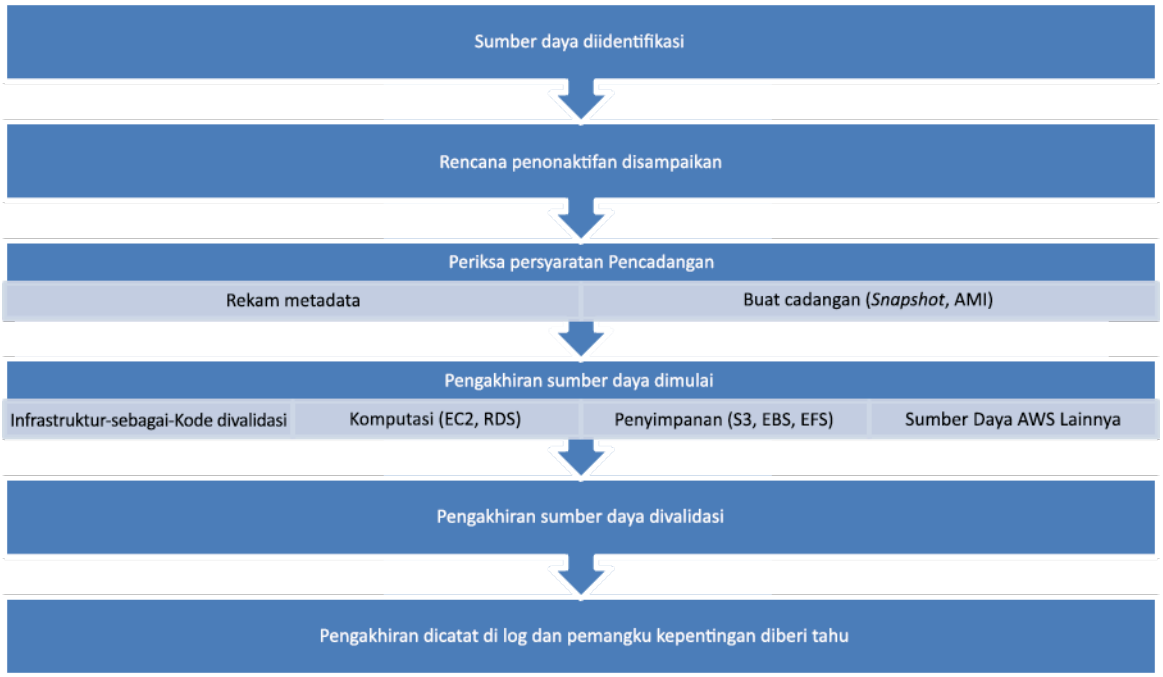
- Validasi infrastructure-as-code: Tentukan apakah sumber daya diterapkan dengan AWS CloudFormation, Terraform, AWS Cloud Development Kit (AWS CDK), atau alat infrastructure-as-code penerapan lainnya sehingga dapat diterapkan kembali jika perlu.
- Mencegah akses: Terapkan kendali batasan untuk jangka waktu tertentu, untuk mencegah penggunaan sumber daya saat Anda sedang memutuskan apakah sumber daya diperlukan. Lakukan verifikasi bahwa lingkungan sumber daya dapat dikembalikan ke status aslinya jika diperlukan.
- Ikuti proses penonaktifan internal Anda: Ikuti tugas administratif dan proses penonaktifan organisasi Anda, seperti menghapus sumber daya dari domain organisasi Anda, menghapus DNS catatan, dan menghapus sumber daya dari alat manajemen konfigurasi, alat pemantauan, alat otomatisasi, dan alat keamanan.

Jika sumber daya adalah EC2 instance Amazon, lihat daftar berikut. Untuk detail selengkapnya, lihat Bagaimana cara menghapus atau menghentikan EC2 sumber daya Amazon saya?

- Hentikan atau hentikan semua EC2 instans Amazon dan penyeimbang beban Anda. EC2Instans Amazon terlihat di konsol untuk waktu yang singkat setelah dihentikan. Anda tidak menerima tagihan untuk instans apa pun yang tidak memiliki status berjalan
- Menghapus infrastruktur penskalaan otomatis Anda
- Lepaskan semua Host Khusus.
- Hapus semua EBS volume Amazon dan EBS snapshot Amazon.
- Lepaskan semua Alamat IP elastis.
- Deregister semua Gambar Mesin Amazon (). AMIs
- Hentikan semua AWS Elastic Beanstalk lingkungan.

Jika sumber daya adalah sebuah objek di dalam penyimpanan Amazon S3 Glacier dan jika Anda menghapus sebuah arsip sebelum memenuhi durasi penyimpanan minimum, Anda akan dikenakan biaya penghapusan dini prorata. Durasi penyimpanan minimum Amazon S3 Glacier tergantung pada kelas penyimpanan yang digunakan. Untuk ringkasan durasi penyimpanan minimum untuk setiap kelas penyimpanan, lihat Kinerja di seluruh kelas penyimpanan Amazon S3. Untuk detail tentang cara menghitung biaya penghapusan awal, lihat harga Amazon S3.

Bagan alur proses penonaktifan sederhana berikut ini menguraikan langkah-langkah penonaktifan. Sebelum menonaktifkan sumber daya, verifikasikan bahwa sumber daya yang telah Anda identifikasi untuk dinonaktifkan tidak sedang digunakan oleh organisasi.




Aliran penonaktifan sumber daya.

Sumber daya

Dokumen terkait:

- AWS Auto Scaling
- AWS Trusted Advisor
- AWS CloudTrail

Video terkait:

- Hapus CloudFormation tumpukan tetapi pertahankan beberapa sumber daya
- Cari tahu pengguna mana yang meluncurkan EC2 instans Amazon

Contoh terkait:

- Hapus atau hentikan sumber daya Amazon EC2
- Cari tahu pengguna mana yang meluncurkan EC2 instans Amazon

### COST04-BP03 Sumber daya dekomisi

Nonaktifkan sumber daya yang diinisiasi oleh peristiwa seperti audit berkala, atau perubahan penggunaan. Penonaktifan umumnya dilakukan secara berkala dan dapat dilakukan secara manual atau otomatis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

#### Panduan implementasi

Frekuensi dan upaya untuk mencari sumber daya yang tidak digunakan harus mencerminkan potensi penghematan, sehingga akun dengan biaya kecil harus dianalisis lebih jarang daripada akun dengan biaya yang lebih besar. Pencarian dan peristiwa penonaktifan bisa diinisiasi oleh perubahan status pada beban kerja, seperti produk yang mendekati akhir masa pakai atau mengalami penggantian. Pencarian dan peristiwa penonaktifan mungkin juga diinisiasi oleh peristiwa eksternal, seperti perubahan kondisi pasar atau penghentian produk.

#### Langkah-langkah implementasi

- Penonaktifan sumber daya: Ini adalah tahap depresiasi sumber daya AWS yang tidak lagi diperlukan atau telah berakhirnya perjanjian lisensi. Selesaikan semua pemeriksaan akhir sebelum beralih ke tahap penghapusan dan menonaktifkan sumber daya untuk mencegah gangguan yang tidak diinginkan seperti mengambil snapshot atau cadangan. Nonaktifkan setiap sumber daya yang telah teridentifikasi tidak terpakai menggunakan proses penonaktifan.

#### Sumber daya

Dokumen terkait:

- AWS Auto Scaling
- AWS Trusted Advisor

Contoh terkait:

- Lab Well-Architected: Menonaktifkan sumber daya (Level 100)

COST04-BP04 Sumber daya penonaktifan secara otomatis

Rancang beban kerja Anda agar menangani pengakhiran sumber daya secara anggun ketika Anda mengidentifikasi dan menonaktifkan sumber daya non-kritis, sumber daya yang tidak diperlukan, atau sumber daya dengan pemanfaatan yang rendah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Gunakan otomatisasi untuk mengurangi atau menyingkirkan biaya terkait untuk proses penonaktifan. Dengan merancang beban kerja agar menjalankan penonaktifan otomatis, Anda akan mengurangi biaya beban kerja secara keseluruhan selama masa pakainya. Anda dapat menggunakan Amazon EC2 Auto Scaling atau Application Auto Scaling untuk melakukan proses penonaktifan. Anda juga dapat menerapkan kode kustom menggunakan APIatau SDK untuk menonaktifkan sumber daya beban kerja secara otomatis.

Aplikasi modern dibangun tanpa server terlebih dahulu, strategi yang memprioritaskan adopsi layanan tanpa server. AWS mengembangkan layanan tanpa server untuk ketiga lapisan tumpukan Anda: komputasi, integrasi, dan penyimpanan data. Menggunakan arsitektur nirserver, Anda dapat menghemat biaya selama periode lalu lintas rendah dengan menaikkan dan menurunkan skala secara otomatis.

Langkah-langkah implementasi

- Terapkan Amazon EC2 Auto Scaling atau Application Auto Scaling: Untuk sumber daya yang didukung, konfigurasikan dengan Amazon Auto Scaling atau Application Auto EC2 Scaling. Layanan ini dapat membantu Anda mengoptimalkan pemanfaatan dan efisiensi biaya saat mengonsumsi AWS layanan. Ketika permintaan menurun, layanan-layanan ini akan menghapus kelebihan kapasitas sumber daya secara otomatis sehingga Anda dapat terhindar dari pengeluaran yang berlebihan.
- Konfigurasi CloudWatch untuk menghentikan instance: Instans dapat dikonfigurasi untuk dihentikan menggunakan alarm. CloudWatch Menggunakan metrik dari proses penonaktifan, implementasikan alarm dengan tindakan Amazon Elastic Compute Cloud. Verifikasi operasi di lingkungan non-produksi sebelum peluncuran.
- Menerapkan kode dalam beban kerja: Anda dapat menggunakan AWS SDK atau AWS CLI untuk menonaktifkan sumber daya beban kerja. Menerapkan kode dalam aplikasi yang terintegrasi dengan AWS dan mengakhiri atau menghapus sumber daya yang tidak lagi digunakan.

- Gunakan layanan tanpa server: Prioritaskan pembuatan arsitektur tanpa server dan arsitektur berbasis peristiwa untuk membangun dan menjalankan aplikasi Anda. AWS AWS menawarkan beberapa layanan teknologi tanpa server yang secara inheren menyediakan pemanfaatan sumber daya yang dioptimalkan secara otomatis dan penonaktifan otomatis (skala masuk dan skala keluar). Dengan aplikasi nirserver, pemanfaatan sumber daya dioptimalkan secara otomatis dan Anda tidak pernah membayar pengadaan yang berlebihan.

Sumber daya

Dokumen terkait:

- EC2Auto Scaling Amazon
- Memulai dengan Amazon EC2 Auto Scaling
- Penskalaan Otomatis Aplikasi
- AWS Trusted Advisor
- Tanpa server di AWS
- Buat Alarm untuk Menghentikan, Mengakhiri, Menyalakan Ulang, atau Memulihkan sebuah Instans
- Menambahkan tindakan penghentian ke alarm Amazon CloudWatch

Contoh terkait:

- Menjadwalkan penghapusan otomatis tumpukan AWS CloudFormation
- Lab Well-Architected: Menonaktifkan sumber daya secara otomatis (Level 100)
- Pembersihan AWS Mobil Servian

COST04-BP05 Menegakkan kebijakan penyimpanan data

Tetapkan kebijakan retensi data pada sumber daya yang didukung untuk menangani penghapusan objek sesuai persyaratan organisasi Anda. Identifikasi dan hapus sumber daya yang tidak diperlukan atau tidak digunakan dan objek yang sudah tidak diperlukan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Gunakan kebijakan retensi data dan kebijakan siklus hidup guna mengurangi biaya terkait untuk proses penonaktifan dan biaya penyimpanan untuk sumber daya yang diidentifikasi. Penetapan kebijakan retensi data dan kebijakan siklus hidup untuk menjalankan migrasi dan penghapusan

kelas penyimpanan otomatis akan mengurangi keseluruhan biaya penyimpanan di sepanjang masa pakainya. Anda dapat menggunakan Amazon Data Lifecycle Manager untuk mengotomatiskan pembuatan dan penghapusan snapshot Amazon Elastic Block Store dan Amazon Machine Images () yang didukung Amazon, AMIs dan menggunakan EBS Amazon S3 Intelligent-Tiering atau konfigurasi siklus hidup Amazon S3 untuk mengelola siklus hidup objek Amazon S3 Anda. Anda juga dapat menerapkan kode kustom menggunakan APIatau SDK untuk membuat kebijakan siklus hidup dan aturan kebijakan untuk objek yang akan dihapus secara otomatis.

Langkah-langkah implementasi

- Menggunakan Amazon Data Lifecycle Manager: Gunakan kebijakan siklus hidup di Amazon Data Lifecycle Manager untuk mengotomatiskan penghapusan snapshot Amazon dan Amazon yang didukung. EBS EBS AMIs
- Siapkan konfigurasi siklus hidup di bucket: Gunakan konfigurasi siklus hidup Amazon S3 di sebuah bucket untuk menentukan tindakan yang harus dilakukan Amazon S3 selama siklus hidup objek, serta penghapusan di akhir siklus hidup objek, berdasarkan kebutuhan bisnis Anda.

Sumber daya

Dokumen terkait:

- AWS Trusted Advisor
- Amazon Data Lifecycle Manager
- Cara untuk membuat konfigurasi siklus hidup bucket Amazon S3.

Video terkait:

- Otomatiskan EBS Snapshot Amazon dengan Amazon Data Lifecycle Manager
- Kosongkan bucket Amazon S3 menggunakan peraturan konfigurasi siklus hidup

Contoh terkait:

- Kosongkan bucket Amazon S3 menggunakan peraturan konfigurasi siklus hidup
- Lab Well-Architected: Menonaktifkan sumber daya secara otomatis (Level 100)

# Sumber daya hemat biaya

Pertanyaan

- COST5. Bagaimana cara mengevaluasi biaya ketika Anda memilih layanan?
- COST6. Bagaimana cara memenuhi target biaya ketika Anda memilih jenis, ukuran, dan jumlah sumber daya?
- COST7. Bagaimana cara menggunakan model harga untuk mengurangi biaya?
- COST8. Bagaimana cara Anda merencanakan biaya transfer data?

## COST5. Bagaimana cara mengevaluasi biaya ketika Anda memilih layanan?

AmazonEC2, AmazonEBS, dan Amazon S3 adalah layanan blok bangunan AWS . Layanan terkelola, seperti Amazon RDS dan Amazon DynamoDB, adalah layanan tingkat yang lebih tinggi, atau tingkat aplikasi. AWS Dengan memilih blok penyusun dan layanan terkelola yang sesuai, Anda dapat mengoptimalkan biaya beban kerja ini. Contohnya, dengan menggunakan layanan terkelola, Anda dapat mengurangi atau menghilangkan sebagian besar dari biaya tambahan untuk administrasi dan operasi, sehingga Anda bebas untuk mengerjakan aplikasi dan aktivitas yang terkait dengan bisnis.

Praktik terbaik

- COST05-BP01 Identifikasi persyaratan organisasi untuk biaya
- COST05-BP02 Menganalisis semua komponen beban kerja
- COST05-BP03 Melakukan analisis menyeluruh dari setiap komponen
- COST05-BP04 Pilih perangkat lunak dengan lisensi hemat biaya
- COST05-BP05 Pilih komponen beban kerja ini untuk mengoptimalkan biaya sesuai dengan prioritas organisasi
- COST05-BP06 Lakukan analisis biaya untuk penggunaan yang berbeda dari waktu ke waktu

COST05-BP01 Identifikasi persyaratan organisasi untuk biaya

Bekerja dengan anggota tim untuk menentukan keseimbangan antara pengoptimalan biaya dan pilar lainnya, seperti keandalan dan performa, untuk beban kerja ini.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Di sebagian besar organisasi, departemen teknologi informasi (IT) terdiri dari beberapa tim kecil, masing-masing dengan agenda dan area fokusnya sendiri, yang mencerminkan spesialisasi dan keterampilan anggota timnya. Anda perlu memahami tujuan keseluruhan, prioritas, dan sasaran organisasi Anda, serta bagaimana setiap departemen atau proyek berkontribusi terhadap tujuan ini. Mengategorikan semua sumber daya penting, termasuk personel, peralatan, teknologi, bahan, dan layanan eksternal, sangat penting untuk mencapai tujuan organisasi dan perencanaan anggaran yang komprehensif. Mengadopsi pendekatan sistematis terhadap identifikasi dan pemahaman biaya ini merupakan hal yang sangat penting untuk menyusun rencana biaya yang realistis dan matang untuk organisasi.

Ketika memilih layanan untuk beban kerja Anda, penting bagi Anda untuk memahami prioritas organisasi Anda. Buat keseimbangan antara optimasi biaya dan pilar AWS Well-Architected Framework lainnya, seperti kinerja dan keandalan. Proses ini harus dilakukan secara sistematis dan teratur untuk mencerminkan perubahan dalam tujuan organisasi, kondisi pasar, dan dinamika operasional. Beban kerja yang biayanya dioptimalkan penuh adalah solusi yang paling selaras dengan persyaratan organisasi Anda, tidak selalu berarti biaya yang paling rendah. Bertemulah dengan semua tim dalam organisasi Anda, seperti tim produk, bisnis, teknis, dan keuangan, untuk mengumpulkan informasi. Evaluasi dampak kompromi antar kepentingan yang bertentangan atau pendekatan alternatif, untuk membantu mengambil keputusan yang lebih tepat saat menentukan ke mana upaya perlu difokuskan atau saat memilih rencana tindakan.

Misalnya, meningkatkan kecepatan masuk pasar untuk fitur baru dapat diprioritaskan daripada optimalisasi biaya, atau Anda bisa memilih basis data relasional untuk data non-relasional guna menyederhanakan upaya migrasi sistem, dibandingkan bermigrasi ke basis data yang dioptimalkan untuk tipe data Anda dan memperbarui aplikasi Anda.

Langkah-langkah implementasi

- Identifikasi kebutuhan biaya organisasi: Bertemulah dengan anggota-anggota tim dari organisasi Anda, termasuk mereka dari tim pengelolaan produk, pemilik aplikasi, tim pengembangan dan operasional, serta peran manajemen dan keuangan. Prioritaskan pilar Well-Architected untuk beban kerja ini dan komponennya. Output-nya harus berupa daftar pilar secara berurutan. Anda juga dapat menambahkan bobot pada masing-masing pilar untuk menunjukkan berapa fokus tambahan yang dimiliki sebuah pilar, atau seberapa serupa fokus antara dua pilar.
- Tangani hutang teknis dan dokumentasikan: Selama peninjauan beban kerja, tangani hutang teknis. Dokumentasikan item backlog untuk mempertahankan beban kerja pada masa mendatang,

dengan tujuan memfaktor ulang atau merancang ulang untuk mengoptimalkannya lebih lanjut. Sangat penting untuk secara jelas mengkomunikasikan kompromi yang dilakukan kepada pemangku kepentingan lainnya.

Sumber daya

Praktik-praktik terbaik terkait:

- REL11-BP07 Arsitek produk Anda untuk memenuhi target ketersediaan dan perjanjian tingkat layanan uptime () SLAs
- OPS01-BP06 Mengevaluasi pengorbanan

Dokumen terkait:

- AWS Total Biaya Kepemilikan (TCO) Kalkulator
- Kelas penyimpanan Amazon S3
- Produk cloud

COST05-BP02 Menganalisis semua komponen beban kerja

Verifikasi bahwa setiap beban kerja telah dianalisis, terlepas dari ukuran atau biaya saat ini. Upaya peninjauan harus menggambarkan manfaat potensial, seperti biaya saat ini dan yang diperkirakan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Komponen beban kerja, yang dirancang untuk memberikan nilai bisnis kepada organisasi, dapat mencakup berbagai layanan. Untuk setiap komponen, seseorang dapat memilih AWS Cloud layanan khusus untuk memenuhi kebutuhan bisnis. Pilihan ini dapat dipengaruhi oleh faktor-faktor seperti pemahaman atau pengalaman sebelumnya dalam menggunakan layanan-layanan tersebut.

Setelah mengidentifikasi persyaratan organisasi Anda seperti yang disebutkan dalam COST05-BP01 Identifikasi persyaratan organisasi untuk biaya, lakukan analisis menyeluruh pada semua komponen dalam beban kerja Anda. Analisis setiap komponen dengan mempertimbangkan biaya dan ukuran saat ini serta proyeksinya. Pertimbangkan biaya analisis terhadap potensi penghematan beban kerja selama siklus hidupnya. Upaya yang dikeluarkan untuk menganalisis semua komponen beban kerja ini harus sesuai dengan potensi penghematan atau peningkatan yang diantisipasi

dari optimalisasi komponen spesifik tersebut. Misalnya, apabila biaya sumber daya yang diajukan adalah 10 USD per bulan, dan berdasarkan beban yang diperkirakan, tidak akan melebihi 15 USD per bulan, mengerahkan usaha seharian penuh untuk mengurangi biaya hingga 50% (lima dolar per bulan) dapat melampaui manfaat potensial selama masa pakai sistem. Gunakan perkiraan berdasarkan data yang lebih cepat dan efisien untuk memberikan hasil terbaik secara keseluruhan untuk komponen ini.

Beban kerja dapat berubah seiring waktu, dan rangkaian layanan yang tepat dapat menjadi tidak optimal jika penggunaan atau arsitektur beban kerja berubah. Analisis pilihan layanan harus menggabungkan tingkat penggunaan dan status beban kerja saat ini serta di masa mendatang. Mengimplementasikan layanan untuk penggunaan atau status beban kerja di masa mendatang dapat menghemat biaya keseluruhan dengan meminimalkan atau tanpa memerlukan usaha untuk membuat perubahan di masa mendatang. Misalnya, menggunakan EMR Tanpa Server mungkin merupakan pilihan yang tepat pada awalnya. Namun, karena konsumsi untuk layanan itu meningkat, transisi ke EMR on EC2 dapat mengurangi biaya untuk komponen beban kerja tersebut.

AWS Cost Explorerdan AWS Cost and Usage Report s (CUR) dapat menganalisis biaya bukti konsep (PoC) atau lingkungan berjalan. Anda juga dapat menggunakan AWS Pricing Calculator untuk memperkirakan biaya beban kerja.

Tulis alur kerja yang perlu diikuti oleh tim teknis untuk meninjau beban kerja mereka. Jaga agar alur kerja ini tetap sederhana, tetapi juga mencakup semua langkah yang diperlukan untuk memastikan tim memahami setiap komponen beban kerja beserta harganya. Organisasi Anda kemudian dapat mengikuti dan menyesuaikan alur kerja ini berdasarkan kebutuhan khusus setiap tim.

1. Buat daftar setiap layanan yang digunakan untuk beban kerja Anda: Ini adalah titik awal yang baik. Identifikasi semua layanan yang sedang digunakan dan dari mana biaya berasal.
2. Pahami cara kerja penetapan harga untuk layanan tersebut: Pahami model penetapan harga dari masing-masing layanan. AWS Layanan yang berbeda memiliki model harga yang berbeda berdasarkan faktor-faktor seperti volume penggunaan, transfer data, dan harga khusus fitur.
3. Fokus pada layanan yang memiliki biaya beban kerja yang tidak terduga dan yang tidak sesuai dengan penggunaan yang diharapkan dan hasil bisnis Anda: Identifikasi outlier atau layanan di mana biaya tidak sebanding dengan nilai atau penggunaan dengan penggunaan atau s. AWS Cost Explorer AWS Cost and Usage Report Untuk memprioritaskan upaya optimalisasi, penting untuk mengaitkan biaya dengan hasil bisnis.
4. AWS Cost Explorer, CloudWatch Log, Log VPC Aliran, dan Lensa Penyimpanan Amazon S3 untuk memahami akar penyebab biaya tinggi tersebut: Alat ini berperan penting dalam diagnosis biaya tinggi. Setiap layanan menawarkan lensa yang berbeda untuk melihat dan menganalisis

penggunaan dan biaya. Misalnya, Cost Explorer membantu menentukan tren biaya secara keseluruhan, CloudWatch Log memberikan wawasan operasional, VPC Flow Logs menampilkan lalu lintas IP, dan Amazon S3 Storage Lens berguna untuk analitik penyimpanan.

5. Gunakan AWS Budgets untuk menetapkan anggaran untuk jumlah tertentu untuk layanan atau akun: Menetapkan anggaran adalah cara proaktif untuk mengelola biaya. Gunakan AWS Budgets untuk menetapkan ambang anggaran khusus dan menerima peringatan ketika biaya melebihi ambang batas tersebut.
6. Konfigurasikan CloudWatch alarm Amazon untuk mengirim peringatan penagihan dan penggunaan: Siapkan pemantauan dan peringatan untuk metrik biaya dan penggunaan. CloudWatch alarm dapat memberi tahu Anda ketika ambang batas tertentu dilanggar, yang meningkatkan waktu respons intervensi.

Fasilitasi penyempurnaan dan penghematan keuangan yang jelas dari waktu ke waktu melalui tinjauan strategis pada semua komponen beban kerja dan terlepas dari atributnya saat ini. Upaya yang dicurahkan dalam proses tinjauan ini harus terencana, dengan pertimbangan yang cermat terhadap potensi keuntungan yang mungkin direalisasikan.

Langkah-langkah implementasi

- Buat daftar komponen beban kerja: Buat daftar komponen beban kerja Anda. Gunakan daftar ini untuk memverifikasi bahwa setiap komponen telah dianalisis. Upaya yang dilakukan harus sesuai dengan kekritisan beban kerja sesuai prioritas organisasi Anda. Kelompokkan sumber daya menurut fungsinya untuk meningkatkan efisiensi (misalnya, penyimpanan basis data produksi, jika terdapat beberapa basis data).
- Prioritaskan daftar komponen: Ambil daftar komponen dan prioritaskan dalam urutan usaha. Daftar tersebut umumnya diurutkan berdasarkan biaya komponen, dari yang paling mahal ke yang paling murah, atau diurutkan sesuai kekritisan sebagaimana ditentukan oleh prioritas organisasi Anda.
- Lakukan analisis: Untuk setiap komponen di dalam daftar, tinjau opsi dan layanan yang tersedia kemudian pilih opsi yang paling sesuai dengan prioritas organisasi Anda.

Sumber daya

Dokumen terkait:

- AWS Pricing Calculator
- AWS Cost Explorer

- Kelas penyimpanan Amazon S3
- Produk AWS Cloud

Video terkait:

- AWS Seri Optimalisasi Biaya: CloudWatch

COST05-BP03 Melakukan analisis menyeluruh dari setiap komponen

Lihat keseluruhan biaya organisasi dari setiap komponen. Hitung total biaya kepemilikan dengan mempertimbangkan faktor biaya operasi dan manajemen, terutama jika menggunakan layanan terkelola oleh penyedia cloud. Hasil upaya peninjauan harus menggambarkan manfaat potensial (misalnya, waktu yang digunakan untuk menganalisis sebanding dengan biaya komponen).

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Pertimbangkan waktu yang dapat dihemat yang memungkinkan tim Anda untuk berfokus pada penghentian utang teknis, inovasi, fitur yang menambah nilai, dan membangun sesuatu yang membedakan bisnis Anda dari yang lain. Misalnya, Anda mungkin perlu mengangkat dan menggeser (juga disebut host ulang) basis data Anda dari lingkungan on-premise Anda ke cloud secepat mungkin dan kemudian mengoptimalkannya. Sebaiknya cari tahu kemungkinan penghematan yang diperoleh menggunakan layanan terkelola di AWS yang dapat menghilangkan atau mengurangi biaya lisensi. Layanan terkelola AWS menghilangkan beban operasional dan administrasi pemeliharaan layanan, seperti menambal atau meningkatkan OS, dan memungkinkan Anda untuk fokus pada inovasi dan bisnis.

Karena layanan terkelola beroperasi dengan skala cloud, biaya yang ditawarkan per transaksi atau layanan dapat lebih rendah. Anda dapat melakukan optimalisasi potensial untuk mencapai beberapa manfaat nyata, tanpa mengubah arsitektur inti aplikasi. Misalnya, Anda mungkin ingin mengurangi jumlah waktu yang Anda habiskan untuk mengelola instance database dengan bermigrasi ke database-as-a-service platform seperti Amazon Relational Database Service (AmazonRDS) atau memigrasikan aplikasi Anda ke platform yang dikelola sepenuhnya seperti. AWS Elastic Beanstalk

Biasanya, layanan terkelola memiliki atribut yang dapat Anda atur untuk memastikan kapasitas yang memadai. Anda harus mengatur dan memantau atribut ini agar kapasitas Anda yang berlebih diminimalkan dan kinerja dimaksimalkan. Anda dapat memodifikasi atribut AWS Managed Services

menggunakan AWS Management Console atau AWS APIs dan SDKs untuk menyelaraskan kebutuhan sumber daya dengan permintaan yang berubah. Misalnya, Anda dapat menambah atau mengurangi jumlah node di EMR klaster Amazon (atau cluster Amazon Redshift) untuk skala atau masuk.

Anda juga dapat mengemas beberapa instance pada AWS sumber daya untuk mengaktifkan penggunaan kepadatan yang lebih tinggi. Misalnya, Anda dapat menyediakan beberapa database kecil pada satu instance database Amazon Relational Database Service (RDSAmazon). Seiring bertambahnya penggunaan, Anda dapat memigrasikan salah satu database ke instans RDS database Amazon khusus menggunakan proses snapshot dan restore.

Ketika menyiapkan beban kerja dalam layanan terkelola, Anda harus memahami persyaratan untuk menyesuaikan kapasitas layanan. Persyaratan ini biasanya berupa waktu, upaya, dan dampak apa pun terhadap operasi beban kerja normal. Sumber daya yang disiapkan harus memberikan waktu untuk perubahan apa pun, siapkan biaya tambahan yang diperlukan untuk melakukan hal ini. Upaya berkelanjutan yang diperlukan untuk memodifikasi layanan dapat dikurangi menjadi hampir nol dengan menggunakan APIs dan SDKs yang terintegrasi dengan sistem dan alat pemantauan, seperti Amazon CloudWatch.

Amazon RDS, Amazon Redshift, dan Amazon ElastiCache menyediakan layanan database terkelola. Amazon Athena, AmazonEMR, dan Amazon OpenSearch Service menyediakan layanan analitik terkelola.

AMSadalah layanan yang mengoperasikan AWS infrastruktur atas nama pelanggan dan mitra perusahaan. Layanan ini menyediakan lingkungan yang aman dan patuh untuk melakukan deployment beban kerja Anda. AMSmenggunakan model operasi cloud perusahaan dengan otomatisasi untuk memungkinkan Anda memenuhi persyaratan organisasi Anda, pindah ke cloud lebih cepat, dan mengurangi biaya manajemen yang sedang berlangsung.

Langkah-langkah implementasi

- Lakukan analisis yang menyeluruh: Dengan menggunakan daftar komponen tersebut, kerjakan setiap komponen mulai dari prioritas tertinggi ke prioritas yang terendah. Untuk komponen yang diprioritaskan dan membutuhkan biaya mahal, jalankan analisis tambahan dan evaluasi semua opsi yang ada serta dampaknya dalam jangka panjang. Untuk komponen dengan prioritas rendah, ukur apakah perubahan penggunaan akan mengubah prioritas komponen, kemudian jalankan analisis upaya yang sesuai.
- Bandingkan sumber daya yang dikelola dan tidak dikelola: Pertimbangkan biaya operasional untuk sumber daya yang Anda kelola dan bandingkan dengan sumber daya yang AWS dikelola.

Misalnya, tinjau database Anda yang berjalan di EC2 instans Amazon dan bandingkan dengan RDS opsi Amazon (layanan AWS terkelola) atau Amazon EMR dibandingkan dengan menjalankan Apache Spark di Amazon. EC2 Saat beralih dari beban kerja yang dikelola sendiri ke beban kerja yang dikelola AWS sepenuhnya, teliti opsi Anda dengan cermat. Tiga faktor terpenting yang perlu dipertimbangkan adalah jenis layanan terkelola yang ingin Anda gunakan, proses yang akan Anda gunakan untuk memigrasikan data Anda dan memahami model tanggung jawab bersama AWS.

Sumber daya

Dokumen terkait:

- AWS Total Biaya Kepemilikan (TCO) Kalkulator
- Kelas penyimpanan Amazon S3
- Produk AWS Cloud
- AWS Model Tanggung Jawab Bersama

Video terkait:

- Mengapa pindah ke basis data terkelola?
- Apa itu Amazon EMR dan bagaimana saya bisa menggunakannya untuk memproses data?

Contoh terkait:

- Mengapa harus berpindah ke basis data terkelola
- Mengkonsolidasikan data dari database SQL Server identik ke dalam satu database Amazon RDS untuk SQL Server menggunakan AWS DMS
- Kirimkan data dalam skala besar ke Amazon Managed Streaming for Apache Kafka (Amazon) MSK
- Migrasi an. ASP NETaplikasi web untuk AWS Elastic Beanstalk

COST05-BP04 Pilih perangkat lunak dengan lisensi hemat biaya

Perangkat lunak sumber terbuka meniadakan biaya lisensi perangkat lunak yang dapat menambah biaya yang besar pada beban kerja. Jika perangkat lunak berlisensi diperlukan, hindari lisensi yang terikat pada atribut arbitrer sepertiCPUs, cari lisensi yang terikat pada output atau hasil. Besar kecilnya biaya lisensi ini lebih sesuai dengan manfaat yang disediakan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Sumber terbuka berasal dari konteks pengembangan perangkat lunak untuk menunjukkan bahwa perangkat lunak tersebut memenuhi kriteria distribusi gratis tertentu. Perangkat lunak sumber terbuka terdiri dari kode sumber yang dapat diperiksa, dimodifikasi, dan disempurnakan oleh siapa pun. Berdasarkan persyaratan bisnis, keterampilan insinyur, perkiraan penggunaan, atau ketergantungan teknologi lainnya, organisasi dapat mempertimbangkan untuk menggunakan perangkat lunak open source AWS untuk meminimalkan biaya lisensi mereka. Dengan kata lain, biaya lisensi perangkat lunak dapat dikurangi melalui penggunaan perangkat lunak sumber terbuka. Penggunaan jenis perangkat lunak ini dapat memberikan pengaruh besar pada biaya beban kerja seiring berubahnya ukuran beban kerja.

Ukur manfaat perangkat lunak berlisensi terhadap total biaya untuk mengoptimalkan beban kerja Anda. Modelkan perubahan dalam lisensi dan bagaimana pengaruhnya terhadap biaya beban kerja Anda. Jika vendor mengubah biaya lisensi basis data Anda, selidiki bagaimana pengaruhnya terhadap keseluruhan efisiensi beban kerja Anda. Pertimbangkan riwayat pengumuman harga dari vendor Anda untuk mengetahui tren perubahan lisensi di seluruh produk mereka. Biaya lisensi juga dapat diskalakan secara independen dari throughput atau penggunaan, seperti lisensi yang diskalakan oleh perangkat keras (lisensi CPU terikat). Lisensi jenis ini harus dihindari karena biaya dapat meningkat pesat tanpa hasil yang seimbang.

Misalnya, mengoperasikan EC2 instans Amazon di us-east-1 dengan sistem operasi Linux memungkinkan Anda memangkas biaya sekitar 45%, dibandingkan dengan menjalankan instans EC2 Amazon lain yang berjalan di Windows.

Ini AWS Pricing Calculatormenawarkan cara komprehensif untuk membandingkan biaya berbagai sumber daya dengan opsi lisensi yang berbeda, seperti RDS instans Amazon dan mesin basis data yang berbeda. Selain itu, AWS Cost Explorer memberikan perspektif yang tak ternilai untuk biaya beban kerja yang ada, terutama yang datang dengan lisensi yang berbeda. Untuk manajemen lisensi, AWS License Manager menawarkan metode efisien untuk mengawasi dan menangani lisensi perangkat lunak. Pelanggan dapat menerapkan dan mengoperasionalkan perangkat lunak sumber terbuka pilihan mereka di AWS Cloud.

Langkah-langkah implementasi

- Analisis opsi lisensi: Tinjau persyaratan lisensi perangkat lunak yang tersedia. Cari versi sumber terbuka yang memiliki fungsionalitas yang diperlukan, dan cari tahu apakah manfaat dari perangkat

lunak berlisensi lebih besar daripada biayanya. Ketentuan yang menguntungkan adalah yang menyelaraskan biaya dengan manfaat yang disediakan.

- Analisis penyedia perangkat lunak: Tinjau riwayat penetapan harga atau perubahan lisensi dari vendor. Cari perubahan yang tidak selaras dengan hasil, seperti ketentuan merugikan yang mengharuskan perangkat lunak dijalankan di perangkat keras atau platform vendor tertentu. Selain itu, cari tahu bagaimana mereka melakukan audit, dan sanksi yang dapat dikenakan.

Sumber daya

Dokumen terkait:

- Open Source di AWS
- AWS Total Biaya Kepemilikan (TCO) Kalkulator
- Kelas penyimpanan Amazon S3
- Produk cloud

Contoh terkait:

- Blog Sumber Terbuka
- AWS Blog Sumber Terbuka
- Evaluasi Optimalisasi dan Pemberian Lisensi

COST05-BP05 Pilih komponen beban kerja ini untuk mengoptimalkan biaya sesuai dengan prioritas organisasi

Pertimbangkan biaya saat memilih semua komponen untuk beban kerja Anda. Termasuk di antaranya adalah menggunakan layanan terkelola dan tingkat aplikasi atau nirserver, kontainer, atau arsitektur yang berbasis peristiwa agar dapat menekan keseluruhan biaya. Minimalkan biaya lisensi menggunakan perangkat lunak sumber terbuka, perangkat lunak yang tidak memiliki biaya lisensi, atau alternatif untuk menekan biaya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Pertimbangkan biaya layanan dan opsi saat memilih semua komponen. Ini termasuk menggunakan tingkat aplikasi dan layanan terkelola, seperti Amazon Relational Database Service (Amazon), RDS

Amazon DynamoDB, Amazon Simple Notification Service (Amazon), dan SNS Amazon Simple Email Service (SESAmazon) untuk mengurangi biaya organisasi secara keseluruhan.

Gunakan layanan nirserver dan kontainer untuk komputasi, seperti AWS Lambda dan Amazon Simple Storage Service (Amazon S3) untuk situs web statis. Kontainerisasi aplikasi Anda jika memungkinkan dan gunakan AWS Managed Container Services seperti Amazon Elastic Container Service (AmazonECS) atau Amazon Elastic Kubernetes Service (Amazon). EKS

Minimalkan biaya lisensi dengan menggunakan perangkat lunak sumber terbuka, atau perangkat lunak yang tidak memiliki ongkos lisensi (misalnya Amazon Linux untuk beban kerja komputasi atau migrasikan basis data ke Amazon Aurora).

Anda dapat menggunakan layanan tanpa server atau tingkat aplikasi seperti Lambda, Amazon Simple Queue Service (Amazon), AmazonSQS, dan Amazon. SNS SES Semua layanan ini menyingkirkan kebutuhan Anda untuk mengelola sumber daya, dan menyediakan fungsi eksekusi kode, layanan pengantrean, dan pengiriman pesan. Manfaat lain layanan-layanan ini adalah menskalakan kinerja dan biaya sesuai dengan penggunaan, sehingga memungkinkan alokasi dan atribusi biaya yang efisien.

Menggunakan arsitektur berbasis peristiwa juga dimungkinkan dengan layanan nirserver. Arsitektur yang didorong peristiwa didasarkan pada push, sehingga semuanya terjadi sesuai permintaan saat peristiwa muncul di dalam router. Dengan demikian, Anda tidak akan membayar polling yang terjadi terus-menerus untuk memeriksa peristiwa. Ini berarti lebih sedikit konsumsi bandwidth jaringan, CPU pemanfaatan yang lebih sedikit, kapasitas armada yang tidak aktif, dan lebih SSL TLS sedikit/jabat tangan.

Untuk informasi lebih lanjut tentang layanan nirserver, lihat Laporan resmi lensa Aplikasi Well-Architected.

Langkah-langkah implementasi

- Pilih setiap layanan untuk mengoptimalisasi biaya: Dengan menggunakan daftar dan analisis yang telah Anda prioritaskan, pilih setiap opsi yang menyediakan pilihan terbaik sesuai prioritas organisasi Anda. Alih-alih meningkatkan kapasitas untuk memenuhi permintaan, pertimbangkan opsi-opsi lain yang dapat memberi Anda kinerja yang lebih baik dengan biaya yang lebih rendah. Misalnya, jika Anda perlu meninjau lalu lintas yang diharapkan untuk database Anda AWS, pertimbangkan untuk meningkatkan ukuran instans atau menggunakan ElastiCache layanan Amazon (Redis atau Memcached) untuk menyediakan mekanisme cache untuk database Anda.

- Evaluasi arsitektur berbasis peristiwa: Menggunakan arsitektur nirserver juga memungkinkan Anda membangun arsitektur berbasis peristiwa untuk aplikasi berbasis layanan mikro yang terdistribusi, yang membantu Anda membangun solusi yang dapat diskalakan, tangguh, gesit, dan hemat biaya.

Sumber daya

Dokumen terkait:

- AWS Total Biaya Kepemilikan (TCO) Kalkulator
- Nirserver AWS
- Apa yang Dimaksud Arsitektur Berbasis Peristiwa?
- Kelas penyimpanan Amazon S3
- Produk cloud
- Amazon ElastiCache (Redis) OSS

Contoh terkait:

- Mulai menggunakan arsitektur berbasis peristiwa
- Arsitektur berbasis peristiwa
- Bagaimana Statsig menjalankan 100x lebih hemat biaya menggunakan Amazon (Redis) ElastiCache OSS
- Praktik terbaik untuk bekerja dengan AWS Lambda fungsi

COST05-BP06 Lakukan analisis biaya untuk penggunaan yang berbeda dari waktu ke waktu

Beban kerja bisa berubah seiring waktu. Beberapa layanan atau fitur lebih hemat biaya pada tingkat penggunaan yang berbeda. Dengan melakukan analisis pada setiap komponen dari waktu ke waktu serta pada penggunaan yang diperkirakan, beban kerja tetap hemat biaya di sepanjang masa pakainya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Saat AWS merilis layanan dan fitur baru, layanan optimal untuk beban kerja Anda dapat berubah. Upaya yang diperlukan harus mencerminkan keuntungan potensial. Frekuensi peninjauan beban kerja tergantung pada persyaratan organisasi Anda. Jika beban kerja memiliki biaya yang signifikan,

penerapan layanan baru lebih dini akan memaksimalkan penghematan biaya, sehingga manfaatnya bisa lebih besar jika peninjauan lebih sering dilakukan. Inisiasi peninjauan lainnya adalah perubahan pola penggunaan. Perubahan yang signifikan pada penggunaan bisa menandakan bahwa layanan alternatif akan lebih optimal.

Jika Anda perlu memindahkan data ke dalam AWS Cloud, Anda dapat memilih berbagai macam AWS penawaran layanan dan alat mitra untuk membantu Anda memigrasikan kumpulan data Anda, apakah itu file, database, gambar mesin, volume blok, atau bahkan backup tape. Misalnya, untuk memindahkan sejumlah besar data ke dan dari AWS atau memproses data di tepi, Anda dapat menggunakan salah satu perangkat yang AWS dibuat khusus untuk memindahkan petabyte data secara offline secara efektif. Contoh lain adalah untuk kecepatan transfer data yang lebih tinggi, layanan koneksi langsung mungkin lebih murah daripada VPN yang menyediakan konektivitas konsisten yang diperlukan untuk bisnis Anda.

Berdasarkan analisis biaya untuk berbagai penggunaan seiring waktu, tinjau aktivitas penskalaan Anda. Analisis hasilnya untuk melihat apakah kebijakan penskalaan dapat disesuaikan untuk menambahkan instans dengan beberapa tipe instans dan opsi pembelian. Tinjau pengaturan Anda untuk melihat apakah pengaturan minimum dapat dikurangi untuk melayani permintaan pengguna tetapi dengan ukuran armada yang lebih kecil, dan tambahkan lebih banyak sumber daya untuk memenuhi permintaan tinggi yang diperkirakan.

Lakukan analisis biaya untuk penggunaan yang berbeda dari waktu ke waktu dengan berdiskusi dengan para pemangku kepentingan yang ada di organisasi Anda dan gunakan fitur perkiraan AWS Cost Explorer untuk memprediksi dampak yang mungkin terjadi dari adanya perubahan layanan. Pantau peluncuran tingkat penggunaan menggunakan AWS Budgets, CloudWatch menagih alarm dan AWS Cost Anomaly Detection untuk mengidentifikasi dan menerapkan layanan yang paling hemat biaya lebih cepat.

Langkah-langkah implementasi

- Tentukan pola penggunaan yang terprediksi:  Bersama organisasi Anda, seperti pemasaran dan pemilik produk, buatlah dokumentasi yang memuat pola penggunaan yang diharapkan dan diprediksi untuk beban kerja. Bersama pemangku kepentingan bisnis, diskusikan riwayat dan prakiraan peningkatan biaya dan penggunaan dan pastikan peningkatan tersebut sesuai dengan persyaratan bisnis. Identifikasi hari kalender, minggu, atau bulan di mana Anda mengharapkan lebih banyak pengguna untuk menggunakan AWS sumber daya Anda, yang menunjukkan bahwa Anda harus meningkatkan kapasitas sumber daya yang ada atau mengadopsi layanan tambahan untuk mengurangi biaya dan meningkatkan kinerja.

- Lakukan analisis biaya pada penggunaan yang terprediksi: Dengan menggunakan pola penggunaan yang sudah ditentukan, lakukan analisis pada masing-masing titik ini. Upaya analisis harus mencerminkan hasil potensial. Sebagai contoh, jika ada perubahan besar pada penggunaan, analisis yang mendalam harus dilakukan untuk memastikan biaya dan perubahan yang terjadi. Dengan kata lain, saat biaya meningkat, penggunaan untuk bisnis juga harus meningkat.

Sumber daya

Dokumen terkait:

- AWS Total Biaya Kepemilikan (TCO) Kalkulator
- Kelas penyimpanan Amazon S3
- Produk cloud
- EC2Auto Scaling Amazon
- Migrasi Data Cloud
- AWS Snow Family

Video terkait:

- AWS OpsHub for Snow Family

## COST6. Bagaimana cara memenuhi target biaya ketika Anda memilih jenis, ukuran, dan jumlah sumber daya?

Pastikan Anda memilih jumlah sumber daya dan ukuran sumber daya yang sesuai untuk tugas yang ada. Anda meminimalkan pemborosan dengan memilih jenis, ukuran, dan jumlah yang paling hemat.

Praktik terbaik

- COST06-BP01 Lakukan pemodelan biaya
- COST06-BP02 Pilih jenis sumber daya, ukuran, dan nomor berdasarkan data
- COST06-BP03 Pilih jenis sumber daya, ukuran, dan nomor secara otomatis berdasarkan metrik
- COST06-BP04 Pertimbangkan untuk menggunakan sumber daya bersama

COST06-BP01 Lakukan pemodelan biaya

Identifikasi kebutuhan organisasi (seperti kebutuhan bisnis dan komitmen yang ada) dan lakukan pemodelan biaya (keseluruhan biaya) untuk beban kerja serta setiap komponennya. Lakukan aktivitas tolok ukur untuk beban kerja di bawah berbagai beban yang diprediksi lalu bandingkan biayanya. Upaya pemodelan harus mencerminkan manfaat potensial. Misalnya, waktu yang digunakan sebanding dengan biaya komponen.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Lakukan pemodelan biaya untuk beban kerja Anda dan setiap komponennya guna memahami keseimbangan antar sumber daya, dan temukan ukuran yang tepat untuk setiap sumber daya di dalam beban kerja, dengan tingkat kinerja tertentu. Pemahaman tentang pertimbangan biaya dapat menjadi landasan untuk kasus bisnis dan proses pengambilan keputusan organisasi Anda saat mengevaluasi hasil realisasi nilai untuk deployment beban kerja yang direncanakan.

Lakukan aktivitas tolok ukur untuk beban kerja di bawah berbagai beban yang diprediksi lalu bandingkan biayanya. Hasil upaya pemodelan harus menggambarkan manfaat potensial, misalnya, waktu yang digunakan sebanding dengan biaya komponen atau penghematan yang diprediksi. Untuk praktik terbaik, lihat bagian Tinjauan dari Pilar Efisiensi Kinerja Kerangka Kerja AWS Well-Architected.

Sebagai contoh, untuk membuat pemodelan biaya untuk beban kerja yang terdiri dari sumber daya komputasi, AWS Compute Optimizer dapat membantu pemodelan biaya untuk menjalankan beban kerja. Layanan ini menyediakan rekomendasi penyesuaian ukuran untuk sumber daya komputasi berdasarkan riwayat penggunaan. Pastikan CloudWatch Agen disebarkan ke EC2 instans Amazon untuk mengumpulkan metrik memori yang membantu Anda dengan rekomendasi yang lebih akurat di dalamnya. AWS Compute Optimizer Ini adalah sumber data ideal untuk sumber daya komputasi karena ini adalah layanan gratis, yang menggunakan machine learning untuk memberikan beberapa rekomendasi tergantung tingkat risiko.

Ada beberapa layanan yang dapat Anda gunakan dengan log kustom sebagai sumber data untuk operasi rightsizing untuk layanan lain dan komponen beban kerja, seperti, AWS Trusted AdvisorAmazon dan CloudWatch Amazon Logs. CloudWatch AWS Trusted Advisor memeriksa sumber daya dan menandai sumber daya dengan pemanfaatan rendah yang dapat membantu Anda mengukur sumber daya dengan tepat dan membuat pemodelan biaya.

Berikut ini adalah beberapa rekomendasi untuk data dan metrik pemodelan biaya:

- Pemantauan harus mencerminkan pengalaman pengguna secara akurat. Pilih tingkat detail yang tepat untuk periode waktu dan dengan cermat pilih persentil maksimum atau ke-99, bukan rata-rata.
- Pilih tingkat detail yang tepat untuk periode waktu analisis yang diperlukan untuk mencakup siklus beban kerja apa pun. Sebagai contoh, jika dilakukan analisis dua minggu, Anda mungkin mengabaikan siklus pemanfaatan tinggi bulanan, yang dapat menyebabkan pengadaan yang terlalu rendah.
- Pilih AWS layanan yang tepat untuk beban kerja yang Anda rencanakan dengan mempertimbangkan komitmen Anda yang ada, model penetapan harga yang dipilih untuk beban kerja lainnya, dan kemampuan untuk berinovasi lebih cepat dan fokus pada nilai bisnis inti Anda.

Langkah-langkah implementasi

- Lakukan pemodelan biaya untuk sumber daya: Deploy beban kerja atau bukti konsep ke dalam sebuah akun terpisah dengan jenis dan ukuran sumber daya tertentu untuk diuji. Jalankan beban kerja dengan data pengujian dan rekam hasil output, beserta data biaya untuk periode pengujian. Kemudian, deploy ulang beban kerja atau ubah tipe dan ukuran sumber daya lalu jalankan ulang pengujian. Sertakan biaya lisensi produk apa pun yang mungkin Anda gunakan dengan sumber daya ini serta perkiraan biaya operasi (tenaga kerja atau teknisi) untuk men-deploy dan mengelola sumber daya ini selama membuat pemodelan biaya. Pertimbangkan pemodelan biaya untuk periode tertentu (jam, harian, bulanan, tahunan, atau tiga tahun).

Sumber daya

Dokumen terkait:

- AWS Auto Scaling
- Mengidentifikasi Peluang untuk Menyesuaikan ke Ukuran yang Tepat
- CloudWatch Fitur Amazon
- Optimalisasi Biaya: Ukuran EC2 Tepat Amazon
- AWS Compute Optimizer
- AWS Kalkulator Harga

Contoh terkait:

- Lakukan Pemodelan Biaya Berbasis Data

- Perkirakan biaya konfigurasi AWS sumber daya yang direncanakan
- Pilih AWS alat yang tepat

COST06-BP02 Pilih jenis sumber daya, ukuran, dan nomor berdasarkan data

Pilih jenis atau ukuran sumber daya berdasarkan data tentang karakteristik sumber daya dan beban kerja. Misalnya, intensif komputasi, memori, throughput, atau tulis. Pilihan ini biasanya dibuat menggunakan beban kerja versi sebelumnya (on-premise), menggunakan dokumentasi, atau menggunakan sumber informasi lain untuk beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Amazon EC2 menyediakan berbagai pilihan jenis instans dengan berbagai tingkatCPU, memori, penyimpanan, dan kapasitas jaringan agar sesuai dengan kasus penggunaan yang berbeda. Jenis instans ini menampilkan perpaduan yang berbeda dariCPU, memori, penyimpanan, dan kemampuan jaringan, memberi Anda keserbagunaan saat memilih kombinasi sumber daya yang tepat untuk proyek Anda. Setiap jenis instans tersedia dalam berbagai ukuran, sehingga Anda dapat menyesuaikan sumber daya berdasarkan tuntutan beban kerja Anda. Untuk menentukan jenis instans yang Anda butuhkan, kumpulkan detail tentang persyaratan sistem aplikasi atau perangkat lunak yang akan Anda jalankan pada instans Anda. Detail ini sebaiknya mencakup hal-hal berikut:

- Sistem operasi
- Jumlah CPU core
- GPUinti
- Jumlah memori sistem (RAM)
- Jenis dan ruang penyimpanan
- Persyaratan bandwidth jaringan

Identifikasi tujuan persyaratan komputasi dan instance mana yang diperlukan, lalu jelajahi berbagai kelompok EC2 instans Amazon. Amazon menawarkan rangkaian jenis instans berikut:

- Tujuan Umum
- Komputasi yang Dioptimalkan
- Memori Dioptimalkan
- Penyimpanan Dioptimalkan

- Komputasi yang Dipercepat
- HPCDioptimalkan

Untuk pemahaman yang lebih dalam tentang tujuan spesifik dan kasus penggunaan yang dapat dipenuhi oleh keluarga EC2 instans Amazon tertentu, lihat Jenis AWS instans.

Pengumpulan persyaratan sistem menjadi langkah yang sangat penting bagi Anda untuk memilih rangkaian instans yang spesifik dan jenis instans yang paling sesuai dengan kebutuhan Anda. Nama jenis instans terdiri dari nama rangkaian dan ukuran instans. Misalnya, instans t2.micro berasal dari rangkaian T2 dan memiliki ukuran mikro.

Pilih jenis atau ukuran sumber daya berdasarkan karakteristik sumber daya dan beban kerja (misalnya, intensif komputasi, memori, throughput, atau tulis). Pilihan ini biasanya dibuat menggunakan pemodelan biaya, beban kerja versi sebelumnya (seperti versi on-premise), menggunakan dokumentasi, atau menggunakan sumber informasi lain tentang beban kerja (laporan resmi atau solusi yang dipublikasikan). Menggunakan kalkulator AWS harga atau alat manajemen biaya dapat membantu dalam membuat keputusan berdasarkan informasi tentang jenis, ukuran, dan konfigurasi instans.

Langkah-langkah implementasi

- Pilih sumber daya berdasarkan data: Gunakan data pemodelan biaya Anda untuk memilih tingkat penggunaan beban kerja yang diantisipasi, dan pilih jenis dan ukuran sumber daya yang tertentu. Mengandalkan data pemodelan biaya, menentukan jumlah virtualCPUs, total memori (GiB), volume penyimpanan instans lokal (GB), volume EBS Amazon, dan tingkat kinerja jaringan, dengan mempertimbangkan kecepatan transfer data yang diperlukan untuk instance tersebut. Selalu buat pilihan berdasarkan analisis mendetail dan data yang akurat untuk mengoptimalkan performa sekaligus mengelola biaya secara efektif.

Sumber daya

Dokumen terkait:

- AWS Tipe instans
- AWS Auto Scaling
- CloudWatch Fitur Amazon
- Optimalisasi Biaya: Ukuran EC2 yang Tepat

Video terkait:

- Memilih EC2 instans Amazon yang tepat untuk beban kerja Anda
- Sesuaikan layanan Anda ke ukuran yang tepat

Contoh terkait:

- Semakin mudah untuk menemukan dan membandingkan jenis EC2 instans Amazon

COST06-BP03 Pilih jenis sumber daya, ukuran, dan nomor secara otomatis berdasarkan metrik

Gunakan metrik dari beban kerja yang sedang berjalan untuk memilih ukuran dan jenis yang tepat untuk mengoptimalkan biaya. Sediakan throughput, ukuran, dan penyimpanan secara tepat untuk layanan komputasi, penyimpanan, data, dan jaringan. Hal ini dapat dilakukan dengan loop umpan balik seperti penskalaan otomatis atau dengan kode kustom di dalam beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Buat loop umpan balik di dalam beban kerja yang menggunakan metrik aktif dari beban kerja yang berjalan untuk melakukan perubahan pada beban kerja tersebut. Anda dapat menggunakan layanan terkelola, seperti AWS Auto Scaling, yang Anda konfigurasikan untuk melakukan operasi ukuran yang tepat untuk Anda. AWS juga menyediakan APIs, SDKs, dan fitur yang memungkinkan sumber daya untuk dimodifikasi dengan sedikit usaha. Anda dapat memprogram beban kerja ke stop-and-start EC2 instans Amazon untuk mengizinkan perubahan ukuran instans atau jenis instans. Hal ini menyediakan manfaat penyesuaian ukuran sambil menghilangkan hampir semua biaya operasional yang diperlukan untuk melakukan perubahan.

Beberapa AWS layanan telah dibangun dalam pemilihan tipe atau ukuran otomatis, seperti Amazon Simple Storage Service Intelligent-Tiering. Amazon S3 Intelligent-Tiering secara otomatis memindahkan data Anda antara dua tingkat akses, akses sering dan akses jarang, berdasarkan pola penggunaan Anda.

Langkah-langkah implementasi

- Tingkatkan observabilitas Anda dengan mengonfigurasi metrik beban kerja: Rekam metrik-metrik kunci untuk beban kerja. Metrik ini memberikan indikasi pengalaman pelanggan, seperti output beban kerja, dan menyelaraskan dengan perbedaan antara jenis dan ukuran sumber daya,

seperti CPU dan penggunaan memori. Untuk sumber daya komputasi, analisis data kinerja untuk mengukur EC2 instans Amazon Anda dengan benar. Identifikasi instans tidak aktif dan instans dengan pemanfaatan terlalu rendah. Metrik utama yang harus dicari adalah CPU penggunaan dan pemanfaatan memori (misalnya, 40% pemanfaatan pada 90% dari waktu seperti yang dijelaskan dalam Rightsizing with AWS Compute Optimizer and Memory CPU Utilization Enabled). Identifikasi contoh dengan CPU penggunaan maksimum dan pemanfaatan memori kurang dari 40% selama periode empat minggu. Ini adalah instans untuk disesuaikan ukurannya guna memangkas biaya. Untuk sumber daya penyimpanan seperti Amazon S3, Anda dapat menggunakan Lensa Penyimpanan Amazon S3, yang memungkinkan Anda melihat 28 metrik di berbagai kategori di tingkat bucket, dan 14 hari data historis di dasbor secara default. Anda dapat memfilter dasbor Lensa Penyimpanan Amazon S3 Anda berdasarkan ringkasan dan pengoptimalan biaya atau peristiwa untuk menganalisis metrik-metrik tertentu.

- Lihat rekomendasi rightsizing: Gunakan rekomendasi rightsizing di dan AWS Compute Optimizer alat penentuan EC2 ukuran Amazon di konsol Manajemen Biaya, atau tinjau sumber daya Anda dengan ukuran yang tepat untuk melakukan penyesuaian pada beban kerja Anda AWS Trusted Advisor . Penting untuk menggunakan alat yang tepat saat mengukur sumber daya yang berbeda dengan benar dan mengikuti pedoman ukuran kanan apakah itu EC2 instans Amazon, kelas AWS penyimpanan, atau jenis instans Amazon. RDS Untuk sumber daya penyimpanan, Anda dapat menggunakan Lensa Penyimpanan Amazon S3, yang memberi Anda visibilitas tentang penggunaan penyimpanan objek dan tren aktivitas, serta memberikan rekomendasi yang dapat ditindaklanjuti untuk mengoptimalkan biaya dan menerapkan praktik terbaik perlindungan data. Dengan menggunakan rekomendasi kontekstual yang diperoleh Lensa Penyimpanan Amazon S3 dari analisis metrik di seluruh organisasi Anda, Anda dapat segera mengambil langkah untuk mengoptimalkan penyimpanan.
- Pilih jenis dan ukuran sumber daya secara otomatis berdasarkan metrik: Dengan menggunakan metrik beban kerja, pilih sumber daya beban kerja Anda secara manual atau otomatis. Untuk sumber daya komputasi, konfigurasi AWS Auto Scaling atau implementasi kode di dalam aplikasi Anda dapat menghemat energi yang diperlukan jika diperlukan perubahan yang sering, dan ini dapat berpotensi mengimplementasikan perubahan lebih awal daripada proses manual. Anda dapat meluncurkan dan secara otomatis menyesuaikan armada Instans Sesuai Permintaan dan Instans Spot dalam satu grup Auto Scaling. Selain menerima diskon untuk menggunakan Spot Instance, Anda dapat menggunakan Instans Pesanan atau Savings Plan untuk menerima tarif diskon dengan harga biasa Instans Sesuai Permintaan. Semua faktor ini digabungkan membantu Anda mengoptimalkan penghematan biaya untuk EC2 instans Amazon dan menentukan skala dan kinerja yang diinginkan untuk aplikasi Anda. Anda juga dapat menggunakan strategi pemilihan tipe instans berbasis atribut (ABS) di Auto Scaling Groups ASG (), yang memungkinkan Anda

mengekspresikan persyaratan instance sebagai sekumpulan atribut, seperti CPU v, memori, dan penyimpanan. Anda dapat secara otomatis menggunakan jenis instans generasi yang lebih baru saat dirilis dan mengakses jangkauan kapasitas yang lebih luas dengan Instans EC2 Spot Amazon. Amazon EC2 Fleet dan Amazon EC2 Auto Scaling memilih dan meluncurkan instance yang sesuai dengan atribut yang ditentukan, sehingga tidak perlu memilih jenis instance secara manual. Untuk sumber daya penyimpanan, Anda dapat menggunakan fitur Amazon S3 Intelligent Tiering EFS dan Amazon Infrequent Access, yang memungkinkan Anda memilih kelas penyimpanan secara otomatis yang memberikan penghematan biaya penyimpanan otomatis saat pola akses data berubah, tanpa dampak kinerja atau overhead operasional.

Sumber daya

Dokumen terkait:

- AWS Auto Scaling
- AWS Ukuran Kanan
- AWS Compute Optimizer
- CloudWatch Fitur Amazon
- CloudWatchMendapatkan Pengaturan
- CloudWatchMenerbitkan Metrik Kustom
- Memulai dengan Amazon EC2 Auto Scaling
- Lensa Penyimpanan Amazon S3
- Amazon S3 Intelligent-Tiering
- Akses EFS Jarang Amazon
- Luncurkan EC2 Instans Amazon Menggunakan SDK

Video terkait:

- Sesuaikan Layanan Anda ke Ukuran yang Tepat

Contoh terkait:

- Pemilihan Jenis Instance berbasis atribut untuk Auto Scaling untuk Amazon Fleet EC2
- Mengoptimalkan biaya Layanan Kontainer Elastis Amazon dengan menggunakan penskalaan terjadwal

- Penskalaan prediktif dengan Amazon EC2 Auto Scaling
- Mengoptimalkan Biaya dan Mendapatkan Visibilitas ke dalam Penggunaan dengan Lensa Penyimpanan Amazon S3
- Lab Well-Architected: Rekomendasi Penyesuaian Ukuran yang Tepat (Level 100)

COST06-BP04 Pertimbangkan untuk menggunakan sumber daya bersama

Untuk layanan yang sudah digunakan di tingkat organisasi untuk beberapa unit bisnis, pertimbangkan untuk menggunakan sumber daya bersama untuk meningkatkan pemanfaatan dan mengurangi total biaya kepemilikan (). TCO Penggunaan sumber daya bersama dapat menjadi pilihan yang hemat biaya untuk memusatkan manajemen dan biaya dengan menggunakan solusi yang sudah ada, berbagi komponen, atau keduanya. Kelola fungsi-fungsi umum seperti pemantauan, pencadangan, dan konektivitas, baik dalam batasan akun maupun di sebuah akun khusus. Anda juga dapat mengurangi biaya dengan menerapkan standardisasi, mengurangi duplikasi, dan mengurangi kompleksitas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Apabila beberapa beban kerja menyebabkan fungsi yang sama, gunakan solusi yang sudah ada serta komponen bersama untuk memperbaiki manajemen dan mengoptimalkan biaya. Pertimbangkan penggunaan sumber daya yang sudah ada (terutama sumber daya bersama), seperti server basis data nonproduksi atau layanan direktori, untuk mengurangi biaya cloud dengan mengikuti praktik terbaik keamanan dan peraturan organisasi. Untuk realisasi dan efisiensi nilai yang optimal, sangat penting untuk mengalokasikan biaya kembali (menggunakan showback dan chargeback) ke area terkait bisnis yang mendorong konsumsi.

Showback mengacu pada laporan yang memecah biaya cloud ke dalam kategori yang dapat diatribusikan, seperti konsumen, unit bisnis, akun buku besar, atau entitas yang bertanggung jawab lainnya. Tujuan showback adalah untuk menunjukkan kepada tim, unit bisnis, atau individu, biaya sumber daya cloud yang mereka gunakan.

Chargeback berarti mengalokasikan pengeluaran layanan pusat ke unit biaya berdasarkan strategi yang sesuai untuk proses manajemen keuangan tertentu. Untuk pelanggan, chargeback membebankan biaya yang dikeluarkan dari satu akun layanan bersama ke kategori biaya keuangan berbeda yang sesuai untuk suatu proses pelaporan pelanggan. Dengan menetapkan mekanisme chargeback, Anda dapat melaporkan biaya yang dikeluarkan oleh unit bisnis, produk, dan tim yang berbeda.

Beban kerja dapat dikategorikan sebagai kritis dan nonkritis. Berdasarkan klasifikasi ini, gunakan sumber daya bersama dengan konfigurasi umum untuk beban kerja yang kurang kritis. Agar lebih mengoptimalkan biaya, cadangkan server khusus semata-mata untuk beban kerja kritis. Bagikan sumber daya atau sediakan di beberapa akun agar dapat dikelola secara efisien. Bahkan dengan lingkungan pengembangan, pengujian, dan produksi yang berbeda, berbagi dapat dilakukan secara aman dan tidak membahayakan struktur organisasi.

Untuk meningkatkan pemahaman Anda dan mengoptimalkan biaya serta penggunaan untuk aplikasi dalam kontainer, gunakan data alokasi biaya terpisah yang membantu Anda mengalokasikan biaya ke entitas bisnis individual berdasarkan cara aplikasi memakai sumber daya komputasi dan memori bersama. Data alokasi biaya terpisah membantu Anda mencapai showback dan chargeback tingkat tugas dalam beban kerja kontainer yang berjalan di Amazon Elastic Container Service (Amazon) atau Amazon Elastic ECS Kubernetes Service (Amazon). EKS

Untuk arsitektur terdistribusi, bangun layanan bersamaVPC, yang menyediakan akses terpusat ke layanan bersama yang diperlukan oleh beban kerja di masing-masing. VPCs Layanan bersama ini dapat mencakup sumber daya seperti layanan direktori atau VPC titik akhir. Untuk mengurangi biaya dan biaya administrasi, bagikan sumber daya dari lokasi pusat alih-alih membangunnya di masing-masing VPC lokasi.

Saat menggunakan sumber daya bersama, Anda dapat menghemat biaya operasional, memaksimalkan pemanfaatan sumber daya, dan meningkatkan konsistensi. Dalam desain multi-akun, Anda dapat meng-host beberapa AWS layanan secara terpusat dan mengaksesnya menggunakan beberapa aplikasi dan akun di hub untuk menghemat biaya. Anda dapat menggunakan AWS Resource Access Manager (AWS RAM) untuk berbagi sumber daya umum lainnya, seperti VPCsubnet dan AWS Transit Gateway lampiran AWS Network Firewall, atau saluran pipa Amazon SageMaker . Dalam lingkungan multi-akun, gunakan AWS RAM untuk membuat sumber daya sekali dan membagikannya dengan akun lain.

Organisasi harus memberikan tag pada biaya bersama secara efektif dan memverifikasi bahwa hanya sedikit biaya mereka yang tidak diberi tag atau dialokasikan. Jika Anda tidak mengalokasikan biaya bersama secara efektif dan tidak ada orang yang bertanggung jawab atas pengelolaan biaya bersama, biaya cloud bersama dapat meningkat. Anda harus tahu di mana Anda telah mengeluarkan biaya pada tingkat sumber daya, beban kerja, tim, atau organisasi, karena pengetahuan ini meningkatkan pemahaman Anda tentang nilai yang dihadirkan pada tingkat yang berlaku saat dibandingkan dengan hasil bisnis yang dicapai. Pada akhirnya, organisasi mendapat manfaat berupa penghematan biaya sebagai hasil dari berbagi infrastruktur cloud. Dorong alokasi biaya pada sumber daya cloud bersama untuk mengoptimalkan pengeluaran cloud.

Langkah-langkah implementasi

- Evaluasi sumber daya yang ada: Tinjau beban kerja yang ada yang menggunakan layanan serupa untuk beban kerja Anda. Tergantung komponen beban kerja, pertimbangkan platform yang ada jika logika bisnis atau persyaratan teknis memungkinkan.
- Gunakan berbagi sumber daya AWS RAM dan batasi yang sesuai: Gunakan AWS RAM untuk berbagi sumber daya dengan AWS akun lain dalam organisasi Anda. Saat berbagi sumber daya, Anda tidak perlu menduplikasi sumber daya di beberapa akun, sehingga meminimalkan beban operasional pemeliharaan sumber daya. Proses ini juga membantu Anda secara aman membagikan sumber daya yang Anda buat kepada peran dan pengguna di akun Anda, serta kepada Akun AWS lainnya.
- Sumber daya tag: Berikan tag pada sumber daya yang merupakan kandidat untuk pelaporan biaya dan kategorikan sumber daya tersebut ke dalam kategori biaya. Aktifkan tag sumber daya terkait biaya ini untuk alokasi biaya guna memberikan visibilitas penggunaan AWS sumber daya. Fokus pada menciptakan tingkat granularitas yang sesuai sehubungan dengan biaya dan visibilitas penggunaan, dan mempengaruhi perilaku konsumsi cloud melalui pelaporan dan pelacakan alokasi biaya. KPI

Sumber daya

Praktik-praktik terbaik terkait:

- SEC03-BP08 Bagikan sumber daya dengan aman di dalam organisasi Anda

Dokumen terkait:

- Apa itu AWS Resource Access Manager?
- AWS Layanan yang dapat Anda gunakan dengan AWS Organizations
- Sumber daya yang dapat dibagikan AWS
- AWS Biaya dan Penggunaan (CUR) Pertanyaan

Video terkait:

- AWS Resource Access Manager - kontrol akses granular dengan izin terkelola
- Bagaimana merancang strategi alokasi AWS biaya Anda
- AWS Cost Categories

Contoh terkait:

- Cara tolak bayar kembali layanan bersama: Contoh AWS Transit Gateway
- Cara membuat model chargeback/showback untuk Savings Plans menggunakan CUR
- Menggunakan VPC Berbagi untuk Arsitektur Microservice Multi-Akun yang Hemat Biaya
- Tingkatkan visibilitas biaya Amazon EKS dengan Data AWS Alokasi Biaya Terpisah
- Meningkatkan visibilitas biaya Amazon ECS dan AWS Batch dengan AWS Split Cost Allocation Data

## COST7. Bagaimana cara menggunakan model harga untuk mengurangi biaya?

Gunakan model harga yang paling sesuai untuk sumber daya Anda untuk meminimalkan pengeluaran.

Praktik terbaik

- COST07-BP01 Lakukan analisis model harga
- COST07-BP02 Pilih Wilayah berdasarkan biaya
- COST07-BP03 Pilih perjanjian pihak ketiga dengan persyaratan hemat biaya
- COST07-BP04 Menerapkan model penetapan harga untuk semua komponen beban kerja ini
- COST07-BP05 Lakukan analisis model penetapan harga di tingkat akun manajemen

### COST07-BP01 Lakukan analisis model harga

Analisis setiap komponen beban kerja. Tentukan apakah komponen dan sumber daya akan dijalankan dalam waktu yang lama (untuk diskon komitmen), atau bersifat dinamis dan dijalankan dalam waktu singkat (untuk spot atau sesuai permintaan). Lakukan analisis pada beban kerja menggunakan rekomendasi dalam alat manajemen biaya dan terapkan aturan bisnis pada rekomendasi tersebut untuk mendapatkan hasil yang tinggi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

AWS memiliki beberapa model harga yang memungkinkan Anda membayar sumber daya Anda dengan cara yang paling hemat biaya yang sesuai dengan kebutuhan organisasi Anda dan tergantung pada produk. Bekerjalah dengan tim Anda untuk menentukan model harga yang

paling sesuai. Sering kali model harga terdiri dari kombinasi beberapa opsi, yang ditentukan oleh ketersediaan Anda

Instans Sesuai Permintaan memungkinkan Anda membayar untuk kapasitas komputasi atau basis data sesuai jam atau detik (minimum 60 detik) tergantung pada instans mana yang Anda jalankan, tanpa perlu ada komitmen jangka panjang atau pembayaran di muka.

Savings Plans adalah model penetapan harga fleksibel yang menawarkan harga rendah di AmazonEC2, Lambda, dan AWS Fargate penggunaan, dengan imbalan komitmen terhadap jumlah penggunaan yang konsisten (diukur dalam dolar per jam) selama satu tahun atau tiga tahun.

Instans Spot adalah mekanisme EC2 penetapan harga Amazon yang memungkinkan Anda meminta kapasitas komputasi cadangan dengan tarif per jam diskon (hingga 90% dari harga sesuai permintaan) tanpa komitmen di muka.

Instans Terpesan akan memungkinkan Anda mendapat diskon hingga 75 persen dengan membayar di muka atas kapasitas yang didapat. Untuk detail selengkapnya, lihat Mengoptimalkan biaya dengan reservasi.

Anda mungkin memilih untuk menyertakan Savings Plans untuk sumber daya yang terkait dengan lingkungan produksi, kualitas, dan pengembangan. Alternatifnya, karena sumber daya sandbox hanya diaktifkan saat diperlukan, Anda mungkin memilih model sesuai permintaan untuk sumber daya dalam lingkungan tersebut. Gunakan Instans Spot Amazon untuk mengurangi EC2 biaya Amazon atau menggunakan Compute Savings Plans untuk mengurangi biaya AmazonEC2, Fargate, dan Lambda. Alat rekomendasi AWS Cost Explorer memberikan peluang untuk diskon komitmen dengan paket Savings Plans.

Jika Anda telah membeli Instans Cadangan untuk Amazon EC2 di masa lalu atau telah menetapkan praktik alokasi biaya di dalam organisasi Anda, Anda dapat terus menggunakan Instans EC2 Cadangan Amazon untuk saat ini. Tetapi, kami menyarankan agar Anda membuat strategi untuk menggunakan Savings Plans di waktu mendatang sebagai mekanisme penghematan biaya yang lebih fleksibel. Anda dapat menyegarkan Rekomendasi Savings Plans (SP) AWS Cost Management untuk menghasilkan Rekomendasi Savings Plans baru kapan saja. Gunakan Instans Cadangan (RI) untuk mengurangi biaya AmazonRDS, Amazon Redshift, Amazon, dan ElastiCache OpenSearch Amazon Service. Savings Plans dan Instans Terpesan tersedia dalam tiga opsi, yakni: semua pembayaran di muka, sebagian pembayaran di muka, dan tanpa pembayaran di muka. Gunakan rekomendasi yang diberikan dalam rekomendasi pembelian AWS Cost Explorer RI dan SP.

Untuk menemukan peluang untuk beban kerja Spot, gunakan tampilan penggunaan secara keseluruhan per jam, dan cari periode penggunaan atau elastisitas yang berubah secara teratur.

Anda dapat menggunakan Spot Instance untuk berbagai aplikasi yang toleran terhadap kesalahan dan fleksibel. Contohnya termasuk server web stateless, API endpoint, data besar dan aplikasi analitik, beban kerja kontainer, CI/CD, dan beban kerja fleksibel lainnya.

Analisis RDS instans Amazon EC2 dan Amazon Anda apakah mereka dapat dimatikan saat Anda tidak menggunakannya (setelah jam kerja dan akhir pekan). Pendekatan ini akan memungkinkan Anda mengurangi biaya hingga 70% atau lebih dibandingkan penggunaan 24/7. Jika Anda memiliki klaster Amazon Redshift yang hanya perlu disediakan pada waktu tertentu, Anda dapat menghentikan sejenak klaster dan melanjutkannya nanti. Saat klaster Amazon Redshift atau Amazon dan RDS Instans EC2 Amazon dihentikan, tagihan komputasi terhenti dan hanya biaya penyimpanan yang berlaku.

Perhatikan bahwa reservasi Kapasitas Sesuai Permintaan (ODCR) bukanlah diskon harga. Reservasi Kapasitas ditagih dengan tarif Sesuai Permintaan yang setara, baik Anda menjalankan instans di kapasitas terpesan atau tidak. Reservasi ini harus dipertimbangkan saat Anda harus memberikan cukup banyak kapasitas untuk sumber daya yang menurut rencana akan Anda jalankan. ODCRstidak harus terikat dengan komitmen jangka panjang, karena dapat dibatalkan saat Anda tidak lagi membutuhkannya, tetapi mereka juga bisa mendapatkan keuntungan dari diskon yang diberikan oleh Savings Plans atau Reserved Instances.

Langkah-langkah implementasi

- Analisis elastisitas beban kerja: Dengan menggunakan perincian per jam di Cost Explorer atau dasbor kustom, lakukan analisis terhadap elastisitas beban kerja Anda. Cari perubahan teratur dalam jumlah instans yang dijalankan. Instans berdurasi pendek merupakan kandidat untuk Instans Spot atau Armada Spot.
  - Lab Well-Architected: Cost Explorer
  - Lab Well-Architected: Visualisasi Biaya
- Tinjau kontrak harga yang ada saat ini: Tinjau kontrak atau komitmen saat ini untuk kebutuhan jangka panjang. Analisis apa yang Anda miliki saat ini dan berapa banyak komitmen tersebut digunakan. Manfaatkan diskon kontrak atau perjanjian perusahaan yang sudah ada sebelumnya. Perjanjian Perusahaan memberi pelanggan pilihan untuk menyesuaikan perjanjian yang paling sesuai dengan kebutuhan mereka. Untuk komitmen jangka panjang, pertimbangkan diskon harga cadangan, Instans Cadangan, atau Savings Plans untuk jenis instans tertentu, keluarga instans Wilayah AWS, dan Availability Zone.
- Lakukan analisis discount komitmen: Dengan menggunakan Cost Explorer di akun Anda, tinjau rekomendasi Savings Plans dan Instans Terpesan. Untuk memverifikasi bahwa Anda

mengimplementasikan rekomendasi yang benar dengan risiko dan diskon yang diperlukan, ikuti lab Well-Architected.

Sumber daya

Dokumen terkait:

- Mengakses Rekomendasi Instans Terpesan
- Opsi pembelian instans
- AWS Perusahaan

Video terkait:

- Hemat hingga 90% dan jalankan beban kerja produksi di Instans Spot

Contoh terkait:

- Lab Well-Architected: Cost Explorer
- Lab Well-Architected: Visualisasi Biaya
- Lab Well-Architected: Model Harga

COST07-BP02 Pilih Wilayah berdasarkan biaya

Penetapan harga sumber daya dapat berbeda di setiap Wilayah. Identifikasi perbedaan biaya sesuai Wilayah dan lakukan deployment di Wilayah dengan biaya yang lebih tinggi hanya untuk memenuhi persyaratan latensi, residensi data, dan kedaulatan data. Dengan mempertimbangkan biaya Wilayah Anda dapat memperoleh harga keseluruhan yang paling rendah untuk beban kerja ini.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS Cloud Infrastruktur bersifat global, diselenggarakan di beberapa lokasi di seluruh dunia, dan dibangun di sekitar, Availability Zones Wilayah AWS, Local Zones, AWS Outposts, dan Wavelength Zones. Wilayah adalah lokasi fisik di dunia dan setiap Wilayah adalah wilayah geografis terpisah di mana AWS memiliki beberapa Availability Zone. Zona Ketersediaan yang merupakan beberapa lokasi terpisah di dalam setiap Wilayah terdiri dari satu atau beberapa pusat data khusus, masing-masing memiliki daya, jaringan, dan konektivitas redundan.

Masing-masing Wilayah AWS beroperasi dalam kondisi pasar lokal, dan harga sumber daya berbeda di setiap Wilayah karena perbedaan biaya tanah, serat, listrik, dan pajak, misalnya. Pilihlah sebuah Wilayah tertentu untuk mengoperasikan komponen atau seluruh solusi Anda sehingga Anda dapat menjalankannya dengan harga serendah mungkin secara global. Gunakan Kalkulator AWS untuk menghitung perkiraan biaya beban kerja Anda di berbagai Wilayah dengan mencari layanan berdasarkan tipe lokasi (Wilayah, zona wavelength, dan zona lokal) dan Wilayah.

Saat Anda merancang solusi, praktik terbaik yang dapat diterapkan adalah berusaha menempatkan sumber daya komputasi lebih dekat dengan pengguna untuk memberikan latensi yang lebih rendah dan kedaulatan data yang kuat. Pilih lokasi geografis berdasarkan persyaratan bisnis, privasi data, kinerja, dan keamanan Anda. Untuk aplikasi dengan pengguna akhir global, gunakan beberapa lokasi.

Gunakan Wilayah yang memberikan harga lebih rendah untuk AWS layanan untuk menyebarkan beban kerja Anda jika Anda tidak memiliki kewajiban dalam privasi data, keamanan, dan persyaratan bisnis. Misalnya, jika Wilayah default Anda adalah Asia Pasifik (Sydney) (`ap-southwest-2`), dan jika tidak ada batasan (privasi data, keamanan, misalnya) untuk menggunakan Wilayah lain, menerapkan EC2 instans Amazon yang tidak kritis (pengembangan dan pengujian) di US East (Virginia N.) (`us-east-1`) akan dikenakan biaya lebih murah.

| | *Kepatuhan* | *Latensi* | *Biaya* | *Layanan/Fitur* |
|---|---|---|---|---|
| *Wilayah 1* | ✓ | 15 milidetik | $$ | ✓ |
| *Wilayah 2* | ✓ | 20 milidetik | $$$ | X |
| *Wilayah 3* | ✓ | 80 milidetik | $ | ✓ |
| *Wilayah 4* | ✓ | 15 milidetik | $$ | ✓ |
| *Wilayah 5* | ✓ | 20 milidetik | $$$ | X |
| *Wilayah 6* | ✓ | 15 milidetik | $ | ✓ |
| *Wilayah 7* | ✓ | 80 milidetik | $ | ✓ |
| *Wilayah 8* | ✓ | 15 milidetik | $ | X |

Tabel matriks fitur Wilayah

Tabel matriks di atas menunjukkan kepada kita bahwa Wilayah 6 adalah opsi terbaik untuk skenario yang diberikan ini karena latensinya rendah dibandingkan dengan Wilayah lain, layanannya tersedia, dan ini adalah Wilayah yang paling murah.

Langkah-langkah implementasi

- Tinjau Wilayah AWS harga: Analisis biaya beban kerja di Wilayah saat ini. Mulai dengan harga tertinggi dari jenis layanan dan penggunaan, lalu hitung biayanya di Wilayah yang tersedia. Jika perkiraan penghematan lebih banyak dari biaya pemindahan komponen atau beban kerja, lakukan migrasi ke Wilayah baru.
- Tinjau persyaratan untuk deployment multi-Wilayah: Analisis persyaratan dan kewajiban bisnis Anda (privasi data, keamanan, atau kinerja) untuk mencari tahu apakah ada pembatasan bagi Anda untuk menggunakan beberapa Wilayah. Gunakan beberapa Wilayah jika tidak ada kewajiban yang membatasi Anda untuk menggunakan Wilayah tunggal.
- Analisis transfer data yang diperlukan: Pertimbangkan biaya transfer data saat memilih Wilayah. Simpan data Anda dekat dengan sumber dayanya dan dengan pelanggan Anda. Pilih yang lebih murah Wilayah AWS di mana data mengalir dan di mana ada transfer data minimal. Bergantung pada kebutuhan bisnis Anda untuk transfer data, Anda dapat menggunakan Amazon CloudFront AWS PrivateLink, AWS Direct Connect,, dan AWS Virtual Private Networkuntuk mengurangi biaya jaringan, meningkatkan kinerja, dan meningkatkan keamanan.

Sumber daya

Dokumen terkait:

- Mengakses Rekomendasi Instans Terpesan
- EC2Harga Amazon
- Opsi pembelian instans
- Tabel Wilayah

Video terkait:

- Hemat hingga 90% dan jalankan beban kerja produksi di Instans Spot

Contoh terkait:

- Ikhtisar Biaya Transfer Data untuk Arsitektur Umum

- Pertimbangan Biaya untuk Deployment Global
- Hal-Hal yang Perlu Dipertimbangkan saat Memilih Wilayah untuk Beban Kerja Anda
- Lab Well-Architected: Membatasi penggunaan layanan berdasarkan Wilayah (Level 200)

COST07-BP03 Pilih perjanjian pihak ketiga dengan persyaratan hemat biaya

Perjanjian dan ketentuan yang hemat biaya memastikan biaya layanan ini dapat disesuaikan dengan manfaat yang disediakan. Pilih perjanjian dan harga yang dapat diskalakan ketika ada keuntungan tambahan yang disediakan untuk organisasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Ada beberapa produk di pasar yang dapat membantu Anda mengelola biaya di lingkungan cloud Anda. Produk-produk tersebut mungkin memiliki fitur yang berbeda-beda sesuai kebutuhan pelanggan; misalnya, beberapa di antaranya berfokus pada tata kelola biaya atau visibilitas biaya, sedangkan produk lainnya berfokus pada pengoptimalan biaya. Salah satu faktor penting dalam pengoptimalan dan tata kelola biaya yang efektif adalah penggunaan alat yang tepat dengan fitur yang sesuai kebutuhan dan model harga yang tepat. Model harga untuk produk-produk ini berbeda. Beberapa produk mengenakan persentase tertentu dari tagihan bulanan Anda, sementara produk lainnya mengenakan persentase dari realisasi penghematan Anda. Idealnya, Anda seharusnya hanya membayar sesuai apa yang Anda butuhkan.

Saat Anda menggunakan solusi atau layanan pihak ketiga di cloud, penting untuk memastikan bahwa struktur harganya selaras dengan hasil yang Anda inginkan. Skema harga harus diskalakan sesuai dengan hasil dan nilai yang disediakan oleh solusi atau layanan tersebut. Contohnya, dalam perangkat lunak yang mengambil persentase dari penghematan yang dihasilkannya, makin banyak Anda menghemat (hasil), makin tinggi juga biayanya. Perjanjian lisensi yang mengharuskan Anda membayar lebih banyak ketika pengeluaran Anda meningkat mungkin bukan cara terbaik bagi Anda untuk mengoptimalkan biaya. Namun, jika vendor tersebut menawarkan manfaat yang jelas untuk semua elemen dalam tagihan Anda, biaya yang meningkat ini mungkin dapat dijustifikasi.

Misalnya, solusi yang memberikan rekomendasi untuk Amazon EC2 dan membebankan persentase dari seluruh tagihan Anda dapat menjadi lebih mahal jika Anda menggunakan layanan lain yang tidak memberikan manfaat. Contoh lainnya adalah layanan terkelola yang dikenakan biaya sekian persen dari biaya sumber daya yang dikelola. Ukuran instans yang lebih besar tidak selalu memerlukan lebih banyak upaya manajemen, tetapi dapat dikenakan biaya lebih banyak. Pastikan bahwa perjanjian

harga layanan ini mencakup fitur atau program pengoptimalan biaya di layanannya untuk mendorong efisiensi.

Pelanggan mungkin merasa produk-produk yang ada di pasar ini lebih canggih atau mudah digunakan. Anda perlu mempertimbangkan biaya produk ini dan memikirkan potensi hasil pengoptimalan biaya dalam jangka panjang.

Langkah-langkah implementasi

- Menganalisis perjanjian dan persyaratan pihak ketiga: Tinjau harga dalam perjanjian pihak ketiga. Lakukan pemodelan untuk berbagai tingkat penggunaan Anda, dan pertimbangkan biaya baru seperti penggunaan layanan baru, atau peningkatan pada layanan saat ini akibat pertumbuhan beban kerja. Tentukan apakah biaya tambahan memberikan keuntungan yang diperlukan pada bisnis Anda.

Sumber daya

Dokumen terkait:

- Mengakses Rekomendasi Instans Terpesan
- Opsi pembelian instans

Video terkait:

- Hemat hingga 90% dan jalankan beban kerja produksi di Instans Spot

COST07-BP04 Menerapkan model penetapan harga untuk semua komponen beban kerja ini

Sumber daya yang berjalan secara permanen harus menggunakan kapasitas terpesan seperti Savings Plans atau Instans Terpesan. Kapasitas jangka pendek dikonfigurasi untuk menggunakan Instans Spot, atau Armada Spot. Instans Sesuai Permintaan hanya digunakan untuk beban kerja jangka pendek yang tidak bisa dihentikan dan tidak berjalan cukup lama untuk kapasitas terpesan, yakni antara 25% sampai 75% dari periode, tergantung pada tipe sumber daya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

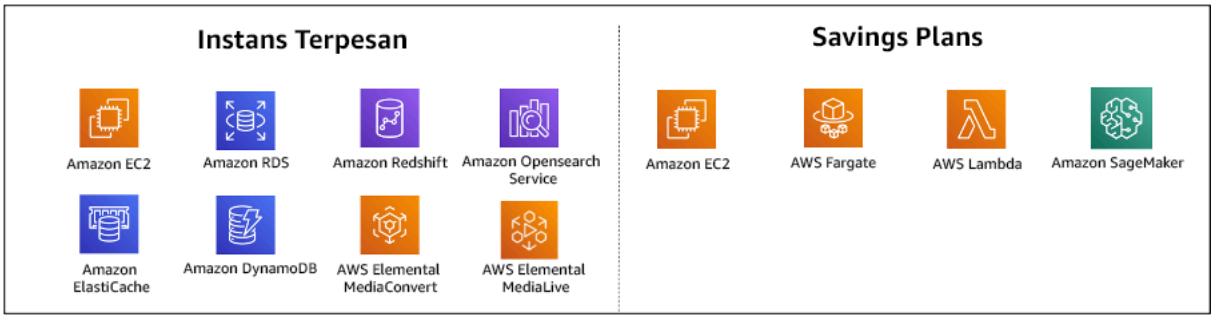
Panduan implementasi

Untuk meningkatkan efisiensi biaya, AWS berikan beberapa rekomendasi komitmen berdasarkan penggunaan Anda sebelumnya. Anda dapat menggunakan rekomendasi ini untuk memahami apa

yang dapat Anda hemat, dan bagaimana penggunaan komitmen tersebut. Anda dapat menggunakan layanan ini sebagai On-Demand, Spot, atau membuat komitmen untuk jangka waktu tertentu dan mengurangi biaya sesuai permintaan Anda dengan Instans Cadangan () dan Savings Plans (RIs). SPs Anda perlu memahami tidak hanya setiap komponen beban kerja dan beberapa AWS layanan, tetapi juga diskon komitmen, opsi pembelian, dan Instans Spot untuk layanan ini guna mengoptimalkan beban kerja Anda.

Pertimbangkan persyaratan komponen beban kerja Anda, dan pahami model harga yang berbeda untuk layanan ini. Tentukan persyaratan ketersediaan komponen ini. Tentukan apakah ada beberapa sumber daya independen yang melakukan fungsinya pada beban kerja, serta apa persyaratan beban kerja dari waktu ke waktu. Bandingkan biaya sumber daya menggunakan model harga Sesuai Permintaan default dan model lain yang dapat diterapkan. Pertimbangkan potensi perubahan apa pun pada sumber daya atau komponen beban kerja.

Sebagai contoh, mari kita lihat Arsitektur Aplikasi Web ini di AWS. Contoh beban kerja ini terdiri dari beberapa AWS layanan, seperti Amazon Route 53, Amazon AWS WAF, instans Amazon CloudFront, EC2 instans Amazon, Load RDS Balancer, penyimpanan Amazon S3, dan Amazon Elastic File System (Amazon). EFS Anda perlu meninjau setiap layanan ini, dan mengidentifikasi peluang penghematan biaya potensial dengan model harga yang berbeda. Beberapa dari mereka mungkin memenuhi syarat untuk RIs atauSPs, sementara beberapa dari mereka mungkin hanya tersedia berdasarkan permintaan. Seperti yang ditunjukkan gambar berikut, beberapa AWS layanan dapat dilakukan menggunakan RIs atauSPs.




AWS layanan yang dilakukan menggunakan Instans Cadangan dan Savings Plans

Langkah-langkah implementasi

- Menerapkan model harga: Dengan menggunakan hasil analisis Anda, lakukan pembelian paket Savings Plans, Instans Terpesan, atau terapkan Instans Spot. Jika ini adalah pembelian komitmen pertama Anda, pilih lima atau sepuluh rekomendasi teratas dalam daftar, lalu pantau dan analisis hasilnya selama satu atau dua bulan ke depan. AWS Cost Management Console memandu Anda melalui proses. Tinjau rekomendasi RI atau SP dari konsol, sesuaikan rekomendasi (jenis,

pembayaran, serta jangka waktu), dan tinjau komitmen per jam (misalnya 20 USD per jam), lalu tambahkan ke keranjang. Diskon berlaku secara otomatis untuk penggunaan yang memenuhi syarat. Beli sejumlah kecil diskon komitmen dalam siklus reguler (misalnya setiap 2 pekan atau setiap bulan). Implementasikan Instans Spot untuk beban kerja yang bisa dihentikan atau stateless. Terakhir, pilih EC2 instans Amazon sesuai permintaan dan alokasikan sumber daya untuk persyaratan yang tersisa.

- Siklus peninjauan beban kerja: Menerapkan siklus peninjauan untuk beban kerja yang secara khusus menganalisis cakupan model penetapan harga. Setelah beban kerja memiliki cakupan yang diperlukan, beli diskon komitmen tambahan secara parsial (setiap beberapa bulan), atau sesuai dengan perubahan penggunaan organisasi Anda.

Sumber daya

Dokumen terkait:

- Memahami rekomendasi Savings Plans Anda
- Mengakses Rekomendasi Instans Terpesan
- Cara Membeli Instans Terpesan
- Opsi pembelian instans
- Instans Spot
- Model reservasi untuk AWS layanan lain
- Layanan yang Didukung Savings Plans

Video terkait:

- Hemat hingga 90% dan jalankan beban kerja produksi di Instans Spot

Contoh terkait:

- Apa yang harus Anda pertimbangkan sebelum membeli Savings Plans?
- Bagaimana cara menggunakan Cost Explorer untuk menganalisis pengeluaran dan penggunaan saya?

COST07-BP05 Lakukan analisis model penetapan harga di tingkat akun manajemen

Periksa alat manajemen biaya dan tagihan dan lihat diskon yang direkomendasikan dengan komitmen dan reservasi untuk melakukan analisis secara teratur di tingkat akun manajemen.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Melakukan pemodelan biaya secara teratur membantu Anda mengimplementasikan peluang untuk mengoptimalkan di beberapa beban kerja. Misalnya, jika beberapa beban kerja menggunakan Instans Sesuai Permintaan, di tingkat agregat, risiko perubahan akan lebih rendah, dan implementasi diskon berbasis komitmen dapat menghasilkan biaya keseluruhan yang lebih rendah. Disarankan agar analisis dilakukan dalam siklus waktu dua minggu hingga satu bulan secara teratur. Ini memungkinkan Anda untuk melakukan pembelian penyesuaian kecil, sehingga cakupan model harga Anda terus berevolusi dengan beban kerja serta komponennya yang terus berubah.

Gunakan alat-alat rekomendasi AWS Cost Explorer untuk menemukan peluang-peluang mendapat diskon komitmen dalam akun manajemen Anda. Rekomendasi di tingkat akun manajemen dihitung dengan mempertimbangkan penggunaan di semua akun di organisasi AWS Anda yang memiliki diskon Instans Terpesan (RI) atau Savings Plans (SP). Rekomendasi tersebut juga dihitung ketika berbagi diskon diaktifkan untuk merekomendasikan komitmen yang memaksimalkan penghematan di seluruh akun.

Meskipun pembelian di tingkat akun manajemen mengoptimalkan penghematan maksimal dalam banyak kasus, mungkin ada situasi di mana Anda mungkin mempertimbangkan untuk membeli SPs di tingkat akun tertaut, seperti ketika Anda ingin diskon diterapkan terlebih dahulu untuk penggunaan di akun tertaut tertentu. Rekomendasi akun anggota dihitung di tingkat akun individu, untuk memaksimalkan penghematan untuk setiap akun terisolasi. Jika akun Anda memiliki komitmen RI dan SP, urutan penerapannya adalah:

1. RI Zona
2. RI Standar
3. RI Konvertibel
4. Savings Plans Instans
5. Compute Savings Plan

Jika Anda membeli SP di tingkat akun manajemen, penghematan akan diterapkan berdasarkan persentase diskon tertinggi hingga terendah. SPsdi tingkat akun manajemen, lihat semua akun yang ditautkan dan terapkan tabungan di mana pun diskon akan menjadi yang tertinggi. Jika Anda ingin membatasi tempat-tempat penghematan diterapkan, Anda dapat membeli Savings Plan di tingkat akun tertaut dan setiap kali akun tersebut menjalankan layanan komputasi yang memenuhi syarat, diskon akan diterapkan di sana terlebih dahulu. Ketika akun tersebut tidak menjalankan layanan komputasi yang memenuhi syarat, diskon akan dibagikan ke akun tertaut lainnya di bawah akun manajemen yang sama. Berbagi diskon diaktifkan secara default, tetapi dapat dinonaktifkan jika diperlukan.

Dalam Keluarga Penagihan Terkonsolidasi, Savings Plans diterapkan terlebih dahulu ke penggunaan akun pemilik, kemudian ke penggunaan akun lain. Ini hanya terjadi jika Anda mengaktifkan fitur berbagi. Savings Plans Anda diterapkan ke persentase penghematan tertinggi Anda terlebih dahulu. Jika ada beberapa penggunaan dengan persentase penghematan yang sama, Savings Plans diterapkan ke penggunaan pertama dengan tingkat Savings Plans terendah. Savings Plans terus berlaku sampai tidak ada lagi penggunaan yang tersisa atau komitmen Anda habis. Sisa penggunaan lainnya akan ditagih dengan tarif Sesuai Permintaan. Anda dapat menyegarkan Rekomendasi Savings Plans dalam Manajemen AWS Biaya untuk menghasilkan Rekomendasi Savings Plans baru kapan saja.

Setelah menganalisis fleksibilitas instans, Anda dapat memberikan komitmen dengan mengikuti rekomendasi. Buat pemodelan biaya dengan menganalisis biaya jangka pendek beban kerja dengan opsi sumber daya potensial yang berbeda, menganalisis model AWS harga, dan menyelaraskannya dengan persyaratan bisnis Anda untuk mengetahui total biaya kepemilikan dan peluang pengoptimalan biaya.

Langkah-langkah implementasi

Lakukan analisis discount komitmen: Gunakan Cost Explorer di akun Anda, tinjau rekomendasi Savings Plans dan Instans Terpesan. Pastikan Anda memahami rekomendasi Savings Plans, dan perkirakan estimasi pengeluaran bulanan serta penghematan bulanan Anda. Tinjau rekomendasi di tingkat akun manajemen yang dihitung dengan mempertimbangkan penggunaan di semua akun anggota di organisasi AWS Anda yang disertai pengaktifan pembagian diskon RI atau Savings Plans, untuk memaksimalkan penghematan di seluruh akun. Anda dapat memverifikasi bahwa Anda telah mengimplementasikan rekomendasi yang benar dengan risiko dan diskon yang diperlukan dengan mengikuti lab Well-Architected.

Sumber daya

Dokumen terkait:

- Bagaimana cara kerja AWS penetapan harga?
- Opsi pembelian instans
- Ikhtisar Savings Plan
- Rekomendasi Savings Plans
- Mengakses Rekomendasi Instans Terpesan
- Memahami rekomendasi Savings Plans Anda
- Bagaimana Savings Plans berlaku untuk AWS penggunaan Anda
- Savings Plans dengan Penagihan Terkonsolidasi
- Mengaktifkan diskon instans terpesan dan Savings Plans bersama

Video terkait:

- Hemat hingga 90% dan jalankan beban kerja produksi di Instans Spot

Contoh terkait:

- Lab Well-Architected AWS : Model Harga (Tingkat 200)
- Lab Well-Architected AWS : Analisis Model Harga (Tingkat 200)
- Apa yang harus saya pertimbangkan sebelum membeli Savings Plans?
- Bagaimana cara menggunakan paket Savings Plans bergulir untuk mengurangi risiko komitmen?
- Kapan Harus Menggunakan Instans Spot

## COST8. Bagaimana cara Anda merencanakan biaya transfer data?

Pastikan Anda merencanakan dan memantau biaya transfer data sehingga Anda dapat mengambil keputusan arsitektur untuk meminimalkan biaya. Perubahan arsitektur yang kecil, tetapi efektif dapat secara drastis mengurangi biaya operasional Anda seiring waktu.

Praktik terbaik

- COST08-BP01 Lakukan pemodelan transfer data

- COST08-BP02 Pilih komponen untuk mengoptimalkan biaya transfer data
- COST08-BP03 Menerapkan layanan untuk mengurangi biaya transfer data

COST08-BP01 Lakukan pemodelan transfer data

Kumpulkan kebutuhan organisasi dan lakukan pemodelan transfer data terhadap beban kerja dan setiap komponennya. Hal ini mengidentifikasi titik biaya terendah untuk persyaratan transfer data saat ini.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Saat merancang solusi di cloud, biaya transfer data biasanya diabaikan karena sudah terbiasa merancang arsitektur menggunakan pusat data on-premise atau ketidaktahuan. Biaya transfer data AWS ditentukan oleh sumber, tujuan, dan volume lalu lintas. Memperhitungkan biaya-biaya ini dalam fase desain dapat menghasilkan penghematan biaya. Memahami di mana transfer data terjadi dalam beban kerja Anda, biaya transfer, dan manfaat terkait sangat penting untuk secara akurat memperkirakan total biaya kepemilikan (TCO). Dengan demikian, Anda dapat membuat keputusan yang lebih tepat untuk mengubah atau menerima keputusan arsitektur. Misalnya, Anda memiliki konfigurasi Multi-Zona Ketersediaan yang datanya Anda replikasi antara Zona Ketersediaan tersebut.

Anda membuat model komponen layanan yang mentransfer data dalam beban kerja Anda, dan memutuskan bahwa ini adalah biaya yang dapat diterima (mirip dengan membayar komputasi dan penyimpanan di kedua Zona Ketersediaan) untuk mencapai keandalan dan ketahanan yang diperlukan. Buat model biaya untuk berbagai tingkat penggunaan. Penggunaan beban kerja dapat berubah dari waktu ke waktu, dan beberapa layanan bisa menjadi lebih hemat biaya pada level tertentu.

Saat membuat model transfer data Anda, pikirkan berapa banyak data yang diserap dan dari mana data tersebut berasal. Selain itu, pertimbangkan berapa banyak data yang diproses dan berapa banyak penyimpanan atau kapasitas komputasi yang dibutuhkan. Selama pemodelan, ikuti praktik terbaik jaringan untuk arsitektur beban kerja Anda guna mengoptimalkan potensi biaya transfer data Anda.

Ini AWS Pricing Calculator dapat membantu Anda melihat perkiraan biaya untuk AWS layanan tertentu dan transfer data yang diharapkan. Jika Anda memiliki beban kerja yang sudah berjalan (untuk tujuan pengujian atau dalam lingkungan pra-produksi), gunakan AWS Cost Exploreratau

AWS Cost and Usage Report(CUR) untuk memahami dan memodelkan biaya transfer data Anda. Konfigurasikan bukti konsep (PoC) atau uji beban kerja Anda, dan lakukan pengujian dengan simulasi beban realistis. Anda dapat membuat model biaya untuk berbagai permintaan beban kerja.

Langkah-langkah implementasi

- Identifikasi persyaratan: Apa tujuan utama dan persyaratan bisnis untuk transfer data terencana antara sumber dan tujuan? Apa hasil bisnis yang diharapkan di akhir nanti? Kumpulkan persyaratan bisnis dan tentukan hasil akhir yang diharapkan.
- Identifikasi sumber dan tujuan: Apa sumber data dan tujuan transfer data, seperti di dalam Wilayah AWS, ke AWS layanan, atau keluar ke internet?
  - Transfer data dalam Wilayah AWS
  - Transfer data antara Wilayah AWS
  - Transfer data ke internet
- Identifikasi klasifikasi data: Apa klasifikasi data untuk transfer data ini? Apa jenis data tersebut? Seberapa besar datanya? Seberapa sering data harus ditransfer? Apakah data sensitif?
- Identifikasi AWS layanan atau alat yang akan digunakan: AWS Layanan apa yang digunakan untuk transfer data ini? Apakah memungkinkan untuk menggunakan layanan yang sudah tersedia untuk beban kerja yang lain?
- Hitung biaya transfer data: Gunakan Penetapan Harga AWS yang pemodelan transfer datanya sudah Anda buat sebelumnya untuk menghitung biaya transfer data untuk beban kerja tersebut. Hitung biaya transfer data di berbagai tingkat penggunaan, baik saat penggunaan beban kerja berkurang maupun bertambah. Jika ada banyak opsi arsitektur beban kerja, maka hitung biaya untuk setiap opsi sebagai perbandingan.
- Menghubungkan pengeluaran biaya dengan hasil yang diperoleh: Untuk setiap biaya transfer data yang dikenakan, tentukan hasil yang akan diperoleh dari transfer data tersebut untuk beban kerja tertentu. Jika transfer antarkomponen, hasilnya mungkin untuk pemisahan. Jika dilakukan di antara Zona Ketersediaan, tujuannya mungkin untuk redundansi.
- Buat pemodelan transfer data: Setelah mengumpulkan semua informasi, buatlah pemodelan transfer data dasar konseptual untuk beberapa kasus penggunaan dan beban kerja yang berbeda.

Sumber daya

Dokumen terkait:

- AWS solusi caching

- AWS Penetapan Harga
- EC2Harga Amazon
- VPCHarga Amazon
- Memahami biaya transfer data

Video terkait:

- Memantau dan Mengoptimalkan Biaya Transfer Data Anda
- Akselerasi Transfer S3

Contoh terkait:

- Ikhtisar Biaya Transfer Data untuk Arsitektur Umum
- AWS Panduan Preskriptif untuk Jaringan

COST08-BP02 Pilih komponen untuk mengoptimalkan biaya transfer data

Semua komponen diseleksi, dan arsitektur didesain untuk mengurangi biaya transfer data. Ini termasuk menggunakan komponen seperti wide-area-network (WAN) optimasi dan konfigurasi Multi-Availability Zone (AZ)

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Perancangan untuk transfer data meminimalkan biaya transfer data. Ini mungkin melibatkan penggunaan jaringan pengiriman untuk menemukan data yang lebih dekat dengan pengguna, atau penggunaan tautan jaringan khusus dari lokasi Anda ke AWS. Anda juga dapat menggunakan WAN optimasi dan optimasi aplikasi untuk mengurangi jumlah data yang ditransfer antar komponen.

Saat mentransfer data ke atau di dalam AWS Cloud, penting untuk mengetahui tujuan berdasarkan berbagai kasus penggunaan, sifat data, dan sumber daya jaringan yang tersedia untuk memilih AWS layanan yang tepat untuk mengoptimalkan transfer data. AWS menawarkan berbagai layanan transfer data yang disesuaikan untuk beragam persyaratan migrasi data. Pilih opsi penyimpanan data dan transfer data yang tepat berdasarkan kebutuhan bisnis dalam organisasi Anda.

Saat merencanakan atau meninjau arsitektur beban kerja Anda, pertimbangkan hal berikut:

- Gunakan VPC titik akhir dalam AWS: VPC titik akhir memungkinkan koneksi pribadi antara layanan Anda VPC dan layanan yang didukung AWS . Hal ini memungkinkan Anda untuk menghindari penggunaan internet publik, yang dapat menimbulkan biaya transfer data.
- Gunakan NAT gateway: Gunakan NATgateway sehingga instance di subnet pribadi dapat terhubung ke internet atau ke layanan di luar Anda. VPC Periksa apakah sumber daya di belakang NAT gateway yang mengirim lalu lintas terbanyak berada di Availability Zone yang sama dengan NAT gateway. Jika tidak, buat NAT gateway baru di Availability Zone yang sama dengan sumber daya untuk mengurangi biaya transfer data lintas-AZ.
- Gunakan AWS Direct Connect AWS Direct Connect bypass internet publik dan membuat koneksi pribadi langsung antara jaringan lokal Anda dan. AWS Hal ini bisa lebih hemat biaya dan konsisten daripada mentransfer data dalam jumlah besar melalui internet.
- Hindari mentransfer data melintasi batas Regional: Transfer data antara Wilayah AWS (dari satu Wilayah ke Wilayah lain) biasanya dikenakan biaya. Sebaiknya putuskan dengan cermat jika ingin memilih jalur multi-Wilayah. Untuk detail selengkapnya, lihat skenario Multi-Wilayah.
- Pantau transfer data: Gunakan Amazon CloudWatch dan VPCflow log untuk menangkap detail tentang transfer data dan penggunaan jaringan Anda. Analisis informasi lalu lintas jaringan yang ditangkap di AndaVPCs, seperti alamat IP atau jangkauan ke dan dari antarmuka jaringan.
- Analisis penggunaan jaringan Anda: Gunakan alat pengukuran dan pelaporan seperti AWS Cost Explorer, CUDOS Dasbor, atau CloudWatch untuk memahami biaya transfer data beban kerja Anda.

Langkah-langkah implementasi

- Pilih komponen untuk transfer data: Gunakan pemodelan transfer data yang dijelaskan di COST08-BP01 Lakukan pemodelan transfer data, fokus pada area dengan biaya transfer data terbesar atau area lain jika penggunaan beban kerja berubah. Cari arsitektur alternatif, atau komponen tambahan yang menghapus atau mengurangi kebutuhan untuk transfer data, atau menghemat biayanya.

Sumber daya

Praktik-praktik terbaik terkait:

- COST08-BP01 Lakukan pemodelan transfer data
- COST08-BP03 Menerapkan layanan untuk mengurangi biaya transfer data

Dokumen terkait:

- Migrasi Data Cloud
- solusi penerapan cache AWS
- Mengirimkan konten lebih cepat dengan Amazon CloudFront

Contoh terkait:

- Ikhtisar Biaya Transfer Data untuk Arsitektur Umum
- AWS Tips Optimasi Jaringan
- Optimalkan kinerja dan kurangi biaya untuk analitik jaringan dengan VPC Flow Logs dalam format Apache Parquet

COST08-BP03 Menerapkan layanan untuk mengurangi biaya transfer data

Implementasikan layanan untuk mengurangi transfer data. Misalnya, gunakan lokasi tepi atau jaringan pengiriman konten (CDN) untuk mengirimkan konten ke pengguna akhir, membangun lapisan caching di depan server aplikasi atau database Anda, dan menggunakan koneksi jaringan khusus alih-alih VPN untuk konektivitas ke cloud.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Ada berbagai AWS layanan yang dapat membantu Anda mengoptimalkan penggunaan transfer data jaringan Anda. Tergantung komponen beban kerja, jenis, dan arsitektur cloud Anda, semua layanan ini dapat membantu Anda mengompresi, melakukan caching, serta berbagi dan mendistribusikan lalu lintas Anda di cloud.

- Amazon CloudFront adalah jaringan pengiriman konten global yang memberikan data dengan latensi rendah dan kecepatan transfer tinggi. Layanan ini meng-cache data di lokasi edge di seluruh dunia, yang mengurangi beban di sumber daya Anda. Dengan menggunakan CloudFront, Anda dapat mengurangi upaya administratif dalam mengirimkan konten ke sejumlah besar pengguna secara global dengan latensi minimum. Paket penghematan keamanan dapat membantu Anda menghemat hingga 30% untuk CloudFront penggunaan Anda jika Anda berencana untuk meningkatkan penggunaan Anda dari waktu ke waktu.

- AWS Direct Connect memungkinkan Anda membangun sebuah koneksi jaringan khusus ke AWS. Ini dapat mengurangi biaya jaringan, meningkatkan bandwidth, dan menyediakan pengalaman jaringan yang lebih konsisten daripada koneksi berbasis internet.
- AWS VPN memungkinkan Anda membangun koneksi yang aman dan privat antara jaringan privat Anda dan jaringan AWS global. Layanan ini ideal untuk kantor kecil atau partner bisnis karena menyediakan konektivitas yang disederhanakan, serta dengan layanan yang dikelola penuh dan elastis.
- VPCEndpoint memungkinkan konektivitas antar AWS layanan melalui jaringan pribadi dan dapat digunakan untuk mengurangi transfer data publik dan biaya NATgateway. VPCTitik akhir Gateway tidak memiliki biaya per jam, dan mendukung Amazon S3 dan Amazon DynamoDB. VPCEndpoint antarmuka disediakan oleh AWS PrivateLinkdan memiliki biaya per jam dan biaya penggunaan per GB.
- NATgateway menyediakan penskalaan dan manajemen bawaan untuk mengurangi biaya dibandingkan dengan instance mandiri. NAT Tempatkan NAT gateway di Availability Zone yang sama dengan instans lalu lintas tinggi dan pertimbangkan untuk menggunakan VPC titik akhir untuk instans yang perlu mengakses Amazon DynamoDB atau Amazon S3 untuk mengurangi biaya transfer dan pemrosesan data.
- Gunakan AWS Snow Familyperangkat yang memiliki sumber daya komputasi untuk mengumpulkan dan memproses data di tepi. AWS Snow Family perangkat (Snowcone, Snowball, dan Snowmobile) memungkinkan Anda memindahkan petabyte data ke biaya secara efektif dan offline. AWS Cloud

Langkah-langkah implementasi

- Menerapkan layanan: Pilih layanan AWS jaringan yang berlaku berdasarkan jenis beban kerja layanan Anda menggunakan pemodelan transfer data dan meninjau Log VPC Aliran. Amati area dengan biaya terbesar dan alur volume tertinggi. Tinjau AWS layanan dan menilai apakah ada layanan yang mengurangi atau menghapus transfer, khususnya jaringan dan pengiriman konten. Cari juga layanan caching yang menyediakan akses berulang ke data, atau data dalam jumlah besar.

Sumber daya

Dokumen terkait:

- AWS Direct Connect

- AWS Jelajahi Produk Kami
- solusi penerapan cache AWS
- Amazon CloudFront
- AWS Snow Family
- Bundel Tabungan CloudFront Keamanan Amazon

Video terkait:

- Memantau dan Mengoptimalkan Biaya Transfer Data Anda
- AWS Seri Optimalisasi Biaya: CloudFront
- Bagaimana cara mengurangi biaya transfer data untuk NAT gateway saya?

Contoh terkait:

- Cara melakukan chargeback layanan bersama: Sebuah contoh AWS Transit Gateway
- Memahami detail transfer AWS data secara mendalam dari laporan biaya dan penggunaan menggunakan kueri Athena dan QuickSight
- Ikhtisar Biaya Transfer Data untuk Arsitektur Umum
- Menggunakan AWS Cost Explorer untuk menganalisis biaya transfer data
- Mengoptimalkan biaya AWS arsitektur Anda dengan memanfaatkan fitur Amazon CloudFront
- Bagaimana cara mengurangi biaya transfer data untuk NAT gateway saya?

## Kelola sumber daya pasokan dan permintaan

Pertanyaan

- COST9. Bagaimana cara mengelola permintaan dan memasok sumber daya?

### COST9. Bagaimana cara mengelola permintaan dan memasok sumber daya?

Untuk beban kerja yang memiliki pengeluaran dan performa seimbang, pastikan semua beban kerja yang Anda bayar akan digunakan dan hindari tingkat penggunaan instans yang terlalu rendah. Metrik pemanfaatan miring di kedua arah memiliki dampak buruk pada organisasi Anda, baik dalam biaya operasional (kinerja yang menurun karena pemanfaatan yang berlebihan), atau pengeluaran yang terbuang AWS (karena penyediaan berlebihan).

Praktik terbaik

- COST09-BP01 Melakukan analisis pada permintaan beban kerja
- COST09-BP02 Menerapkan buffer atau throttle untuk mengelola permintaan
- COST09-BP03 Menyediakan sumber daya secara dinamis

COST09-BP01 Melakukan analisis pada permintaan beban kerja

Analisis permintaan beban kerja dari waktu ke waktu. Verifikasikan bahwa analisis ini mencakup tren musiman dan merepresentasikan secara akurat kondisi operasi di sepanjang masa pakai beban kerja penuh. Upaya analisis harus mencerminkan potensi manfaat, misalnya, waktu yang digunakan sebanding dengan biaya beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Analisis permintaan beban kerja untuk komputasi cloud melibatkan pemahaman pola dan karakteristik tugas komputasi yang dimulai di lingkungan cloud. Analisis ini membantu pengguna mengoptimalkan alokasi sumber daya, mengelola biaya, dan memastikan tingkat kinerja yang diperlukan.

Ketahui persyaratan beban kerja. Persyaratan organisasi Anda harus menunjukkan waktu respons beban kerja untuk permintaan. Waktu respons dapat digunakan untuk menentukan apakah permintaan dikelola, atau apakah pasokan sumber daya harus berubah untuk memenuhi permintaan.

Analisis harus mencakup prediktabilitas dan pengulangan permintaan, tingkat perubahan dalam permintaan, serta jumlah perubahan dalam permintaan. Lakukan analisis selama periode yang cukup lama untuk memasukkan varians musiman apa pun, seperti end-of-month pemrosesan atau puncak liburan.

Upaya analisis harus mencerminkan potensi manfaat implementasi penskalaan. Amati perkiraan total biaya komponen serta peningkatan atau penurunan penggunaan dan biaya di sepanjang masa pakai beban kerja.

Berikut ini adalah beberapa aspek kunci yang perlu dipertimbangkan saat melakukan analisis permintaan beban kerja untuk komputasi cloud:

1. Pemanfaatan sumber daya dan metrik kinerja: Menganalisis bagaimana AWS sumber daya digunakan dari waktu ke waktu. Tentukan pola penggunaan puncak dan di luar puncak untuk mengoptimalkan alokasi sumber daya dan strategi penskalaan. Pantau metrik kinerja seperti waktu

respons, latensi, throughput, dan tingkat kesalahan. Metrik-metrik ini membantu menilai kondisi dan efisiensi infrastruktur cloud secara keseluruhan.

2. Perilaku penskalaan pengguna dan aplikasi: Memahami perilaku pengguna dan bagaimana hal itu memengaruhi permintaan beban kerja. Pemeriksaan pola lalu lintas pengguna bermanfaat dalam meningkatkan pengiriman konten dan responsivitas aplikasi. Analisis bagaimana beban kerja diskalakan seiring meningkatnya permintaan. Tentukan apakah parameter penskalaan otomatis dikonfigurasi dengan benar dan efektif untuk menangani fluktuasi beban.
3. Jenis-jenis beban kerja: Identifikasi berbagai jenis beban kerja yang berjalan di cloud, seperti pemrosesan batch, pemrosesan data waktu nyata, aplikasi web, basis data, atau machine learning. Setiap jenis beban kerja mungkin memiliki persyaratan sumber daya dan profil kinerja yang berbeda-beda.
4. Perjanjian tingkat layanan (SLAs): Bandingkan kinerja aktual dengan SLAs untuk memastikan kepatuhan dan mengidentifikasi area yang perlu diperbaiki.

Anda dapat menggunakan Amazon CloudWatch untuk mengumpulkan dan melacak metrik, memantau file log, mengatur alarm, dan secara otomatis bereaksi terhadap perubahan sumber daya Anda AWS . Anda juga dapat menggunakan Amazon CloudWatch untuk mendapatkan visibilitas seluruh sistem ke dalam pemanfaatan sumber daya, kinerja aplikasi, dan kesehatan operasional.

Dengan AWS Trusted Advisor, Anda dapat menyediakan sumber daya Anda sesuai praktik terbaik untuk meningkatkan kinerja dan keandalan sistem, meningkatkan keamanan, dan mencari peluang penghematan biaya. Anda juga dapat menonaktifkan instans non-produksi dan menggunakan Amazon CloudWatch dan Auto Scaling untuk mencocokkan kenaikan atau pengurangan permintaan.

Terakhir, Anda dapat menggunakan AWS Cost Exploreratau Amazon QuickSight dengan file AWS Cost and Usage Report (CUR) atau log aplikasi Anda untuk melakukan analisis lanjutan permintaan beban kerja.

Secara keseluruhan, analisis permintaan beban kerja yang komprehensif memungkinkan organisasi untuk mengambil keputusan yang berdasar tentang penyediaan sumber daya, penskalaan, dan pengoptimalan, yang menghasilkan perbaikan kinerja, efisiensi biaya, dan kepuasan pengguna.

Langkah-langkah implementasi

- Analisis data beban kerja yang ada: Analisis data dari beban kerja yang ada, versi beban kerja sebelumnya, atau pola penggunaan yang diprediksi. Gunakan Amazon CloudWatch, file log, dan data pemantauan untuk mendapatkan wawasan tentang bagaimana beban kerja digunakan. Analisis siklus penuh beban kerja, dan kumpulkan data untuk setiap perubahan musiman

seperti end-of-month atau end-of-year peristiwa. Upaya yang tercermin dalam analisis harus mencerminkan karakteristik beban kerja. Upaya terbesar harus ditempatkan pada beban kerja bernilai tinggi dengan perubahan permintaan terbesar. Upaya terkecil harus ditempatkan pada beban kerja bernilai rendah dengan perubahan permintaan yang minim.

- Perkirakan pengaruh dari luar: Temui anggota tim dari seluruh organisasi yang dapat memengaruhi atau mengubah permintaan pada beban kerja. Tim umum terdiri dari penjualan, pemasaran, atau pengembangan bisnis. Bekerjalah dengan mereka untuk mengetahui siklus operasi mereka, dan apakah ada peristiwa yang akan mengubah permintaan beban kerja. Prakirakan permintaan beban kerja dengan data ini.

Sumber daya

Dokumen terkait:

- Amazon CloudWatch
- AWS Trusted Advisor
- AWS X-Ray
- AWS Auto Scaling
- Penjadwal Instans AWS
- Memulai dengan Amazon SQS
- AWS Cost Explorer
- Amazon QuickSight

Video terkait:

Contoh terkait:

- Pantau, Lacak, dan Analisis untuk pengoptimalan biaya
- Mencari dan menganalisis log di CloudWatch

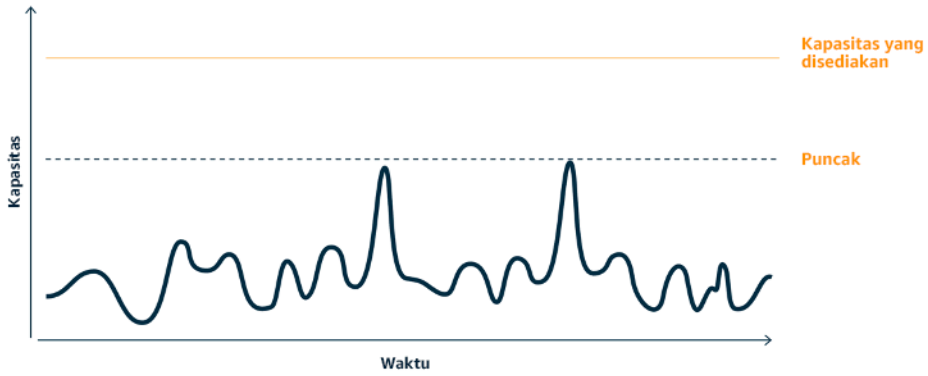
COST09-BP02 Menerapkan buffer atau throttle untuk mengelola permintaan

Buffering dan throttling memodifikasi permintaan di beban kerja Anda, sehingga akan menstabilkan fluktuasi. Implementasikan throttling ketika klien Anda mencoba ulang. Implementasikan buffering untuk menyimpan permintaan dan menunda pemrosesan ke lain waktu. Pastikan throttle dan buffer Anda didesain sehingga klien menerima respons dalam waktu yang diperlukan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Mengimplementasikan buffer atau throttle sangat penting dalam komputasi cloud untuk mengelola permintaan dan mengurangi penyediaan kapasitas yang diperlukan untuk beban kerja Anda. Untuk performa yang optimal, penting agar Anda mengukur total permintaan, termasuk puncak, laju perubahan permintaan, dan waktu respons yang diperlukan. Ketika klien memiliki kemampuan untuk mengirim ulang permintaan mereka, maka penerapan throttling menjadi praktis. Sebaliknya, untuk klien yang tidak memiliki fungsi coba ulang, mengimplementasikan solusi buffer merupakan pendekatan yang ideal. Buffer tersebut itu merampingkan masuknya permintaan dan mengoptimalkan interaksi aplikasi dengan kecepatan operasional yang bervariasi.




Kurva permintaan dengan dua puncak berbeda yang membutuhkan kapasitas penyediaan tinggi

Asumsikan beban kerja dengan kurva permintaan yang ditunjukkan pada gambar berikut. Beban kerja ini memiliki dua puncak, dan untuk menangani puncak-puncak ini, disediakan kapasitas sumber daya sebagaimana yang ditunjukkan oleh garis berwarna oranye. Sumber daya dan energi yang digunakan untuk beban kerja ini tidak ditunjukkan dengan area kurva permintaan, tetapi dengan area garis kapasitas yang disediakan, karena kapasitas yang disediakan diperlukan untuk menangani kedua puncak ini. Meratakan kurva permintaan beban kerja dapat membantu Anda mengurangi kapasitas tersedia untuk beban kerja dan mengurangi dampaknya terhadap lingkungan. Untuk memperlancar puncak, pertimbangkan untuk mengimplementasikan solusi throttling atau buffering.

Untuk memahaminya lebih lanjut, mari kita bahas apa itu throttling dan buffering.

Throttling: Jika sumber permintaan memiliki kemampuan coba ulang, maka Anda dapat mengimplementasikan throttling. Throttling memberi tahu sumber bahwa jika sistem saat ini tidak dapat melayani permintaan, sumber harus mencoba lagi nanti. Sumber menunggu untuk jangka waktu tertentu, lalu mencoba kembali permintaan. Implementasi throttling memiliki

manfaat membatasi jumlah maksimum sumber daya dan biaya beban kerja. Di AWS, Anda dapat menggunakan Amazon API Gateway untuk menerapkan throttling.

Berbasis buffer: Sebuah pendekatan berbasis buffer menggunakan produsen (komponen yang mengirimkan pesan ke antrean), konsumen (komponen yang menerima pesan dari antrean), dan antrean (yang menyimpan pesan) untuk menyimpan pesan. Pesan dibaca oleh konsumen dan diproses, sehingga pesan dapat dijalankan dengan tingkat yang memenuhi persyaratan-persyaratan bisnis konsumen. Dengan menggunakan metodologi buffer-sentris, pesan dari produsen disimpan dalam antrean atau aliran, siap diakses oleh konsumen dalam kecepatan yang selaras dengan tuntutan operasional mereka.

Di AWS, Anda dapat memilih dari beberapa layanan untuk menerapkan pendekatan buffering. Amazon Simple Queue Service (AmazonSQS) adalah layanan terkelola yang menyediakan antrian yang memungkinkan satu konsumen membaca pesan individual. Amazon Kinesis memberikan aliran yang memungkinkan banyak konsumen untuk membaca pesan yang sama.

Buffering dan throttling dapat memperlancar setiap puncak dengan memodifikasi permintaan pada beban kerja Anda. Gunakan throttling saat klien mencoba kembali tindakan dan gunakan buffering untuk menahan permintaan dan memprosesnya nanti. Ketika menggunakan pendekatan berbasis buffer, rancang dan konfigurasi beban kerja untuk melayani permintaan dalam waktu yang diperlukan, verifikasi bahwa Anda dapat menangani permintaan kerja duplikat. Lakukan analisis terhadap permintaan secara keseluruhan, tingkat perubahan, dan waktu respons yang diperlukan untuk ukuran throttling atau buffering yang tepat.

Langkah-langkah implementasi

- Analisis kebutuhan klien: Analisis permintaan klien untuk menentukan apakah klien ini mampu melakukan percobaan ulang. Untuk klien yang tidak dapat melakukan percobaan ulang, buffer perlu diimplementasikan. Analisis permintaan secara keseluruhan, tingkat perubahan, dan waktu respons yang diperlukan untuk menentukan ukuran throttle atau buffer yang diperlukan.
- Menerapkan buffer atau throttling: Menerapkan buffer atau throttle di beban kerja. Antrian seperti Amazon Simple Queue Service SQS (Amazon) dapat menyediakan buffer ke komponen beban kerja Anda. Amazon API Gateway dapat menyediakan pembatasan untuk komponen beban kerja Anda.

Sumber daya

Praktik-praktik terbaik terkait:

- SUS02-BP06 Menerapkan buffering atau throttling untuk meratakan kurva permintaan
- REL05-BP02 Permintaan Throttle

Dokumen terkait:

- AWS Auto Scaling
- Penjadwal Instans AWS
- APIGerbang Amazon
- Amazon Simple Queue Service
- Memulai dengan Amazon SQS
- Amazon Kinesis

Video terkait:

- Memilih Layanan Pesan yang Tepat untuk Aplikasi Terdistribusi Anda

Contoh terkait:

- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Melambat skala REST API multi-tenant berjenjang menggunakan Gateway API
- Mengaktifkan Tiering dan Throttling dalam Solusi Amazon SaaS EKS Multi-Tenant Menggunakan Amazon Gateway API
- Integrasi Aplikasi Menggunakan Antrian dan Pesan

COST09-BP03 Menyediakan sumber daya secara dinamis

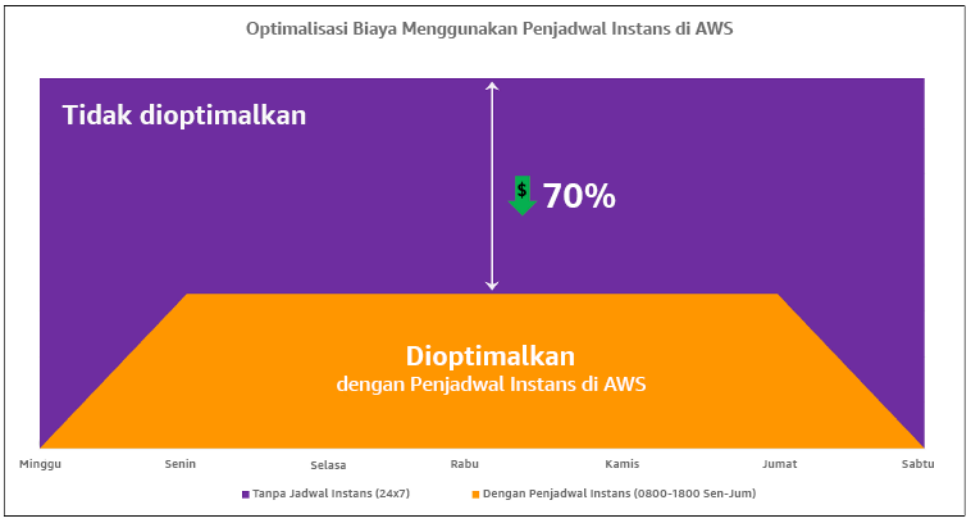
Sumber daya disediakan sesuai dengan perencanaan. Penyediaan dapat berdasarkan permintaan, yaitu melalui penskalaan otomatis, atau berdasarkan waktu, yaitu permintaan dapat diprediksi dan sumber daya disediakan berdasarkan waktu. Metode ini dapat meminimalkan penyediaan yang terlalu banyak atau terlalu sedikit.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Ada beberapa cara bagi AWS pelanggan untuk meningkatkan sumber daya yang tersedia untuk aplikasi mereka dan menyediakan sumber daya untuk memenuhi permintaan. Salah satu opsi ini adalah menggunakan AWS Instance Scheduler, yang mengotomatiskan memulai dan menghentikan instans Amazon Elastic Compute Cloud (AmazonEC2) dan Amazon Relational Database Service (Amazon). RDS Opsi lainnya adalah menggunakan AWS Auto Scaling, yang memungkinkan Anda untuk secara otomatis menskalakan sumber daya komputasi Anda berdasarkan permintaan aplikasi atau layanan Anda. Memasok sumber daya berdasarkan permintaan akan memungkinkan Anda membayar sumber daya yang Anda gunakan saja, menekan biaya dengan meluncurkan sumber daya hanya saat dibutuhkan, dan menghentikannya saat tidak dibutuhkan.

AWS Penjadwal Instance memungkinkan Anda mengonfigurasi penghentian dan awal RDS instans Amazon EC2 dan Amazon pada waktu yang ditentukan sehingga Anda dapat memenuhi permintaan sumber daya yang sama dalam pola waktu yang konsisten seperti setiap hari pengguna mengakses EC2 instans Amazon pada pukul delapan pagi yang tidak mereka butuhkan setelah jam enam malam. Solusi ini membantu mengurangi biaya operasional dengan menghentikan sumber daya yang tidak digunakan dan memulainya saat diperlukan.




Optimalisasi biaya dengan AWS Instance Scheduler.

Anda juga dapat dengan mudah mengonfigurasi jadwal untuk EC2 instans Amazon di seluruh akun dan Wilayah Anda dengan antarmuka pengguna (UI) sederhana menggunakan Pengaturan AWS Systems Manager Cepat. Anda dapat menjadwalkan RDS instans Amazon EC2 atau Amazon dengan Penjadwal AWS Instance dan Anda dapat menghentikan dan memulai instans yang ada. Namun, Anda tidak dapat menghentikan dan memulai instance yang merupakan bagian dari grup

Auto Scaling ASG () atau yang mengelola layanan seperti Amazon Redshift atau Amazon Service. OpenSearch Grup Auto Scaling (penskalaan otomatis) memiliki penjadwalannya sendiri untuk instans-instans di grup dan instans-instans ini dibuat.

AWS Auto Scaling akan membantu Anda menyesuaikan kapasitas Anda untuk menjaga kinerja yang stabil dan terprediksi dengan biaya serendah mungkin untuk memenuhi permintaan yang berubah. Ini adalah layanan yang dikelola sepenuhnya dan gratis untuk meningkatkan kapasitas aplikasi Anda yang terintegrasi dengan EC2 instans Amazon dan Spot Fleets, Amazon, Amazon ECS DynamoDB, dan Amazon Aurora. Penskalaan otomatis menyediakan pencarian sumber daya secara otomatis yang dapat dikonfigurasi untuk membantu Anda menemukan sumber daya dalam beban kerja Anda, dilengkapi dengan strategi penskalaan bawaan untuk mengoptimalkan kinerja, biaya, atau keseimbangan antara keduanya, serta memberikan penskalaan prediktif untuk membantu menangani lonjakan yang rutin terjadi.

Ada beberapa opsi penskalaan yang tersedia untuk menskalakan kapasitas grup Auto Scaling Anda:

- Menjaga tingkat instance saat ini setiap saat
- Menskalakan secara manual
- Menskalakan berdasarkan jadwal
- Menskalakan berdasarkan permintaan
- Gunakan penskalaan prediktif

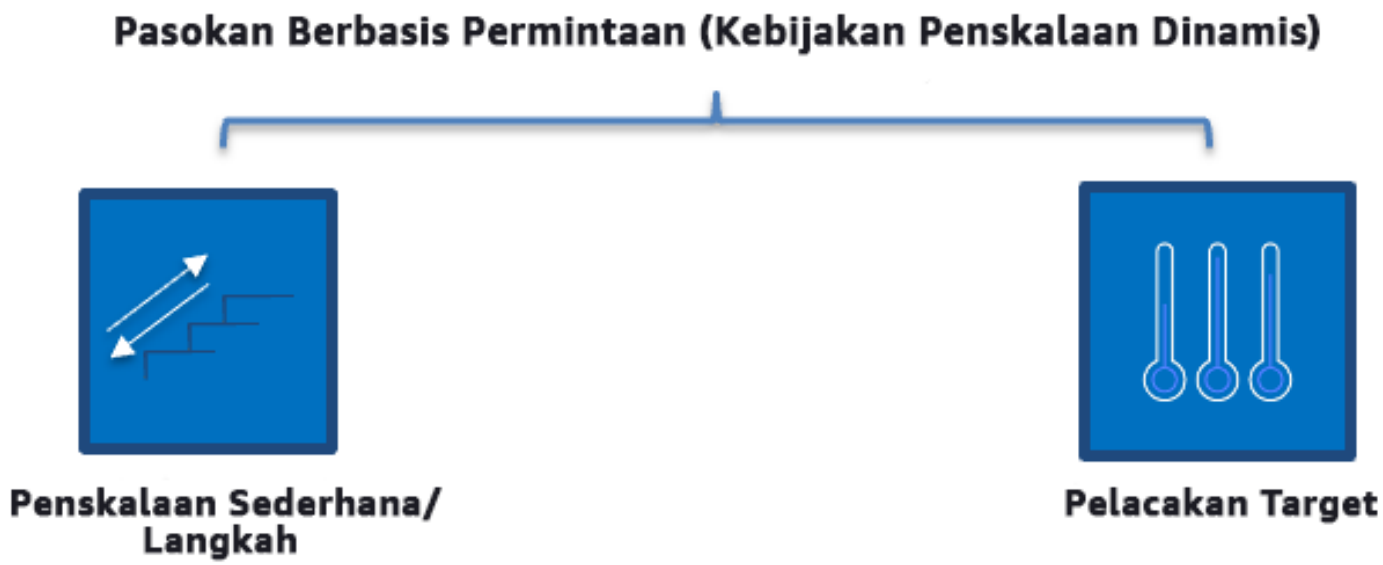
Kebijakan Auto Scaling (penskalaan otomatis) berbeda-beda dan dapat dikategorikan ke dalam kebijakan penskalaan dinamis dan terjadwal. Kebijakan dinamis ditujukan untuk penskalaan manual atau dinamis, sedangkan kebijakan terjadwal ditujukan untuk penskalaan terjadwal atau prediktif. Anda dapat menggunakan kebijakan penskalaan untuk penskalaan yang dinamis, terjadwal, dan prediktif. Anda juga dapat menggunakan metrik dan alarm dari Amazon CloudWatch untuk memicu peristiwa penskalaan untuk beban kerja Anda. Kami menyarankan Anda menggunakan templat peluncuran, yang memungkinkan Anda mengakses fitur-fitur dan peningkatan terbaru. Tidak semua fitur-fitur Auto Scaling (penskalaan otomatis) tersedia saat Anda menggunakan konfigurasi peluncuran. Misalnya, Anda tidak dapat membuat grup Auto Scaling yang meluncurkan Instans Spot dan Instans Sesuai Permintaan atau yang menentukan beberapa jenis instans. Anda harus menggunakan templat peluncuran untuk mengonfigurasi fitur ini. Saat menggunakan templat peluncuran, kami sarankan Anda melakukan penentuan versi masing-masing. Dengan penentuan versi templat peluncuran, Anda dapat membuat subset dari set lengkap parameter. Kemudian, Anda dapat menggunakannya kembali untuk membuat versi lain dari templat peluncuran yang sama.

Anda dapat menggunakan AWS Auto Scaling atau menggabungkan penskalaan dalam kode Anda dengan AWS APIsatau SDKs. Hal ini akan menghemat biaya beban kerja secara keseluruhan dengan menghilangkan biaya operasional yang diperlukan untuk membuat perubahan secara manual di lingkungan Anda, dan perubahan dapat dilakukan dengan lebih cepat. Hal ini juga menyesuaikan pengadaan sumber daya beban kerja Anda dengan permintaan Anda kapan saja. Untuk mengikuti praktik terbaik ini dan menyediakan sumber daya secara dinamis untuk organisasi Anda, Anda harus memahami penskalaan horizontal dan vertikal dalam AWS Cloud, serta sifat aplikasi yang berjalan di instans AmazonEC2. Sebaiknya tim Manajemen Keuangan Cloud Anda bekerja dengan tim teknis untuk mengikuti praktik terbaik ini.

Penyeimbang Beban Elastis (Penyeimbang Beban Elastis) membantu Anda melakukan penskalaan dengan mendistribusikan permintaan ke berbagai sumber daya. Dengan menggunakan ASG dan Elastic Load Balancing, Anda dapat mengelola permintaan masuk dengan merutekan lalu lintas secara optimal sehingga tidak ada instans yang kewalahan dalam grup Auto Scaling. Permintaan akan didistribusikan di antara semua target dari grup target secara round-robin tanpa mempertimbangkan kapasitas atau pemanfaatan.

Metrik tipikal dapat berupa EC2 metrik Amazon standar, seperti CPU pemanfaatan, throughput jaringan, dan permintaan yang diamati Elastic Load Balancing dan latensi respons. Jika memungkinkan, Anda harus menggunakan metrik yang menggambarkan pengalaman pelanggan, biasanya berupa metrik kustom yang berasal dari kode aplikasi di dalam beban kerja Anda. Untuk menguraikan cara memenuhi permintaan secara dinamis dalam dokumen ini, kami akan mengelompokkan Penskalaan Otomatis ke dalam dua kategori yakni model penyediaan berdasarkan permintaan dan berdasarkan waktu, dan kami akan menjelaskan masing-masing.

Pasokan berdasarkan permintaan: Manfaatkan elastisitas cloud untuk memasok sumber daya guna memenuhi permintaan yang berubah-ubah dengan mengandalkan kondisi permintaan yang mendekati waktu nyata. Untuk penawaran, penggunaan, APIs atau fitur layanan berbasis permintaan untuk memvariasikan jumlah sumber daya cloud dalam arsitektur Anda secara terprogram. Hal ini memungkinkan Anda untuk menskalakan komponen di arsitektur Anda, serta meningkatkan jumlah sumber daya saat permintaan melonjak guna mempertahankan kinerja, dan mengurangi kapasitas saat permintaan menurun untuk mengurangi biaya.




Kebijakan penskalaan dinamis berdasarkan permintaan

- Penskalaan Sederhana per Langkah: Memantau metrik dan menambahkan/menghapus instans sesuai langkah yang ditentukan oleh pelanggan secara manual.
- Pelacakan Target: Mekanisme kontrol mirip termostat yang secara otomatis menambahkan atau menghapus instans untuk mempertahankan metrik pada target yang ditentukan pelanggan.

Jika Anda menggunakan pendekatan berdasarkan permintaan saat merancang, selalu pertimbangkan dua hal yang utama. Pertama, ketahui seberapa cepat Anda harus menyediakan sumber daya baru. Kedua, ketahui bahwa ukuran margin antara penyediaan dan permintaan akan berubah. Anda harus siap menangani rasio perubahan dalam permintaan dan juga siap dengan kegagalan sumber daya.

Pasokan berdasarkan wkatu: Pendekatan berdasarkan waktu yang selaras dengan kapasitas sumber daya untuk permintaan yang dapat diprediksi atau telah ditentukan berdasarkan waktu. Pendekatan ini biasanya tidak bergantung pada tingkat pemanfaatan sumber daya. Pendekatan berdasarkan waktu memastikan ketersediaan sumber daya pada waktu tertentu saat diperlukan serta dapat disediakan tanpa penundaan yang disebabkan sistem dan prosedur menghidupkan atau pemeriksaan konsistensi. Menggunakan pendekatan berdasarkan waktu, Anda dapat menyediakan sumber daya tambahan atau meningkatkan kapasitas selama periode sibuk.


Kebijakan penskalaan berdasarkan waktu

Anda dapat menggunakan penskalaan otomatis yang terjadwal atau prediktif untuk menerapkan pendekatan berdasarkan waktu. Beban kerja dapat dijadwalkan untuk menambahkan skala (scale out) atau menurunkan skala (scale in) pada waktu yang ditentukan (misalnya, awal jam kerja), sehingga sumber daya tersedia saat pengguna datang atau permintaan meningkat. Penskalaan prediktif menggunakan pola untuk menambahkan skala (scale out) sedangkan penskalaan terjadwal menggunakan waktu yang ditetapkan di awal untuk menambahkan skala (scale out). Anda juga dapat menggunakan strategi pemilihan tipe instance (ABS) berbasis atribut dalam grup Auto Scaling, yang memungkinkan Anda mengekspresikan persyaratan instance sebagai sekumpulan atribut, seperti CPU v, memori, dan penyimpanan. Ini juga memungkinkan Anda untuk secara otomatis menggunakan jenis instans generasi yang lebih baru saat dirilis dan mengakses jangkauan kapasitas yang lebih luas dengan Instans EC2 Spot Amazon. Amazon EC2 Fleet dan Amazon EC2 Auto Scaling memilih dan meluncurkan instance yang sesuai dengan atribut yang ditentukan, sehingga tidak perlu memilih jenis instance secara manual.

Anda juga dapat memanfaatkan AWS APIsdan SDKs dan AWS CloudFormationsecara otomatis menyediakan dan menonaktifkan seluruh lingkungan saat Anda membutuhkannya. Pendekatan ini ideal untuk lingkungan pengujian atau pengembangan yang hanya berjalan pada jam kerja atau periode waktu yang ditentukan. Anda dapat menggunakan APIs untuk menskalakan ukuran

sumber daya dalam suatu lingkungan (penskalaan vertikal). Misalnya, Anda dapat menaikkan skala beban kerja produksi dengan mengubah ukuran atau kelas instans. Hal ini dapat dicapai dengan menghentikan lalu memulai instans, kemudian memilih kelas dan ukuran instans yang berbeda. Teknik ini juga dapat diterapkan ke sumber daya lain, seperti Amazon EBS Elastic Volume, yang dapat dimodifikasi untuk meningkatkan ukuran, menyesuaikan kinerja (IOPS) atau mengubah jenis volume saat digunakan.

Jika Anda menggunakan pendekatan berdasarkan waktu saat merancang, selalu pertimbangkan dua hal utama. Pertama, seberapa konsisten pola penggunaannya? Kedua, apa dampak dari perubahan pola tersebut? Anda dapat meningkatkan akurasi prediksi dengan memantau beban kerja Anda dan menggunakan kecerdasan bisnis. Jika Anda mendapati perubahan yang signifikan dalam pola penggunaan, Anda dapat menyesuaikan waktu untuk memastikan bahwa cakupan tersedia.

Langkah-langkah implementasi

- Konfigurasikan penskalaan terjadwal: Untuk perubahan permintaan yang dapat diprediksi, penskalaan berdasarkan waktu dapat memberikan jumlah sumber daya yang benar pada waktu yang tepat. Penskalaan ini juga bermanfaat jika pembuatan dan konfigurasi sumber daya tidak cukup cepat untuk merespons perubahan permintaan. Menggunakan analisis beban kerja untuk mengonfigurasi penskalaan terjadwal menggunakan AWS Auto Scaling. Untuk mengonfigurasi penjadwalan berbasis waktu, Anda dapat menggunakan penskalaan prediktif penskalaan terjadwal untuk meningkatkan jumlah instans Amazon EC2 di grup Auto Scaling Anda terlebih dahulu sesuai dengan perubahan beban yang diharapkan atau yang dapat diprediksi.
- Konfigurasikan penskalaan prediktif: Penskalaan prediktif memungkinkan Anda meningkatkan jumlah EC2 instans Amazon di grup Auto Scaling sebelum pola harian dan mingguan dalam arus lalu lintas. Jika Anda sering mengalami lonjakan lalu lintas dan aplikasi Anda perlu waktu lama untuk memulai, Anda sebaiknya mempertimbangkan untuk menggunakan penskalaan prediktif. Penskalaan prediktif dapat membantu Anda melakukan penskalaan lebih cepat dengan memulai kapasitas sebelum beban yang diperkirakan dibandingkan dengan penskalaan dinamis saja, yang memiliki sifat reaktif. Sebagai contoh, jika pengguna mulai menggunakan beban kerja Anda pada awal jam kerja dan tidak menggunakannya setelah jam kerja, maka penskalaan prediktif dapat menambah kapasitas sebelum jam kerja, sehingga tidak terjadi keterlambatan penskalaan dinamis dalam menanggapi perubahan lalu lintas.
- Konfigurasikan penskalaan otomatis dinamis:  Untuk mengonfigurasi penskalaan berdasarkan metrik beban kerja yang aktif, gunakan Auto Scaling. Gunakan analisis dan konfigurasikan Auto Scaling (penskalaan otomatis) untuk melakukan peluncuran pada tingkat sumber daya yang benar, dan pastikan beban kerja menskalakan pada waktu yang diinginkan. Anda dapat meluncurkan

dan secara otomatis menyesuaikan armada Instans Sesuai Permintaan dan Instans Spot dalam satu grup Auto Scaling. Selain menerima diskon untuk menggunakan Spot Instance, Anda dapat menggunakan Instans Pesanan atau Savings Plan untuk menerima tarif diskon dengan harga biasa Instans Sesuai Permintaan. Semua faktor ini digabungkan membantu Anda mengoptimalkan penghematan biaya untuk EC2 instans Amazon dan membantu Anda mendapatkan skala dan kinerja yang diinginkan untuk aplikasi Anda.

Sumber daya

Dokumen terkait:

- AWS Auto Scaling
- AWS Penjadwal Instans
- Menskalakan ukuran grup Auto Scaling Anda
- Memulai dengan Amazon EC2 Auto Scaling
- Memulai dengan Amazon SQS
- Penskalaan Terjadwal untuk EC2 Auto Scaling Amazon
- Penskalaan prediktif untuk Amazon EC2 Auto Scaling

Video terkait:

- Kebijakan Penskalaan Pelacakan Target untuk Penskalaan Otomatis
- AWS Penjadwal Instance

Contoh terkait:

- Pemilihan Jenis Instance berbasis atribut untuk Auto Scaling untuk Amazon Fleet EC2
- Mengoptimalkan biaya Layanan Kontainer Elastis Amazon dengan menggunakan penskalaan terjadwal
- Penskalaan Prediktif dengan Amazon EC2 Auto Scaling
- Bagaimana cara menggunakan Penjadwal Instance untuk AWS CloudFormation menjadwalkan EC2 instans Amazon?

# Pengoptimalan dari waktu ke waktu

Pertanyaan

- COST10. Bagaimana cara mengevaluasi layanan baru?
- COST11. Bagaimana cara mengevaluasi biaya upaya?

## COST10. Bagaimana cara mengevaluasi layanan baru?

Saat AWS merilis layanan dan fitur baru, ini adalah praktik terbaik untuk meninjau keputusan arsitektur Anda yang ada untuk memverifikasi bahwa mereka terus menjadi yang paling hemat biaya.

Praktik terbaik

- COST10-BP01 Mengembangkan proses peninjauan beban kerja
- COST10-BP02 Tinjau dan analisis beban kerja ini secara teratur

### COST10-BP01 Mengembangkan proses peninjauan beban kerja

Kembangkan proses yang menentukan kriteria dan proses untuk peninjauan beban kerja. Upaya peninjauan tersebut harus mencerminkan potensi manfaat. Misalnya, beban kerja inti atau beban kerja dengan nilai di atas sepuluh persen dari tagihan ditinjau setiap kuartal atau setiap enam bulan, sementara beban kerja di bawah sepuluh persen ditinjau setiap tahun.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

Untuk memiliki beban kerja yang paling hemat biaya, Anda harus meninjau beban kerja secara rutin untuk mengetahui apakah ada peluang untuk menerapkan layanan, fitur, dan komponen baru. Untuk mendapatkan biaya yang secara keseluruhan lebih rendah, proses harus seimbang dengan potensi jumlah penghematan. Misalnya, beban kerja yang biayanya 50% dari seluruh pengeluaran Anda harus ditinjau secara lebih rutin, dan lebih menyeluruh, daripada beban kerja yang hanya lima persen dari seluruh pengeluaran Anda. Pertimbangkan beberapa faktor eksternal atau volatilitas. Jika beban kerja melayani lokasi geografis atau segmen pasar spesifik, dan perubahan pada area tersebut diprediksi, peninjauan yang lebih sering dapat menghasilkan penghematan biaya. Faktor lain dalam peninjauan adalah upaya untuk mengimplementasikan perubahan. Jika ada biaya besar dalam pengujian dan validasi perubahan, peninjauan harus dilakukan lebih jarang.

Pertimbangkan biaya jangka panjang dalam memelihara komponen dan sumber daya yang usang dan warisan serta ketidakmampuan untuk mengimplementasikan fitur-fitur baru di dalamnya. Biaya pengujian dan validasi saat ini mungkin lebih besar dari manfaat yang diajukan. Namun, seiring waktu, biaya untuk membuat perubahan mungkin meningkat secara signifikan karena meningkatnya kesenjangan antara beban kerja dan teknologi saat ini, sehingga biayanya menjadi makin besar. Misalnya, biaya pindah ke bahasa pemrograman baru saat ini mungkin tidak hemat biaya. Namun, dalam jangka waktu lima tahun, biaya orang memiliki keterampilan bahasa tersebut mungkin meningkat, dan akibat pertumbuhan beban kerja, Anda akan memindahkan sistem yang lebih besar ke bahasa baru tersebut sehingga memerlukan upaya yang lebih banyak daripada sebelumnya.

Urai beban kerja Anda ke dalam komponen, tentukan biaya komponen (perkiraan saja sudah cukup), kemudian buat daftar faktor (misalnya, upaya dan pasar eksternal) di sebelah setiap komponen. Gunakan indikator ini untuk menentukan frekuensi peninjauan untuk setiap beban kerja. Misalnya, Anda mungkin memiliki server web dengan biaya yang tinggi, upaya perubahan yang rendah, dan faktor eksternal yang tinggi, sehingga frekuensi peninjauannya tinggi. Basis data pusat mungkin memerlukan biaya sedang, upaya perubahannya tinggi, dan faktor eksternalnya rendah, sehingga frekuensi peninjauannya sedang.

Tetapkan proses untuk mengevaluasi layanan, pola desain, tipe sumber daya, dan konfigurasi baru untuk mengoptimalkan biaya beban kerja Anda saat sudah tersedia. Sama seperti proses peninjauan pilar kinerja dan proses peninjauan pilar keandalan, identifikasi, validasi, dan prioritaskan kegiatan optimalisasi dan peningkatan serta perbaikan masalah dan masukkan ini ke dalam backlog Anda.

Langkah-langkah implementasi

- Tentukan frekuensi peninjauan:  Tentukan seberapa sering beban kerja dan komponennya harus ditinjau. Alokasikan waktu dan sumber daya untuk perbaikan kontinu dan tinjau frekuensi untuk meningkatkan efisiensi dan optimalisasi beban kerja Anda. Ini adalah kombinasi faktor yang mungkin berbeda-beda dari satu beban kerja ke beban kerja lain dalam organisasi Anda dan antara komponen-komponen dalam beban kerja tersebut. Faktor yang umum antara lain tingkat kepentingan bagi organisasi yang diukur dalam hal pendapatan dan merek, biaya total untuk menjalankan beban kerja (termasuk biaya operasi dan sumber daya), kompleksitas beban kerja, tingkat kesulitan implementasi perubahan, perjanjian lisensi perangkat lunak apa pun, dan apakah perubahan akan mendatangkan peningkatan yang signifikan pada biaya lisensi akibat lisensi yang merugikan. Komponen dapat ditentukan secara fungsi atau secara teknis, seperti server web dan basis data, atau sumber daya komputasi dan penyimpanan. Seimbangkan faktor sesuai kebutuhan dan kembangkan periode bagi beban kerja serta komponennya. Anda mungkin memutuskan untuk meninjau beban kerja penuh setiap 18 bulan, meninjau server web setiap enam bulan,

basis data setiap 12 bulan, komputasi dan penyimpanan jangka pendek setiap enam bulan, serta penyimpanan jangka panjang setiap 12 bulan.

- Tentukan keseluruhan peninjauan:  Tentukan seberapa besar upaya yang dikeluarkan untuk meninjau beban kerja atau komponen beban kerja. Seperti halnya frekuensi peninjauan, ini adalah keseimbangan antara beberapa faktor. Evaluasi dan prioritaskan peluang untuk peningkatan guna memfokuskan upaya ke hal-hal yang memberikan manfaat paling besar sekaligus memperkirakan seberapa besar upaya yang diperlukan untuk aktivitas-aktivitas tersebut. Jika hasil yang diperkirakan tidak memenuhi tujuan, dan upaya yang diperlukan memerlukan biaya lebih besar, lakukan iterasi menggunakan tindakan alternatif. Proses peninjauan Anda harus mencakup waktu dan sumber daya yang didedikasikan untuk melakukan peningkatan yang bertambah terus menerus. Sebagai contoh, Anda mungkin memutuskan untuk menghabiskan satu pekan untuk menganalisis komponen basis data, satu minggu analisis untuk sumber daya komputasi, dan empat jam untuk peninjauan penyimpanan.

Sumber daya

Dokumen terkait:

- AWS Blog Berita
- Jenis-jenis Komputasi Cloud
- Yang Baru dengan AWS

Contoh terkait:

- AWS Support Layanan Proaktif
- Ulasan beban kerja reguler untuk SAP beban kerja

COST10-BP02 Tinjau dan analisis beban kerja ini secara teratur

Beban kerja yang sudah ada ditinjau secara teratur berdasarkan setiap proses yang ditetapkan untuk mengetahui apakah layanan baru dapat diadopsi, layanan yang sudah ada dapat diganti, atau beban kerja dapat dirancang ulang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS terus menambahkan fitur baru sehingga Anda dapat bereksperimen dan berinovasi lebih cepat dengan teknologi terbaru. AWS Apa yang baru merinci bagaimana melakukan AWS ini dan memberikan gambaran singkat tentang AWS layanan, fitur, dan pengumuman ekspansi Regional saat dirilis. Anda dapat mendalami peluncuran yang telah diumumkan dan menggunakannya untuk meninjau dan menganalisis beban kerja Anda yang sudah ada. Untuk mewujudkan manfaat AWS layanan dan fitur baru, Anda meninjau beban kerja Anda dan menerapkan layanan dan fitur baru sesuai kebutuhan. Ini berarti Anda mungkin perlu mengganti layanan yang ada yang Anda gunakan untuk beban kerja Anda, atau memodernisasi beban kerja Anda untuk mengadopsi layanan baru ini. AWS Misalnya, Anda mungkin meninjau beban kerja Anda dan mengganti komponen perpesanan dengan Layanan Email Sederhana Amazon. Ini menghilangkan biaya operasi dan pemeliharaan armada instans, sekaligus memberikan semua fungsionalitas dengan harga yang lebih murah.

Untuk menganalisis beban kerja Anda dan menyorot potensi peluang, Anda juga harus mempertimbangkan cara baru untuk membangun solusi, bukan hanya tentang layanan baru. Tinjau video This is My Architecture AWS untuk mempelajari tentang desain arsitektur pelanggan lain, tantangan mereka, dan solusi mereka. Periksa seri All-In untuk mengetahui aplikasi AWS layanan dan kisah pelanggan dunia nyata. Anda juga dapat menonton seri video Kembli ke Dasar yang menjelaskan, memeriksa, dan memecah praktik terbaik pola arsitektur cloud dasar. Sumber lain adalah How to Build This video, yang dirancang untuk membantu orang-orang dengan ide-ide besar tentang cara menghidupkan produk minimum mereka (MVP) menggunakan AWS layanan. Ini adalah cara bagi pembangun dari seluruh dunia yang memiliki ide kuat untuk mendapatkan bimbingan arsitektur dari Arsitek AWS Solusi berpengalaman. Terakhir, Anda dapat meninjau materi sumber daya Memulai, yang di dalamnya terdapat tutorial langkah demi langkah.

Sebelum menjalankan proses peninjauan Anda, ikuti persyaratan bisnis Anda seputar persyaratan beban kerja, keamanan, dan persyaratan privasi data untuk menggunakan layanan tertentu atau persyaratan kinerja dan Wilayah sembari menjalani proses peninjauan yang Anda disetujui.

Langkah-langkah implementasi

- Secara rutin tunjau beban kerja: Menggunakan proses yang telah Anda tetapkan, lakukan peninjauan dengan frekuensi yang telah ditentukan. Pastikan Anda memberikan upaya yang cukup di setiap komponen. Proses ini serupa dengan proses desain awal ketika Anda memilih layanan untuk pengoptimalan biaya. Analisis layanan dan manfaatnya, kali ini perhitungkan biaya perubahan, tidak hanya manfaat jangka panjang saja.

- Terapkan layanan baru: Jika hasil analisis mengatakan perubahan harus diterapkan, pertama lakukan hal terdasar dari beban kerja untuk mengetahui biaya saat ini untuk setiap output. Implementasikan perubahan, kemudian lakukan analisis untuk mengonfirmasi biaya baru untuk setiap output.

Sumber daya

Dokumen terkait:

- AWS Blog Berita
- Yang Baru dengan AWS
- AWS Dokumentasi
- AWS Memulai
- AWS Sumber Daya Umum

Video terkait:

- AWS - Ini Arsitektur Saya
- AWS - Kembali ke Dasar-dasar
- AWS - Seri All-In
- Bagaimana Membangun Ini

## COST11. Bagaimana cara mengevaluasi biaya upaya?

Praktik terbaik

- COST11-BP01 Lakukan otomatisasi untuk operasi

### COST11-BP01 Lakukan otomatisasi untuk operasi

Evaluasi biaya operasional di cloud, dengan berfokus pada penghitungan penghematan waktu dan upaya dalam tugas-tugas administrasi, deployment, mitigasi risiko kesalahan manusia, kepatuhan, dan operasi lainnya melalui otomatisasi. Nilai waktu serta biaya terkait yang diperlukan untuk upaya operasional dan implementasikan otomatisasi untuk tugas-tugas administrasi guna meminimalkan upaya manual apabila memungkinkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Otomatisasi operasi mengurangi frekuensi tugas manual, meningkatkan efisiensi, dan menguntungkan pelanggan dengan memberikan pengalaman yang konsisten dan andal saat melakukan deployment, memberikan, atau mengoperasikan beban kerja. Anda dapat membebaskan sumber daya infrastruktur dari tugas operasional manual dan menggunakannya untuk inovasi dan tugas nilai lebih tinggi, sehingga meningkatkan nilai bisnis. Korporasi memerlukan cara yang terbukti dan teruji untuk mengelola beban kerja mereka di cloud. Solusi tersebut harus aman, cepat, dan hemat biaya, dengan risiko minimum dan keandalan maksimum.

Mulailah dengan memprioritaskan aktivitas operasional Anda berdasarkan upaya yang diperlukan dengan melihat biaya operasi secara keseluruhan. Contohnya, berapa lama waktu yang diperlukan untuk melakukan deployment sumber daya baru di cloud, membuat perubahan optimalisasi pada sumber daya yang sudah ada, atau mengimplementasikan konfigurasi-konfigurasi yang diperlukan? Lihat biaya total dari tindakan manusia dengan memperhitungkan biaya operasi dan manajemen. Prioritaskan otomatisasi untuk tugas administrasi guna mengurangi upaya manusia.

Upaya peninjauan harus mencerminkan potensi manfaat. Contohnya, periksa waktu yang diluangkan untuk melakukan tugas secara manual dibandingkan tugas otomatis. Prioritaskan otomatisasi aktivitas repetitif, bernilai tinggi, memakan waktu, dan kompleks. Aktivitas yang menimbulkan nilai tinggi atau risiko tinggi dalam hal kesalahan manusia biasanya merupakan tempat yang baik untuk memulai otomatisasi karena sering kali risikonya menimbulkan biaya operasional tambahan yang tidak diinginkan (seperti jam kerja tambahan untuk tim operasi).

Gunakan alat otomatisasi seperti AWS Systems Manager atau AWS Config untuk merampingkan operasi, kepatuhan, pemantauan, siklus hidup, dan proses penghentian. Dengan AWS layanan, alat, dan produk pihak ketiga, Anda dapat menyesuaikan otomatisasi yang Anda terapkan untuk memenuhi kebutuhan spesifik Anda. Tabel berikut menunjukkan beberapa dari kemampuan dan fungsi operasi inti yang dapat Anda capai dengan layanan AWS untuk mengotomatiskan administrasi dan operasi:

- AWS Audit Manager: Terus audit AWS penggunaan Anda untuk menyederhanakan penilaian risiko dan kepatuhan
- AWS Backup: Secara terpusat mengelola dan mengotomatiskan perlindungan data.
- AWS Config: Mengonfigurasi sumber daya komputasi, penilaian, audit, mengevaluasi konfigurasi, dan melakukan inventarisasi sumber daya.
- AWS CloudFormation: Meluncurkan sumber daya yang sangat tersedia dengan Infrastruktur sebagai Kode.

- AWS CloudTrail: Manajemen, kepatuhan, dan kontrol perubahan TI.
- Amazon EventBridge Jadwalkan acara dan pemicu AWS Lambda untuk mengambil tindakan.
- AWS Lambda: Otomatiskan proses berulang dengan memicunya dengan peristiwa atau dengan menjalankannya pada jadwal tetap dengan. AWS EventBridge
- AWS Systems Manager: Memulai dan menghentikan beban kerja, menambal sistem operasi, mengotomatiskan konfigurasi, dan manajemen berkelanjutan.
- AWS Step Functions: Jadwalkan pekerjaan dan otomatiskan alur kerja.
- AWS Service Catalog: Konsumsi templat, infrastruktur sebagai kode dengan kepatuhan dan kontrol.

Jika Anda ingin segera mengadopsi otomatisasi dengan menggunakan AWS produk dan layanan dan jika tidak memiliki keterampilan dalam organisasi Anda, hubungi AWS Managed Services (AMS), Layanan AWS Profesional, atau AWS Mitra untuk meningkatkan adopsi otomatisasi dan meningkatkan keunggulan operasional Anda di cloud.

AWS Managed Services (AMS) adalah layanan yang mengoperasikan AWS infrastruktur atas nama pelanggan dan mitra perusahaan. Layanan ini menyediakan lingkungan yang aman dan patuh untuk melakukan deployment beban kerja Anda. AMSmenggunakan model operasi cloud perusahaan dengan otomatisasi untuk memungkinkan Anda memenuhi persyaratan organisasi Anda, pindah ke cloud lebih cepat, dan mengurangi biaya manajemen yang sedang berlangsung.

AWS Layanan Profesional juga dapat membantu Anda mencapai hasil bisnis yang Anda inginkan dan mengotomatiskan operasi dengan AWS. Praktik ini membantu pelanggan melakukan deployment operasi IT otomatis yang andal dan tangkas, dan kemampuan tata kelola yang dioptimalkan untuk cloud. Untuk contoh pemantauan mendetail dan praktik terbaik yang direkomendasikan, lihat laporan resmi Pilar Keunggulan Operasional.

Langkah-langkah implementasi

- Bangun sekali dan terapkan banyak: Gunakan infrastructure-as-code seperti CloudFormation, AWS SDK, atau AWS CLI gunakan sekali dan gunakan berkali-kali untuk lingkungan serupa atau untuk skenario pemulihan bencana. Terapkan tag saat melakukan deployment untuk melacak pemakaian Anda sebagaimana ditetapkan dalam praktik terbaik lainnya. Gunakan AWS Launch Wizarduntuk mengurangi waktu untuk menyebarkan banyak beban kerja perusahaan populer. AWS Launch Wizard memandu Anda melalui ukuran, konfigurasi, dan penyebaran beban kerja perusahaan mengikuti AWS praktik terbaik. Anda juga dapat menggunakan Service Catalog, yang membantu

Anda membuat dan mengelola templat yang infrastructure-as-code disetujui untuk digunakan AWS sehingga siapa pun dapat menemukan sumber daya cloud swalayan yang disetujui.

- Otomatiskan kepatuhan berkelanjutan: Pertimbangkan untuk melakukan otomatisasi terhadap pelaksanaan evaluasi dan remediasi atas konfigurasi terekam berdasarkan standar yang telah ditentukan sebelumnya. Ketika Anda menggabungkan AWS Organizations dengan kemampuan AWS Config dan AWS CloudFormation, Anda dapat mengelola dan mengotomatiskan kepatuhan konfigurasi secara efisien dalam skala besar untuk ratusan akun anggota. Anda dapat meninjau perubahan konfigurasi dan hubungan antar AWS sumber daya dan menyelami riwayat konfigurasi sumber daya.
- Otomatiskan tugas pemantauan  AWS menyediakan berbagai alat yang dapat Anda gunakan untuk memantau layanan. Anda dapat mengonfigurasi alat-alat tersebut untuk mengotomatiskan tugas pemantauan. Buat dan implementasikan rencana pemantauan yang mengumpulkan data pemantauan dari semua bagian dalam beban kerja Anda, sehingga Anda dapat lebih mudah melakukan debugging kegagalan multititik apabila terjadi. Misalnya, Anda dapat menggunakan alat pemantauan otomatis untuk mengamati Amazon EC2 dan melaporkan kembali kepada Anda ketika ada sesuatu yang salah untuk pemeriksaan status sistem, pemeriksaan status instans, dan CloudWatch alarm Amazon.
- Otomatiskan pemeliharaan dan operasi: Jalankan operasi rutin secara otomatis tanpa campur tangan manusia. Dengan menggunakan AWS layanan dan alat, Anda dapat memilih AWS otomatisasi mana yang akan diterapkan dan disesuaikan untuk kebutuhan spesifik Anda. Misalnya, gunakan EC2Image Builder untuk membangun, menguji, dan menyebarkan mesin virtual dan gambar kontainer untuk digunakan di AWS atau di tempat atau menambal instance AndaEC2. AWS SSM Jika tindakan yang Anda inginkan tidak dapat dilakukan dengan AWS layanan atau Anda memerlukan tindakan yang lebih kompleks dengan memfilter sumber daya, maka otomatiskan operasi Anda dengan menggunakan AWS Command Line Interface(AWS CLI) atau AWS SDK alat. AWS CLI menyediakan kemampuan untuk mengotomatiskan seluruh proses pengendalian dan pengelolaan AWS layanan dengan skrip tanpa menggunakan. AWS Management Console Pilih pilihan Anda AWS SDKs untuk berinteraksi dengan AWS layanan. Untuk contoh kode lainnya, lihat Repositori contoh AWS SDK kode.
- Buat siklus hidup berkelanjutan dengan otomatisasi: Penting bagi Anda untuk membuat dan mempertahankan kebijakan siklus hidup yang matang tidak hanya untuk peraturan atau redundansi tetapi juga untuk pengoptimalan biaya. Anda dapat menggunakannya AWS Backup untuk mengelola dan mengotomatiskan perlindungan data penyimpanan data secara terpusat, seperti bucket, volume, database, dan sistem file Anda. Anda juga dapat menggunakan Amazon Data Lifecycle Manager untuk mengotomatiskan pembuatan, penyimpanan, dan penghapusan snapshot dan -backed. EBS EBS AMIs

- Hapus sumber daya yang tidak perlu: Sangat umum untuk mengakumulasi sumber daya yang tidak digunakan di kotak pasir atau pengembangan. Akun AWS Developer membuat dan bereksperimen dengan berbagai layanan dan sumber daya sebagai bagian dari siklus pengembangan normal, kemudian mereka tidak menghapus sumber daya tersebut ketika sudah tidak diperlukan. Sumber daya yang tidak digunakan dapat menimbulkan biaya yang tidak perlu dan terkadang mahal bagi organisasi. Dengan menghapus sumber daya tersebut, biaya pengoperasian lingkungan ini dapat berkurang. Pastikan data Anda tidak diperlukan atau sudah dicadangkan jika Anda tidak yakin. Anda dapat menggunakan AWS CloudFormation untuk membersihkan stack yang telah di-deploy, yang secara otomatis menghapus sebagian besar sumber daya yang ditentukan dalam templat. Atau, Anda dapat membuat otomatisasi untuk penghapusan AWS sumber daya menggunakan alat seperti aws-nuke.

Sumber daya

Dokumen terkait:

- Modernisasi operasi di AWS Cloud
- Layanan AWS untuk otomatisasi
- Infrastruktur dan Otomatisasi
- AWS Otomatisasi Systems Manager
- Pemantauan otomatis dan manual
- AWS otomatisasi untuk SAP administrasi dan operasi
- AWS Managed Services
- Layanan Profesional AWS

Video terkait:

- Otomatiskan Kepatuhan Berkelanjutan pada Skala di AWS
- AWS Backup Demo: Cadangan Lintas Akun & Lintas Wilayah
- Menambal untuk Instans Amazon EC2 Anda

Contoh terkait:

- Menyempurnakan operasi yang diotomatisasi (Bagian I)
- Menyempurnakan operasi yang diotomatisasi (Bagian II)

- Mengotomatiskan penghapusan AWS sumber daya dengan menggunakan aws-nuke
- Hapus EBS volume Amazon yang tidak digunakan dengan menggunakan AWS Config dan AWS SSM
- Otomatiskan kepatuhan berkelanjutan dalam skala besar AWS
- Otomatisasi TI dengan AWS Lambda

# Keberlanjutan

Pilar keberlanjutan mencakup pemahaman tentang dampak layanan yang digunakan, menghitung dampak melalui seluruh siklus hidup beban kerja, dan menerapkan prinsip-prinsip desain dan praktik terbaik untuk mengurangi dampak-dampak ini saat membangun beban kerja cloud. Anda dapat menemukan panduan preskriptif tentang implementasi di laporan resmi Pilar Keberlanjutan.

Area praktik terbaik

- Pemilihan wilayah
- Penyelarasan dengan permintaan
- Perangkat lunak dan arsitektur
- Data
- Perangkat keras dan layanan
- Proses dan budaya

## Pemilihan wilayah

Pertanyaan

- SUS1 Bagaimana Anda memilih Wilayah untuk beban kerja Anda?

### SUS1 Bagaimana Anda memilih Wilayah untuk beban kerja Anda?

Pilihan Wilayah untuk beban kerja Anda sangat memengaruhiKPIs, termasuk kinerja, biaya, dan jejak karbon. Untuk meningkatkan ini secara efektifKPIs, Anda harus memilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan keberlanjutan.

Praktik terbaik

- SUS01-BP01 Pilih Wilayah berdasarkan persyaratan bisnis dan tujuan keberlanjutan

SUS01-BP01 Pilih Wilayah berdasarkan persyaratan bisnis dan tujuan keberlanjutan

Pilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan sasaran keberlanjutan Anda untuk mengoptimalkannyaKPIs, termasuk kinerja, biaya, dan jejak karbon.

Anti-pola umum:

- Anda memilih Wilayah beban kerja berdasarkan lokasi Anda sendiri.
- Anda menggabungkan semua sumber daya beban kerja ke dalam satu lokasi geografis.

Manfaat menerapkan praktik terbaik ini: Menempatkan beban kerja berada dalam lokasi yang dekat dengan proyek energi terbarukan Amazon atau Wilayah dengan intensitas karbon diterbitkan yang rendah dapat membantu Anda dalam menurunkan jejak karbon dari beban kerja cloud.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS Cloud Ini adalah jaringan Wilayah dan titik kehadiran (PoP) yang terus berkembang, dengan infrastruktur jaringan global yang menghubungkan mereka bersama-sama. Pilihan Wilayah untuk beban kerja Anda sangat memengaruhiKPIs, termasuk kinerja, biaya, dan jejak karbon. Untuk meningkatkan ini secara efektifKPIs, Anda harus memilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan keberlanjutan Anda.

Langkah-langkah implementasi

- Ikuti langkah-langkah ini untuk mengevaluasi dan merangkum Wilayah potensial untuk beban kerja Anda berdasarkan persyaratan bisnis Anda, termasuk persyaratan kepatuhan, fitur yang tersedia, biaya, dan latensi:
  - Konfirmasikan bahwa Wilayah-wilayah tersebut mematuhi persyaratan peraturan setempat yang berlaku.
  - Gunakan Daftar Layanan Regional AWS untuk memeriksa apakah Wilayah-Wilayah tersebut memiliki layanan dan fitur yang Anda perlukan untuk menjalankan beban kerja Anda.
  - Hitung biaya beban kerja di setiap Wilayah dengan menggunakan AWS Pricing Calculator.
  - Uji latensi jaringan antara lokasi pengguna akhir Anda dan masing-masing Wilayah AWS lokasi.
- Pilih Wilayah yang dekat dengan proyek-proyek energi terbarukan Amazon dan Wilayah dengan jaringan energi yang memiliki intensitas karbon terpublikasi lebih rendah daripada lokasi (atau Wilayah) lain.

- Identifikasi pedoman keberlanjutan Anda yang relevan untuk melacak dan membandingkan emisi year-to-year karbon berdasarkan Protokol Gas Rumah Kaca (metode berbasis pasar dan berbasis lokasi).
- Pilih wilayah berdasarkan metode yang Anda gunakan untuk melacak emisi karbon. Untuk detail selengkapnya tentang cara memilih Wilayah berdasarkan pedoman keberlanjutan Anda, lihat Cara memilih Wilayah untuk beban kerja Anda berdasarkan tujuan keberlanjutan.

Sumber daya

Dokumen terkait:

- Memahami perkiraan emisi karbon Anda
- Amazon di Seluruh Dunia
- Metodologi Energi Terbarukan
- Hal-Hal yang Perlu Dipertimbangkan saat Memilih Wilayah untuk Beban Kerja Anda

Video terkait:

- AWS Re:invent 2023 - Inovasi keberlanjutan dalam Infrastruktur Global AWS
- AWS Re:invent 2023 - Arsitektur berkelanjutan: Masa lalu, sekarang, dan masa depan
- AWS re:invent 2022 - Memberikan arsitektur yang berkelanjutan dan berkinerja tinggi
- AWS re:invent 2022 - Merancang secara berkelanjutan dan mengurangi jejak karbon Anda AWS
- AWS re:invent 2022 - Keberlanjutan dalam infrastruktur global AWS

## Penyelarasan dengan permintaan

Pertanyaan

- SUS2 Bagaimana Anda menyelaraskan sumber daya cloud dengan permintaan Anda?

### SUS2 Bagaimana Anda menyelaraskan sumber daya cloud dengan permintaan Anda?

Cara pengguna dan aplikasi menggunakan beban kerja Anda dan sumber daya lainnya dapat membantu Anda mengidentifikasi peningkatan untuk memenuhi tujuan keberlanjutan. Skalakan infrastruktur agar dapat terus sesuai dengan permintaan dan pastikan bahwa Anda hanya

menggunakan sumber daya minimum yang diperlukan untuk mendukung para pengguna Anda. Selaraskan tingkat layanan dengan kebutuhan para pelanggan. Posisikan sumber daya guna membatasi jaringan yang diperlukan para pengguna dan aplikasi untuk memakainya. Hapus aset yang tidak digunakan. Bekali anggota tim Anda dengan perangkat yang mendukung kebutuhan mereka dan meminimalkan dampak terhadap keberlanjutan.

Praktik terbaik

- SUS02-BP01 Menskalakan infrastruktur beban kerja secara dinamis
- SUS02-BP02 Sejajar dengan tujuan keberlanjutan SLAs
- SUS02-BP03 Hentikan pembuatan dan pemeliharaan aset yang tidak terpakai
- SUS02-BP04 Optimalkan penempatan geografis beban kerja berdasarkan persyaratan jaringan mereka
- SUS02-BP05 Optimalkan sumber daya anggota tim untuk kegiatan yang dilakukan
- SUS02-BP06 Menerapkan buffering atau throttling untuk meratakan kurva permintaan

SUS02-BP01 Menskalakan infrastruktur beban kerja secara dinamis

Gunakan elastisitas cloud dan skalakan infrastruktur Anda secara dinamis untuk menyesuaikan pasokan sumber daya cloud dengan permintaan dan menghindari terjadinya kelebihan penyediaan kapasitas di beban kerja Anda.

Anti-pola umum:

- Anda tidak menskalakan infrastruktur Anda dengan beban pengguna.
- Anda menskalakan secara manual infrastruktur Anda sepanjang waktu.
- Anda membiarkan peningkatan kapasitas setelah terjadi peristiwa penskalaan, bukannya menurunkan kembali skala.

Manfaat menerapkan praktik terbaik ini: Mengkonfigurasi dan menguji elastisitas beban kerja akan Anda membantu dalam mencocokkan pasokan sumber daya cloud secara efisien dengan permintaan dan menghindari terjadinya kelebihan penyediaan kapasitas. Anda dapat memanfaatkan elastisitas di cloud untuk menskalakan kapasitas secara otomatis selama dan setelah terjadi lonjakan permintaan. Hal ini bertujuan untuk memastikan bahwa Anda hanya menggunakan jumlah sumber daya yang benar-benar diperlukan untuk memenuhi persyaratan-persyaratan bisnis Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Cloud menyediakan fleksibilitas untuk memperluas atau mengurangi sumber daya Anda secara dinamis melalui beragam mekanisme untuk memenuhi perubahan-perubahan sesuai dengan permintaan. Menyesuaikan pasokan dengan permintaan secara optimal akan memberikan dampak lingkungan terendah untuk sebuah beban kerja.

Permintaan dapat bersifat tetap atau bervariasi, sehingga akan memerlukan metrik-metrik dan otomatisasi untuk memastikan bahwa manajemen permintaan tersebut tidak akan menyulitkan. Aplikasi dapat diskalakan secara vertikal (naik atau turun) dengan mengubah ukuran instans, secara horizontal (ke dalam atau ke luar) dengan mengubah jumlah instans, atau melakukan kombinasi keduanya.

Anda dapat menggunakan sejumlah pendekatan yang berbeda untuk menyesuaikan pasokan sumber daya dengan permintaan.

- Pendekatan pelacakan target: Pantau metrik penskalaan Anda dan tingkatkan atau turunkan kapasitas secara otomatis sesuai kebutuhan.
- Penskalaan prediktif: Lakukan pengurangan skala (scale in) dalam mengantisipasi tren harian dan mingguan.
- Pendekatan berbasis jadwal: Tetapkan jadwal penskalaan Anda sendiri sesuai dengan perubahan beban yang dapat diprediksi.
- Penskalaan layanan: Pilih layanan (seperti nirserver) yang secara native menskalakan berdasarkan desain atau menyediakan penskalaan otomatis sebagai fitur.

Identifikasi periode penggunaan rendah atau nol dan skalakan sumber daya untuk menghapus kapasitas berlebih dan meningkatkan efisiensi.

Langkah-langkah implementasi

- Elastisitas menyesuaikan pasokan sumber daya yang Anda miliki dengan permintaan untuk sumber daya tersebut. Instans, kontainer, dan fungsi menyediakan mekanisme elastisitas, baik dalam kombinasi dengan penskalaan otomatis atau sebagai fitur layanan. AWS menyediakan berbagai mekanisme penskalaan otomatis untuk memastikan bahwa beban kerja dapat diturunkan dengan cepat dan mudah selama periode beban pengguna yang rendah. Berikut ini adalah beberapa contoh mekanisme penskalaan otomatis:

| Mekanisme penskalaan otomatis | Harus digunakan di mana |
| --- | --- |
| EC2Auto Scaling Amazon | Gunakan untuk memverifikasi bahwa Anda memiliki jumlah EC2 instans Amazon yang benar yang tersedia untuk menangani pemuatan pengguna untuk aplikasi Anda. |
| Penskalaan Otomatis Aplikasi | Gunakan untuk secara otomatis menskalakan sumber daya untuk AWS layanan individual di luar AmazonEC2, seperti fungsi Lambda atau layanan Amazon Elastic Container Service ECS (Amazon). |
| Penskala Otomatis Klaster Kubernetes | Gunakan untuk menskalakan cluster Kubernetes secara otomatis. AWS |

- Penskalaan sering dibahas terkait dengan layanan komputasi seperti EC2 instans atau fungsi Amazon. AWS Lambda Pertimbangkan konfigurasi layanan non-komputasi seperti unit kapasitas baca dan tulis Amazon DynamoDB atau serpihan (shard) Amazon Kinesis Data Streams agar sesuai dengan permintaan.
- Pastikan bahwa metrik-metrik untuk melakukan peningkatan atau penurunan skala telah divalidasi terhadap jenis beban kerja yang di-deploy. Jika Anda menerapkan aplikasi transcoding video, CPU pemanfaatan 100% diharapkan dan seharusnya tidak menjadi metrik utama Anda. Anda dapat menggunakan metrik kustom (seperti pemanfaatan memori) untuk kebijakan penskalaan Anda jika diperlukan. Untuk memilih metrik yang tepat, pertimbangkan panduan berikut untuk AmazonEC2:
  - Metrik tersebut harus merupakan metrik pemanfaatan yang valid dan mendeskripsikan tingkat kesibukan suatu instans.
  - Nilai metrik harus meningkatkan atau menurunkan secara proporsional jumlah instance dalam grup Auto Scaling.
- Gunakan penskalaan dinamis alih-alih penskalaan manual untuk grup Auto Scaling Anda. Kami juga menyarankan agar Anda menggunakan kebijakan penskalaan pelacakan target dalam penskalaan dinamis Anda.
- Pastikan deployment beban kerja dapat menangani peristiwa penambahan skala dan pengurangan skala. Buatlah skenario pengujian untuk peristiwa-peristiwa penambahan skala guna memastikan bahwa beban kerja berperilaku sesuai harapan dan tidak memengaruhi pengalaman pengguna

(seperti kehilangan sesi lekat (sticky session)). Anda dapat menggunakan Riwayat aktivitas untuk melakukan verifikasi terhadap aktivitas penskalaan untuk sebuah grup Auto Scaling.

- Lakukan evaluasi terhadap beban kerja Anda untuk memeriksa pola-pola terprediksi dan secara proaktif skalakan saat Anda mengantisipasi perubahan permintaan yang terencana dan terprediksi. Dengan penskalaan prediktif, Anda dapat menghilangkan kebutuhan untuk menyediakan kapasitas secara berlebih. Untuk detail selengkapnya, lihat Penskalaan Prediktif dengan Auto EC2 Scaling Amazon.

Sumber daya

Dokumen terkait:

- Memulai dengan Amazon EC2 Auto Scaling
- Penskalaan Prediktif untukEC2, Didukung oleh Machine Learning
- Menganalisis perilaku pengguna menggunakan Amazon OpenSearch Service, Amazon Data Firehose dan Kibana
- Apa itu Amazon CloudWatch?
- Memantau beban DB dengan Performance Insights di Amazon RDS
- Memperkenalkan Dukungan Asli untuk Penskalaan Prediktif dengan Amazon EC2 Auto Scaling
- Memperkenalkan Karpenter - Penskala Otomatis Klaster Kubernetes Sumber Terbuka dan Performa Tinggi
- Menyelam Jauh di Auto Scaling Amazon ECS Cluster

Video terkait:

- AWS Re:invent 2023 - Menskalakan AWS untuk 10 juta pengguna pertama
- AWS re: Invent 2023 - Arsitektur berkelanjutan: Masa lalu, sekarang, dan masa depan
- AWS re:invent 2022 - Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya
- AWS re:invent 2022 - Menskalakan wadah dari satu pengguna menjadi jutaan
- AWS re: invent 2023 - Menskalakan inferensi FM ke ratusan model dengan Amazon SageMaker
- AWS re:invent 2023 - Memanfaatkan kekuatan Karpenter untuk menskalakan, mengoptimalkan & meningkatkan Kubernetes

Contoh terkait:

- Penskalaan otomatis

SUS02-BP02 Sejajar dengan tujuan keberlanjutan SLAs

Tinjau dan optimalkan perjanjian tingkat layanan beban kerja (SLA) berdasarkan sasaran keberlanjutan Anda untuk meminimalkan sumber daya yang diperlukan untuk mendukung beban kerja Anda sambil terus memenuhi kebutuhan bisnis.

Anti-pola umum:

- Beban kerja tidak SLAs diketahui atau ambigu.
- Anda menentukan SLA hanya untuk ketersediaan dan kinerja.
- Anda menggunakan pola desain yang sama (seperti arsitektur Multi-AZ) untuk semua beban kerja Anda.

Manfaat membangun praktik terbaik ini: Menyelaraskan SLAs dengan tujuan keberlanjutan mengarah pada penggunaan sumber daya yang optimal sambil memenuhi kebutuhan bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

SLAsmenentukan tingkat layanan yang diharapkan dari beban kerja cloud, seperti waktu respons, ketersediaan, dan retensi data. SLA memengaruhi arsitektur, penggunaan sumber daya, dan dampak lingkungan yang ditimbulkan sebuah beban kerja cloud. Dengan irama reguler, tinjau SLAs dan lakukan trade-off yang secara signifikan mengurangi penggunaan sumber daya dengan imbalan penurunan tingkat layanan yang dapat diterima.

Langkah-langkah implementasi

- Memahami tujuan keberlanjutan: Identifikasi tujuan-tujuan keberlanjutan yang ditetapkan dalam organisasi Anda, seperti pengurangan karbon atau peningkatan pemanfaatan sumber daya.
- UlasanSLAs: Evaluasi Anda SLAs untuk menilai apakah mereka mendukung persyaratan bisnis Anda. Jika Anda melebihiSLAs, lakukan tinjauan lebih lanjut.
- Memahami kompromi: Memahami kompromi yang bisa dilakukan di seluruh kompleksitas beban kerja Anda (seperti pengguna yang menggunakan volume tinggi secara bersamaan), kinerja (seperti latensi), dan dampak keberlanjutan (seperti sumber daya yang diperlukan). Umumnya, ketika dua faktor diprioritaskan, faktor ketiga akan dikorbankan.

- SesuaikanSLAs: Sesuaikan Anda SLAs dengan melakukan trade-off yang secara signifikan mengurangi dampak keberlanjutan dengan imbalan penurunan tingkat layanan yang dapat diterima.
  - Keberlanjutan dan keandalan: Beban kerja dengan ketersediaan yang sangat tinggi cenderung mengkonsumsi lebih banyak sumber daya.
  - Keberlanjutan dan kinerja: Menggunakan lebih banyak sumber daya untuk meningkatkan kinerja cenderung memiliki dampak lingkungan yang lebih tinggi.
  - Keberlanjutan dan keamanan: Beban kerja yang terlalu aman cenderung memiliki dampak lingkungan yang lebih tinggi.
- Tentukan keberlanjutan SLAs jika memungkinkan: Sertakan keberlanjutan SLAs untuk beban kerja Anda. Misalnya, tentukan tingkat pemanfaatan minimum sebagai keberlanjutan SLA untuk instans komputasi Anda.
- Gunakan pola desain yang efisien: Gunakan pola desain seperti layanan mikro AWS yang memprioritaskan fungsi penting bisnis dan memungkinkan tingkat layanan yang lebih rendah (seperti waktu respons atau tujuan waktu pemulihan) untuk fungsi non-kritis.
- Berkomunikasi dan membangun akuntabilitas: Berbagi SLAs dengan semua pemangku kepentingan yang relevan, termasuk tim pengembangan Anda dan pelanggan Anda. Gunakan pelaporan untuk melacak dan memantauSLAs. Tetapkan akuntabilitas untuk memenuhi target keberlanjutan untuk Anda. SLAs
- Gunakan insentif dan penghargaan: Gunakan insentif dan penghargaan untuk mencapai atau melampaui SLAs selaras dengan tujuan keberlanjutan.
- Tinjau dan ulangi: Tinjau dan sesuaikan secara teratur SLAs untuk memastikan mereka selaras dengan keberlanjutan dan tujuan kinerja yang berkembang.

Sumber daya

Dokumen terkait:

- Memahami pola ketahanan dan kompromi untuk merancang secara efisien di cloud
- Pentingnya Perjanjian Tingkat Layanan untuk Penyedia SaaS

Video terkait:

- AWS RE: invent 2023 - Kapasitas, ketersediaan, efisiensi biaya: Pilih tiga
- AWS re: Invent 2023 - Arsitektur berkelanjutan: Masa lalu, sekarang, dan masa depan

- AWS re: invent 2023 - Pola integrasi lanjutan & trade-off untuk sistem yang digabungkan secara longgar
- AWS re:invent 2022 - Memberikan arsitektur yang berkelanjutan dan berkinerja tinggi
- AWS re:invent 2022 - Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya

SUS02-BP03 Hentikan pembuatan dan pemeliharaan aset yang tidak terpakai

Nonaktifkan aset yang tak terpakai di beban kerja Anda untuk mengurangi jumlah sumber daya cloud yang diperlukan untuk mendukung permintaan Anda dan meminimalkan limbah.

Anti-pola umum:

- Anda tidak melakukan analisis terhadap aplikasi Anda untuk mengetahui aset-aset yang redundan atau tidak diperlukan lagi.
- Anda tidak menyingkirkan aset yang redundan atau tidak diperlukan lagi.

Manfaat menerapkan praktik terbaik ini: Menghapus aset yang tidak lagi digunakan akan membebaskan sumber daya dan meningkatkan efisiensi keseluruhan beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Aset yang tak terpakai mengonsumsi sumber daya cloud seperti ruang penyimpanan dan daya komputasi. Dengan mengidentifikasi dan mengeliminasi aset-aset ini, Anda dapat membebaskan berbagai sumber daya ini, sehingga arsitektur cloud akan menjadi lebih efisien. Lakukan analisis terhadap aset-aset aplikasi secara rutin, yakni aset-aset seperti laporan pra-kompilasi, set data, gambar statis, dan pola akses aset untuk mengidentifikasi redundansi, pemanfaatan yang terlalu rendah, dan potensi target penonaktifan. Singkirkan aset-aset redundan tersebut untuk mengurangi limbah sumber daya di beban kerja Anda.

Langkah-langkah implementasi

- Lakukan inventarisasi: Lakukan inventarisasi secara komprehensif untuk mengidentifikasi semua aset yang ada dalam beban kerja Anda.
- Lakukan analisis penggunaan: Gunakan pemantauan terus-menerus untuk mengidentifikasi aset-aset statis yang tidak diperlukan lagi.

- Hapus aset yang tidak digunakan: Buatlah rencana untuk menghapus aset yang tidak lagi diperlukan.
  - Sebelum menyingkirkan aset apa pun, evaluasi terlebih dahulu dampak penyingkirannya di dalam arsitektur.
  - Gabungkan aset tumpang tindih yang dihasilkan untuk menghindari redundansi pemrosesan.
  - Perbarui aplikasi Anda hingga tidak lagi yang membuat dan menyimpan aset -aset yang tidak diperlukan.
- Komunikasikan dengan pihak ketiga: Arahkan pihak ketiga untuk berhenti memproduksi dan menyimpan aset yang dikelola atas nama Anda yang tidak diperlukan lagi. Mintalah untuk menggabungkan aset-aset redundan.
- Gunakan kebijakan siklus hidup: Gunakan kebijakan siklus hidup untuk menghapus aset yang tidak digunakan secara otomatis.
  - Anda dapat menggunakan Siklus Hidup Amazon S3 untuk mengelola objek-objek Anda di sepanjang siklus hidupnya.
  - Anda dapat menggunakan Amazon Data Lifecycle Manager untuk mengotomatiskan pembuatan, penyimpanan, dan penghapusan snapshot Amazon dan Amazon yang didukung. EBS EBS AMIs
- Tinjau dan optimalkan: Lakukan peninjauan secara teratur terhadap beban kerja Anda untuk mengidentifikasi dan menghapus aset yang tak terpakai.

Sumber daya

Dokumen terkait:

- Mengoptimalkan AWS Infrastruktur Anda untuk Keberlanjutan, Bagian II: Penyimpanan
- Bagaimana cara menghentikan sumber daya aktif yang tidak lagi saya Akun AWS perlukan?

Video terkait:

- AWS Re:invent 2023 - Arsitektur berkelanjutan: Masa lalu, sekarang, dan masa depan
- AWS re:invent 2022 - Melestarikan dan memaksimalkan nilai aset media digital menggunakan Amazon S3
- AWS Re:invent 2023 - Optimalkan biaya di lingkungan multi-akun Anda

SUS02-BP04 Optimalkan penempatan geografis beban kerja berdasarkan persyaratan jaringan mereka

Pilih layanan dan lokasi cloud untuk beban kerja Anda yang mengurangi jarak yang harus ditempuh lalu lintas jaringan dan menurunkan total sumber daya jaringan yang diperlukan untuk mendukung beban kerja Anda.

Anti-pola umum:

- Anda memilih Wilayah beban kerja berdasarkan lokasi Anda sendiri.
- Anda menggabungkan semua sumber daya beban kerja ke dalam satu lokasi geografis.
- Semua lalu lintas mengalir melalui pusat data Anda.

Manfaat menerapkan praktik terbaik ini: Menempatkan beban kerja dekat dengan penggunanya akan menghasilkan latensi terendah sambil mengurangi pergerakan data di seluruh jaringan dan mengurangi dampak lingkungan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

AWS Cloud Infrastruktur dibangun di sekitar opsi lokasi seperti Wilayah, Zona Ketersediaan, grup penempatan, dan lokasi tepi seperti AWS Outpostsdan AWS Local Zones. Opsi-opsi lokasi ini bertanggung jawab untuk memelihara konektivitas yang ada di antara komponen-komponen aplikasi, layanan cloud, jaringan edge, dan pusat data on-premise.

Lakukan analisis terhadap pola akses jaringan yang ada di beban kerja Anda untuk mengidentifikasi cara menggunakan opsi lokasi cloud ini dan mengurangi jarak yang harus ditempuh lalu lintas jaringan.

Langkah-langkah implementasi

- Lakukan analisis terhadap pola akses jaringan di beban kerja Anda untuk mengidentifikasi cara pengguna menggunakan aplikasi Anda.
  - Gunakan alat pemantauan, seperti Amazon CloudWatch dan AWS CloudTrail, untuk mengumpulkan data tentang aktivitas jaringan.
  - Analisis data untuk mengidentifikasi pola akses jaringan.
- Pilihlah Wilayah untuk deployment beban kerja Anda berdasarkan elemen-elemen utama berikut ini:

  - Tujuan Keberlanjutan Anda: seperti yang dijelaskan dalam Pemilihan Wilayah.
  - Dimana lokasi data Anda: Untuk aplikasi-aplikasi dengan banyak data (seperti big data dan machine learning), kode aplikasi harus dijalankan sedekat mungkin dengan data.
  - Dimana lokasi pengguna Anda: Untuk aplikasi-aplikasi yang ditampilkan kepada pengguna, pilihlah sebuah Wilayah (Wilayah-wilayah) yang dekat dengan para pengguna beban kerja Anda.
  - Kendala lainnya: Pertimbangkan kendala-kendala seperti biaya dan kepatuhan sebagaimana yang dijelaskan dalam Hal-Hal yang Perlu Dipertimbangkan saat Memilih Wilayah untuk Beban Kerja Anda.
- Gunakan caching lokal atau Solusi Penerapan Cache AWS untuk aset-aset yang sering digunakan untuk meningkatkan performa, mengurangi perpindahan data, dan mengurangi dampak pada lingkungan.

| Layanan | Kapan harus digunakan |
| --- | --- |
| Amazon CloudFront | Gunakan untuk menyimpan konten statis seperti gambar, skrip, dan video, serta konten dinamis seperti API respons atau aplikasi web. |
| Amazon ElastiCache | Gunakan untuk meng-cache konten bagi aplikasi web. |
| DynamoDB Accelerator | Gunakan untuk menambahkan percepatan dalam memori ke tabel DynamoDB Anda. |

- Gunakan layanan-layanan yang dapat membantu Anda menjalankan kode lebih dekat dengan para pengguna beban kerja Anda:

| Layanan | Kapan harus digunakan |
| --- | --- |
| Lambda@Edge | Gunakan untuk operasi-operasi yang memiliki banyak komputasi yang dimulai saat objek tidak ada dalam cache. |
| CloudFront Fungsi Amazon | Gunakan untuk kasus penggunaan sederhana seperti HTTP manipulasi permintaan atau |

| Layanan | Kapan harus digunakan |
| --- | --- |
| | respons yang dapat dimulai oleh fungsi berumur pendek. |
| AWS IoT Greengrass | Gunakan untuk menjalankan komputasi lokal, olahpesan, dan caching data untuk perangkat yang terhubung. |

- Gunakan pooling koneksi untuk mengizinkan penggunaan ulang koneksi dan mengurangi sumber daya yang diperlukan.
- Gunakan penyimpanan data terdistribusi yang tidak mengandalkan koneksi persisten dan pembaruan-pembaruan selaras untuk mendapatkan konsistensi guna melayani populasi wilayah.
- Ganti kapasitas jaringan statis yang disediakan di awal dengan kapasitas dinamis bersama, dan bagikan dampak keberlanjutan kapasitas jaringan kepada pelanggan lain.

Sumber daya

Dokumen terkait:

- Mengoptimalkan AWS Infrastruktur Anda untuk Keberlanjutan, BagianIII: Jaringan
- ElastiCache Dokumentasi Amazon
- Apa itu Amazon CloudFront?
- Fitur CloudFront Utama Amazon
- AWS Infrastruktur Global
- AWS Local Zones dan AWS Outposts, memilih teknologi yang tepat untuk beban kerja edge Anda
- Grup penempatan
- AWS Local Zones
- AWS Outposts

Video terkait:

- Mengungkap transfer data pada AWS
- Menskalakan kinerja jaringan pada instans Amazon generasi berikutnya EC2
- AWS Video Penjelasan Local Zones

- AWS Outposts: Ikhtisar dan Cara Kerjanya
- AWS re:invent 2023 - Strategi migrasi untuk beban kerja edge dan lokal
- AWS Re:invent 2021 - AWS Outposts: Membawa pengalaman di tempat AWS
- AWS re:invent 2020 - AWS Wavelength: Jalankan aplikasi dengan latensi ultra-rendah di tepi 5G
- AWS re:invent 2022 - AWS Local Zones: Membangun aplikasi untuk tepi terdistribusi
- AWS re:invent 2021 - Membangun situs web latensi rendah dengan Amazon CloudFront
- AWS re:invent 2022 - Tingkatkan kinerja dan ketersediaan dengan AWS Global Accelerator
- AWS re:invent 2022 - Bangun jaringan area luas global Anda menggunakan AWS
- AWS Re:invent 2020: Manajemen lalu lintas global dengan Amazon Route 53

Contoh terkait:

- AWS Lokakarya Jaringan
- Merancang arsitektur untuk keberlanjutan - Meminimalkan pergerakan data lintas jaringan

SUS02-BP05 Optimalkan sumber daya anggota tim untuk kegiatan yang dilakukan

Optimalkan sumber daya yang disediakan bagi anggota tim untuk meminimalkan dampak keberlanjutan sambil mendukung kebutuhan mereka.

Anti-pola umum:

- Anda mengabaikan dampak yang ditimbulkan oleh perangkat yang digunakan oleh anggota tim Anda terhadap efisiensi aplikasi cloud secara keseluruhan.
- Anda mengelola dan memperbarui sumber daya yang digunakan oleh anggota tim secara manual.

Manfaat menerapkan praktik terbaik ini: Mengoptimalkan sumber daya anggota tim akan meningkatkan efisiensi keseluruhan aplikasi yang berkemampuan cloud.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Pahami sumber daya yang digunakan anggota tim Anda untuk mengonsumsi layanan Anda, ekspektasi siklus hidup mereka, dan dampak finansial serta dampak pada keberlanjutan. Implementasikan strategi untuk melakukan optimalisasi terhadap berbagai sumber daya ini. Sebagai

contoh, lakukan operasi yang kompleks, seperti rendering dan kompilasi, pada infrastruktur yang dapat diskalakan dan sangat banyak digunakan, bukan pada sistem pengguna tunggal dan berdaya tinggi namun jarang digunakan.

Langkah-langkah implementasi

- Gunakan workstation hemat energi: Sediakan workstation dan periferal hemat energi kepada anggota tim. Gunakan fitur manajemen daya yang efisien (seperti mode daya rendah) di perangkat-perangkat tersebut untuk mengurangi penggunaan energinya
- gunakan virtualisasi: Gunakan streaming aplikasi dan desktop virtual untuk membatasi persyaratan perangkat dan pemutakhiran.
- Dorong kolaborasi jarak jauh: Dorong anggota tim untuk menggunakan alat kolaborasi jarak jauh seperti Amazon Chime atau AWS Wickr untuk mengurangi kebutuhan akan perjalanan dan emisi karbon terkait.
- Gunakan perangkat lunak hemat energi: Berikan anggota tim perangkat lunak yang hemat energi dengan menghapus atau mematikan fitur dan proses yang tidak perlu.
- Lakukan pengelolaan siklus hidup: Evaluasi dampak proses dan sistem atas siklus hidup perangkat, dan pilih solusi yang meminimalkan persyaratan untuk penggantian perangkat sekaligus memenuhi persyaratan bisnis. Lakukan pemeliharaan dan pembaruan secara rutin terhadap stasiun kerja atau perangkat lunak untuk menjaga dan memperbaiki efisiensi.
- Pengelolaan perangkat jarak jauh: Implementasikan manajemen jarak jauh untuk perangkat guna mengurangi perjalanan bisnis yang diperlukan.
  - AWS Systems Manager Fleet Manager adalah pengalaman antarmuka pengguna terpadu (UI) yang membantu Anda mengelola node dari jarak jauh yang berjalan di AWS atau di tempat.

Sumber daya

Dokumen terkait:

- Apa itu Amazon WorkSpaces?
- Pengoptimal Biaya untuk Amazon WorkSpaces
- Dokumentasi Amazon AppStream 2.0
- NICE DCV

Video terkait:

- Mengelola biaya untuk Amazon WorkSpaces di AWS

SUS02-BP06 Menerapkan buffering atau throttling untuk meratakan kurva permintaan

Buffering dan throttling meratakan kurva permintaan dan mengurangi kapasitas tersedia yang diperlukan untuk beban kerja Anda.

Anti-pola umum:

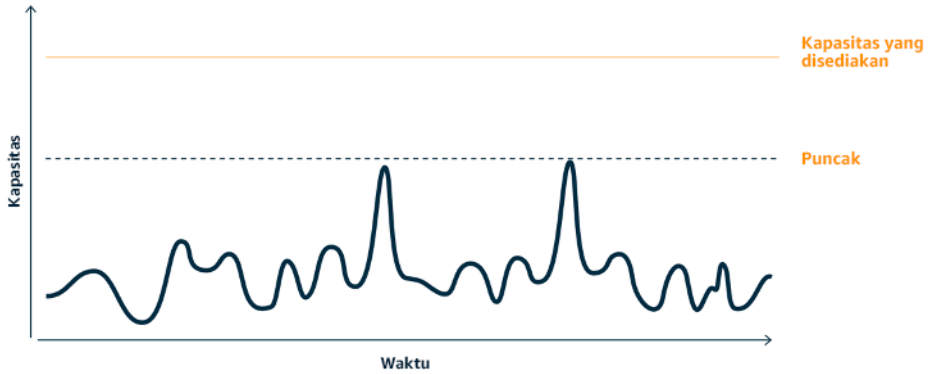
- Anda memproses permintaan klien dengan segera walaupun tidak diperlukan.
- Anda tidak menganalisis persyaratan-persyaratan untuk permintaan klien.

Manfaat menerapkan praktik terbaik ini: Dengan meratakan kurva permintaan Anda akan mengurangi kapasitas yang disediakan untuk beban kerja. Mengurangi kapasitas tersedia artinya konsumsi energi berkurang dan dampak pada lingkungan juga akan berkurang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

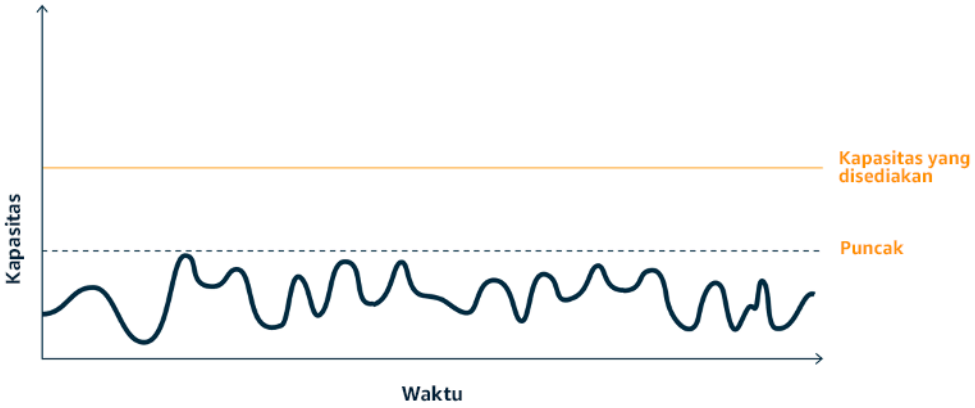
Panduan implementasi

Meratakan kurva permintaan beban kerja dapat membantu Anda mengurangi kapasitas tersedia untuk beban kerja dan mengurangi dampaknya terhadap lingkungan. Asumsikan sebuah beban kerja dengan kurva permintaan seperti yang ditunjukkan pada gambar di bawah ini. Beban kerja ini memiliki dua puncak, dan untuk menangani puncak-puncak ini, disediakan kapasitas sumber daya sebagaimana yang ditunjukkan oleh garis berwarna oranye. Sumber daya dan energi yang digunakan untuk beban kerja ini tidak diindikasikan oleh area di bawah kurva permintaan, tetapi ditunjukkan oleh area di bawah garis kapasitas tersedia, karena kapasitas tersedia diperlukan untuk menangani kedua puncak ini.



Kurva permintaan dengan dua puncak berbeda yang membutuhkan kapasitas penyediaan tinggi.

Anda dapat menggunakan buffering atau throttling untuk melakukan modifikasi terhadap kurva permintaan dan meratakan puncak, yang artinya konsumsi energi menjadi lebih sedikit energi dan penyediaan kapasitas menjadi lebih rendah. Implementasikan throttling ketika klien Anda dapat mencoba ulang. Implementasikan buffering untuk menyimpan permintaan dan menunda pemrosesan ke lain waktu.




Efek throttling pada kurva permintaan dan kapasitas penyediaan.

Langkah-langkah implementasi

- Analisis permintaan klien untuk menentukan cara merespons permintaan. Pertanyaan yang harus dipertimbangkan antara lain:
  - Dapatkah permintaan ini diproses secara tak selaras?
  - Apakah klien memiliki kemampuan untuk melakukan percobaan ulang?
- Jika klien memiliki kemampuan untuk melakukan percobaan ulang, maka Anda dapat mengimplementasikan throttling, yang memberi tahu sumber bahwa jika sumber tidak dapat melayani permintaan pada saat ini maka sumber harus mencoba lagi nanti.
  - Anda dapat menggunakan Amazon API Gateway untuk menerapkan throttling.
- Untuk klien yang tidak dapat melakukan percobaan ulang, buffering harus diimplementasikan untuk meratakan kurva permintaan. Buffering akan menunda pemrosesan permintaan, sehingga aplikasi yang dijalankan pada tingkat yang berlainan dapat berkomunikasi secara efektif. Pendekatan berbasis buffering menggunakan antrean atau aliran untuk menerima pesan dari penghasil pesan. Pesan dibaca oleh konsumen dan diproses, sehingga pesan dapat dijalankan dengan tingkat yang memenuhi persyaratan-persyaratan bisnis konsumen.

  - Amazon Simple Queue Service (AmazonSQS) adalah layanan terkelola yang menyediakan antrian yang memungkinkan satu konsumen membaca pesan individual.
  - Amazon Kinesis memberikan aliran yang memungkinkan banyak konsumen untuk membaca pesan yang sama.
- Lakukan analisis terhadap permintaan secara keseluruhan, tingkat perubahan, dan waktu respons yang diperlukan untuk ukuran throttling atau buffering yang tepat.

Sumber daya

Dokumen terkait:

- Memulai dengan Amazon SQS
- Integrasi Aplikasi Menggunakan Antrian dan Pesan
- Mengelola dan memantau API pembatasan dalam beban kerja Anda
- Melambat skala REST API multi-tenant berjenjang menggunakan Gateway API
- Integrasi Aplikasi Menggunakan Antrian dan Pesan

Video terkait:

- AWS re:invent 2022 - Pola integrasi aplikasi untuk layanan mikro
- AWS Re: invent 2023 - Penghematan cerdas: Strategi pengoptimalan biaya Amazon EC2
- AWS RE: invent 2023 - Pola integrasi lanjutan & trade-off untuk sistem yang digabungkan secara longgar

## Perangkat lunak dan arsitektur

Pertanyaan

- SUS3 Bagaimana Anda memanfaatkan pola perangkat lunak dan arsitektur untuk mendukung tujuan keberlanjutan Anda?

## SUS3 Bagaimana Anda memanfaatkan pola perangkat lunak dan arsitektur untuk mendukung tujuan keberlanjutan Anda?

Implementasikan pola untuk melancarkan beban dan mempertahankan penggunaan yang tinggi dan konsisten atas sumber daya yang di-deploy guna meminimalkan sumber daya yang dipakai. Komponen dapat menjadi tidak aktif akibat kurangnya pemakaian, karena adanya perubahan perilaku pengguna dari waktu ke waktu. Revisi pola dan arsitektur untuk menggabungkan komponen dengan pemanfaatan rendah guna meningkatkan pemanfaatan secara keseluruhan. Pensiunkan komponen-komponen yang tidak lagi diperlukan. Pahami kinerja dari komponen-komponen beban kerja Anda, dan optimalkan komponen yang memakai sumber daya terbanyak. Ketahui perangkat yang digunakan pelanggan untuk mengakses layanan Anda, dan implementasikan pola untuk meminimalkan kebutuhan pemutakhiran perangkat.

Praktik terbaik

- SUS03-BP01 Optimalkan perangkat lunak dan arsitektur untuk pekerjaan asinkron dan terjadwal
- SUS03-BP02 Hapus atau refactor komponen beban kerja dengan penggunaan rendah atau tanpa penggunaan
- SUS03-BP03 Optimalkan area kode yang paling banyak menghabiskan waktu atau sumber daya
- SUS03-BP04 Optimalkan dampak pada perangkat dan peralatan
- SUS03-BP05 Gunakan pola dan arsitektur perangkat lunak yang paling mendukung akses data dan pola penyimpanan

### SUS03-BP01 Optimalkan perangkat lunak dan arsitektur untuk pekerjaan asinkron dan terjadwal

Gunakan pola arsitektur dan perangkat lunak yang efisien seperti berbasis antrean untuk mempertahankan pemanfaatan sumber daya ter-deploy yang terus-menerus tinggi.

Anti-pola umum:

- Anda melakukan pengadaan sumber daya secara berlebihan di beban kerja cloud Anda untuk memenuhi lonjakan permintaan yang tidak terduga.
- Arsitektur Anda tidak memisahkan pengirim dan penerima pesan tak selaras berdasarkan komponen pesan.

Manfaat menjalankan praktik terbaik ini:

- Pola arsitektur dan perangkat lunak yang efisien dapat meminimalkan sumber daya tidak terpakai di dalam beban kerja Anda dan akan meningkatkan efisiensi secara keseluruhan.
- Anda dapat menskalakan pemrosesan tanpa terikat penerimaan pesan tak selaras.
- Melalui sebuah komponen perpesanan, Anda memiliki persyaratan ketersediaan yang longgar yang dapat Anda penuhi dengan sumber daya yang lebih sedikit.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Gunakan pola arsitektur yang efisien seperti arsitektur berbasis peristiwa yang dapat menghasilkan pemanfaatan komponen yang merata dan meminimalkan penyediaan berlebihan dalam beban kerja Anda. Menggunakan pola-pola arsitektur yang efisien akan meminimalkan sumber daya tidak aktif akibat kurangnya pemakaian yang disebabkan oleh perubahan permintaan seiring berjalannya waktu.

Pahami persyaratan-persyaratan komponen beban kerja Anda dan adopsi pola-pola arsitektur yang dapat meningkatkan pemanfaatan sumber daya secara keseluruhan. Pensiunkan komponen-komponen yang tidak lagi diperlukan.

Langkah-langkah implementasi

- Analisis permintaan untuk beban kerja Anda guna menentukan cara meresponsnya.
- Untuk permintaan atau tugas-tugas yang tidak memerlukan respons selaras, gunakan arsitektur berbasis antrean dan pekerja penskalaan otomatis untuk memaksimalkan pemanfaatan. Berikut ini adalah beberapa contoh kapan Anda mungkin perlu mempertimbangkan arsitektur berbasis antrean:

| Mekanisme antrean | Deskripsi |
| --- | --- |
| AWS Batch antrian pekerjaan | AWS Batch pekerjaan diserahkan ke antrian pekerjaan di mana mereka tinggal sampai mereka dapat dijadwalkan untuk berjalan di lingkungan komputasi. |
| Layanan Antrian Sederhana Amazon dan Instans EC2 Spot Amazon | Memasangkan Amazon SQS dan Instans Spot untuk membangun arsitektur yang toleran terhadap kesalahan dan efisien. |

- Untuk permintaan atau tugas yang dapat diproses kapan saja, gunakan mekanisme penjadwalan untuk memproses tugas-tugas yang ada dalam batch untuk mendapatkan efisiensi yang lebih tinggi. Berikut adalah beberapa contoh mekanisme penjadwalan pada AWS:

| Mekanisme penjadwalan | Deskripsi |
| --- | --- |
| EventBridge Penjadwal Amazon | Kemampuan dari Amazon EventBridge yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas terjadwal dalam skala besar. |
| AWS Glue jadwal berbasis waktu | Tentukan jadwal berbasis waktu untuk crawler dan pekerjaan Anda di. AWS Glue |
| Tugas terjadwal Amazon Elastic Container Service (AmazonECS) | Amazon ECS mendukung pembuatan tugas terjadwal. Tugas terjadwal menggunakan EventBridge aturan Amazon untuk menjalank an tugas baik pada jadwal atau dalam menanggapi suatu EventBridge peristiwa. |
| Penjadwal Instans | Konfigurasikan jadwal mulai dan berhenti untuk instans Amazon EC2 Relational Database Service Anda. |

- Jika Anda menggunakan mekanisme polling dan webhook dalam arsitektur Anda, maka gantilah dengan mekanisme peristiwa. Gunakan arsitektur berbasis peristiwa untuk membangun beban kerja yang sangat efisien.
- Manfaatkan nirserver di AWS untuk menghilangkan infrastruktur yang disediakan secara berlebihan.
- Sesuaikan ukuran dari setiap komponen yang ada dalam arsitektur Anda untuk menghindari sumber daya yang tidak aktif karena menunggu input.
  - Anda dapat menggunakan Rekomendasi Penyesuaian Ukuran di AWS Cost Explorer atau AWS Compute Optimizer untuk mengidentifikasi peluang penyesuaian ukuran.
  - Untuk detail selengkapnya, lihat Penentuan Ukuran yang Tepat: Menyediakan Instans yang Sesuai dengan Beban Kerja.

Sumber daya

Dokumen terkait:

- Apa itu Amazon Simple Queue Service?
- Apa itu Amazon MQ?
- Penskalaan berdasarkan Amazon SQS
- Apa itu AWS Step Functions?
- Apa itu AWS Lambda?
- Menggunakan AWS Lambda dengan Amazon SQS
- Apa itu Amazon EventBridge?
- Mengelola Alur Kerja Asinkron dengan REST API

Video terkait:

- AWS re:invent 2023 - Menavigasi perjalanan ke arsitektur berbasis peristiwa tanpa server
- AWS re:invent 2023 - Menggunakan tanpa server untuk arsitektur berbasis peristiwa & desain berbasis domain
- AWS re: invent 2023 - Pola berbasis peristiwa tingkat lanjut dengan Amazon EventBridge
- AWS re: Invent 2023 - Arsitektur berkelanjutan: Masa lalu, sekarang, dan masa depan
- Pola Pesan Asinkron | Acara AWS

Contoh terkait:

- Arsitektur berbasis peristiwa dengan AWS Prosesor Graviton dan Instans Spot Amazon EC2

SUS03-BP02 Hapus atau refactor komponen beban kerja dengan penggunaan rendah atau tanpa penggunaan

Singkirkan komponen yang tidak digunakan dan sudah tidak diperlukan, dan faktorkan ulang komponen dengan sedikit pemanfaatan, untuk meminimalkan limbah di beban kerja Anda.

Anti-pola umum:

- Anda tidak secara rutin memeriksa tingkat penggunaan masing-masing komponen beban kerja Anda.

- Anda tidak memeriksa dan menganalisis rekomendasi dari alat AWS hak cipta seperti. AWS Compute Optimizer

Manfaat menerapkan praktik terbaik ini: Menghapus komponen yang tidak terpakai dapat meminimalkan pemborosan dan meningkatkan efisiensi keseluruhan beban kerja cloud Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Lakukan peninjauan terhadap beban kerja Anda untuk mengidentifikasi komponen-komponen yang pasif atau tidak digunakan. Tindakan ini adalah proses peningkatan yang berulang, yang dapat diinisiasi oleh perubahan-perubahan yang terjadi dalam permintaan atau rilis layanan cloud baru. Misalnya, penurunan waktu fungsi AWS Lambda yang signifikan dapat menjadi indikator yang menunjukkan kebutuhan untuk menurunkan ukuran memori. Selain itu, saat AWS merilis layanan dan fitur baru, layanan dan arsitektur optimal untuk beban kerja Anda dapat berubah.

Lakukan pemantauan terus menerus terhadap aktivitas beban kerja Anda dan carilah peluang untuk meningkatkan tingkat pemanfaatan masing-masing komponen. Dengan menyingkirkan komponen-komponen yang pasif dan melakukan aktivitas penyesuaian ukuran, Anda akan memenuhi persyaratan bisnis dengan menggunakan sumber daya cloud sesedikit mungkin.

Langkah-langkah implementasi

- Miliki inventaris sumber AWS daya Anda. Di AWS, Anda dapat mengaktifkan Penjelajah Sumber Daya AWSuntuk menjelajahi dan mengatur AWS sumber daya Anda. Untuk detail selengkapnya, lihat AWS re:Invent 2022 - Cara mengelola sumber daya dan aplikasi dalam skala besar. AWS
- Pantau dan tangkap metrik pemanfaatan untuk komponen penting dari beban kerja Anda (seperti CPU pemanfaatan, pemanfaatan memori, atau throughput jaringan dalam metrik Amazon). CloudWatch
- Identifikasi komponen-komponen yang tidak digunakan atau kurang dimanfaatkan dalam arsitektur Anda.
  - Untuk beban kerja yang stabil, periksa alat AWS pengatur ukuran seperti AWS Compute Optimizersecara berkala untuk mengidentifikasi komponen yang tidak digunakan, tidak digunakan, atau kurang dimanfaatkan.
  - Untuk beban kerja sementara, lakukan evaluasi terhadap metrik-metrik pemanfaatan untuk mengidentifikasi komponen yang langsam, tidak digunakan, atau kurang dimanfaatkan.
- Pensiun komponen dan aset terkait (seperti ECR gambar Amazon) yang tidak lagi diperlukan.

  - Pembersihan Otomatis Gambar yang Tidak Digunakan di Amazon ECR
  - Hapus volume Amazon Elastic Block Store (AmazonEBS) yang tidak digunakan dengan menggunakan AWS Config dan AWS Systems Manager
- Faktor ulang atau gabungkan komponen-komponen yang kurang dimanfaatkan dengan sumber daya lain untuk meningkatkan efisiensi pemanfaatan. Misalnya, Anda dapat menyediakan beberapa database kecil pada satu instance database Amazon alih-alih menjalankan RDS database pada instans individual yang kurang dimanfaatkan.
- Pahami sumber daya yang disediakan oleh beban kerja Anda untuk menyelesaikan sebuah unit kerja.

Sumber daya

Dokumen terkait:

- AWS Trusted Advisor
- Apa itu Amazon CloudWatch?
- Penentuan Ukuran yang Tepat: Menyediakan Instans yang Sesuai dengan Beban Kerja
- Mengoptimalkan biaya Anda dengan Rekomendasi Penentuan Ukuran yang Tepat

Video terkait:

- AWS RE: invent 2023 - Kapasitas, ketersediaan, efisiensi biaya: Pilih tiga

Contoh terkait:

- Optimalkan Pola Perangkat Keras dan Amati Keberlanjutan KPIs

SUS03-BP03 Optimalkan area kode yang paling banyak menghabiskan waktu atau sumber daya

Optimalkan kode Anda yang dijalankan di dalam berbagai macam komponen arsitektur Anda untuk meminimalkan penggunaan sumber daya sambil memaksimalkan performa.

Anti-pola umum:

- Anda mengabaikan optimalisasi kode Anda untuk penggunaan sumber daya.
- Anda biasanya merespons masalah-masalah performa dengan meningkatkan sumber daya.

- Proses pengembangan dan peninjauan kode Anda tidak melacak perubahan-perubahan performa.

Manfaat menerapkan praktik terbaik ini: Menggunakan kode yang efisien dapat meminimalkan penggunaan sumber daya dan meningkatkan kinerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Setiap area fungsional harus diperiksa, termasuk kode untuk aplikasi dengan arsitektur cloud, untuk mengoptimalkan penggunaan sumber daya dan performanya. Lakukan pemantauan terhadap performa beban kerja Anda secara terus menerus di lingkungan pembangunan dan produksi dan identifikasi peluang-peluang untuk meningkatkan snippet kode yang memiliki penggunaan sumber daya sangat tinggi. Adopsi proses peninjauan secara teratur untuk mengidentifikasi bug atau anti-pola yang ada di dalam kode Anda yang menggunakan sumber daya secara tidak efisien. Manfaatkan algoritme sederhana dan efisien yang memberikan hasil yang sama untuk kasus penggunaan Anda.

Langkah-langkah implementasi

- Gunakan bahasa pemrograman yang efisien: Gunakan sistem operasi dan bahasa pemrograman yang efisien untuk beban kerja. Untuk informasi mendetail tentang bahasa pemrograman yang hemat energi (termasuk Rust), lihat Keberlanjutan dengan Rust.
- Gunakan pendamping pengkodean AI: Pertimbangkan untuk menggunakan pendamping pengkodean AI seperti Amazon CodeWhisperer untuk menulis kode secara efisien.
- Lakukan otomatisasi peninjauan kode: Saat mengembangkan beban kerja Anda, adopsi proses peninjauan kode otomatis untuk meningkatkan kualitas dan mengidentifikasi bug dan antipola.
  - Otomatiskan ulasan kode dengan Amazon CodeGuru Reviewer
  - Mendeteksi bug konkurensi dengan Amazon CodeGuru
  - Meningkatkan kualitas kode untuk aplikasi Python menggunakan Amazon CodeGuru
- Gunakan profiler kode: Gunakan profiler kode untuk mengidentifikasi area kode yang menggunakan waktu atau sumber daya paling banyak sebagai target optimasi.
  - Mengurangi jejak karbon organisasi Anda dengan Amazon Profiler CodeGuru
  - Memahami penggunaan memori di aplikasi Java Anda dengan Amazon CodeGuru Profiler
  - Meningkatkan pengalaman pelanggan dan mengurangi biaya dengan Amazon CodeGuru Profiler

- Pantau dan optimalkan: Gunakan sumber daya pemantauan berkelanjutan untuk mengidentifikasi komponen dengan persyaratan sumber daya yang tinggi atau konfigurasi mendekati optimal.
  - Ganti algoritme yang banyak memerlukan komputasi dengan versi algoritme yang lebih sederhana dan lebih efisien, yang akan memberikan hasil yang sama.
  - Singkirkan kode yang tidak perlu, seperti kode penyortiran dan pemformatan.
- Gunakan pemfaktoran ulang kode atau transformasi: Jelajahi kemungkinan transformasi kode Amazon Q untuk pemeliharaan dan peningkatan aplikasi.
  - Tingkatkan versi bahasa dengan Transformasi Kode Amazon Q
  - AWS re:invent 2023 - Otomatiskan peningkatan & pemeliharaan aplikasi menggunakan Transformasi Kode Q Amazon

Sumber daya

Dokumen terkait:

- Apa itu Amazon CodeGuru Profiler?
- FPGAcontoh
- Alat AWS SDKs on untuk Dibangun AWS

Video terkait:

- Tingkatkan Efisiensi Kode Menggunakan Amazon CodeGuru Profiler
- AWS Re:invent 2023 - Praktik terbaik untuk Amazon CodeWhisperer
- Otomatiskan Ulasan Kode dan Rekomendasi Kinerja Aplikasi dengan Amazon CodeGuru

Contoh terkait:

- Mengoptimalkan Kode dengan Amazon CodeGuru

SUS03-BP04 Optimalkan dampak pada perangkat dan peralatan

Pahami perangkat dan perlengkapan yang digunakan dalam arsitektur Anda dan gunakan strategi untuk mengurangi penggunaannya. Tindakan ini dapat meminimalkan dampak beban kerja cloud Anda pada lingkungan secara keseluruhan.

Anti-pola umum:

- Anda mengabaikan dampak yang ditimbulkan oleh perangkat yang digunakan oleh pelanggan Anda terhadap lingkungan.
- Anda mengelola dan memperbarui sumber daya yang digunakan oleh pelanggan secara manual.

Manfaat menerapkan praktik terbaik ini: Menerapkan pola dan fitur perangkat lunak yang sudah dioptimalkan untuk perangkat pelanggan dapat mengurangi dampak lingkungan yang ditimbulkan beban kerja cloud Anda secara keseluruhan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Mengimplementasikan fitur dan pola perangkat lunak yang dioptimalkan untuk perangkat pelanggan dapat mengurangi dampaknya terhadap lingkungan dengan beberapa cara:

- Mengimplementasikan fitur baru yang kompatibel dengan versi lama dapat mengurangi jumlah penggantian perangkat keras.
- Mengoptimalkan aplikasi untuk beroperasi secara efisien di perangkat dapat membantu Anda mengurangi pemakaian energi dan memperpanjang masa pakai baterai (jika menggunakan tenaga baterai).
- Mengoptimalkan aplikasi untuk perangkat dapat juga mengurangi transfer data lewat jaringan.

Pahami perangkat dan perlengkapan yang digunakan dalam arsitektur Anda, siklus hidupnya yang diharapkan, dan dampak dari penggantian komponen-komponen tersebut. Implementasikan fitur dan pola perangkat lunak yang dapat membantu Anda meminimalkan pemakaian energi perangkat, keharusan pelanggan untuk mengganti perangkat dan melakukan pemutakhiran perangkat secara manual.

Langkah-langkah implementasi

- Lakukan inventarisasi: Buatlah inventarisasi perangkat yang digunakan dalam arsitektur Anda. Perangkat dapat berupa ponsel, tablet, IOT perangkat, lampu pintar, atau bahkan perangkat pintar di pabrik.
- Gunakan perangkat hemat energi: Pertimbangkan untuk menggunakan perangkat hemat energi dalam arsitektur Anda. Gunakan konfigurasi manajemen daya pada perangkat untuk masuk ke mode daya rendah saat tidak digunakan.
- Jalankan aplikasi yang efisien: Optimalkan aplikasi yang berjalan di perangkat:

  - Gunakan strategi seperti menjalankan tugas di latar belakang untuk mengurangi pemakaian energi.
  - Perhitungkan bandwidth jaringan dan latensi saat membangun payload, dan implementasikan kemampuan yang dapat membantu aplikasi bekerja dengan baik pada tautan yang memiliki bandwidth rendah dan latensi tinggi.
  - Ubah format payload dan file ke format optimal yang diperlukan oleh perangkat. Misalnya, Anda dapat menggunakan Amazon Elastic Transcoder atau AWS Elemental MediaConvert untuk mengonversi file media digital yang besar dan berkualitas tinggi ke dalam format yang dapat diputar ulang pengguna di perangkat seluler, tablet, browser web, dan televisi yang terhubung.
  - Lakukan aktivitas yang membutuhkan banyak komputasi di sisi server (seperti melakukan rendering gambar), atau gunakan streaming aplikasi untuk meningkatkan pengalaman pengguna pada perangkat yang lebih lama.
  - Segmentasikan dan beri nomor halaman pada output, terutama untuk sesi-sesi interaktif, guna mengelola payload dan membatasi persyaratan penyimpanan lokal.
- Libatkan pemasok: Bekerja samalah dengan pemasok perangkat yang menggunakan bahan berkelanjutan dan memberikan transparansi dalam rantai pasokan dan sertifikasi lingkungan mereka.
- Use over-the-air (OTA) update: Gunakan mekanisme automated over-the-air (OTA) untuk menyebarkan pembaruan ke satu atau beberapa perangkat.
  - Anda dapat menggunakan sebuah pipeline CI/CD untuk memperbarui aplikasi seluler.
  - Anda dapat menggunakan AWS IoT Device Management untuk mengelola perangkat yang terhubung dari jarak jauh dalam skala besar.
- Gunakan device farm terkelola: Untuk menguji fitur baru dan pembaruan, gunakan device farm terkelola dengan set perangkat keras representatif dan ulang pengembangan untuk memaksimalkan perangkat yang didukung. Untuk detail selengkapnya, lihat SUS06-BP04 Gunakan peternakan perangkat terkelola untuk pengujian.
- Pemantau dan peningkatan terus-menerus: Lacak penggunaan energi perangkat untuk mengidentifikasi area-area yang bisa diperbaiki. Gunakan teknologi atau praktik terbaik terbaru untuk memperbaiki dampak lingkungan yang ditimbulkan oleh perangkat-perangkat tersebut.

Sumber daya

Dokumen terkait:

- Apa itu AWS Device Farm?

- AppStream 2.0 Dokumentasi
- NICE DCV
- OTAtutorial untuk memperbarui firmware pada perangkat yang berjalan Gratis RTOS
- Mengoptimalkan Perangkat IoT Anda untuk Kelestarian Lingkungan

Video terkait:

- AWS re:invent 2023 - Tingkatkan kualitas aplikasi seluler dan web Anda menggunakan AWS Device Farm

SUS03-BP05 Gunakan pola dan arsitektur perangkat lunak yang paling mendukung akses data dan pola penyimpanan

Pahami bagaimana data digunakan di dalam beban kerja Anda, dipakai oleh pengguna Anda, ditransfer, dan disimpan. Gunakan pola perangkat lunak dan arsitektur yang paling mendukung akses dan penyimpanan data untuk meminimalkan sumber daya komputasi, jaringan, dan penyimpanan yang diperlukan untuk mendukung beban kerja.

Anti-pola umum:

- Anda berasumsi bahwa semua beban kerja memiliki pola penyimpanan data dan akses data yang serupa.
- Anda hanya menggunakan satu tingkat penyimpanan, dengan anggapan semua beban kerja masuk dalam tingkat tersebut.
- Anda berasumsi bahwa pola akses data tidak akan berubah.
- Arsitektur Anda mendukung potensi lonjakan akses data yang tinggi, yang dapat mengakibatkan sumber daya tetap menjadi tidak aktif dalam sebagian besar waktu.

Manfaat menerapkan praktik terbaik ini: Memilih dan mengoptimalkan arsitektur Anda berdasarkan pola akses data dan pola penyimpanan data akan membantu Anda untuk mengurangi kompleksitas pengembangan dan meningkatkan pemanfaatan secara keseluruhan. Memahami kapan harus menggunakan tabel global, partisi data, dan caching akan membantu Anda untuk mengurangi biaya operasional dan menskalakan sesuai kebutuhan beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Gunakan pola-pola arsitektur dan perangkat lunak yang paling sesuai dengan karakteristik data dan pola akses Anda. Misalnya, gunakan arsitektur data modern di AWS yang memungkinkan Anda untuk menggunakan layanan yang dibuat khusus yang dioptimalkan untuk kasus penggunaan analitik unik Anda. Pola-pola arsitektur ini akan memungkinkan pemrosesan data yang efisien dan mengurangi penggunaan sumber daya.

Langkah-langkah implementasi

- Lakukan analisis terhadap karakteristik data dan pola akses Anda untuk mengidentifikasi konfigurasi yang tepat untuk sumber daya cloud Anda. Karakteristik utama yang perlu dipertimbangkan antara lain:
  - Jenis data: terstruktur, semi-terstruktur, tidak terstruktur
  - Pertumbuhan data: terbatas, tidak terbatas
  - Ketahanan data: persisten, sementara, transien
  - Pola akses baca atau tulis, frekuensi pembaruan, berfluktuasi, atau konsisten
- Gunakan pola-pola arsitektur yang paling mendukung pola akses dan penyimpanan data.
  - Pola untuk memungkinkan persistensi data
  - Mari Merancang! Arsitektur data modern
  - Database pada AWS: Alat yang Tepat untuk Pekerjaan yang Tepat
- Gunakan teknologi yang berfungsi secara native dengan data terkompresi.
  - Format file Dukungan Kompresi Athena
  - Opsi Format untuk ETL Input dan Output di AWS Glue
  - Memuat file data terkompresi dari Amazon S3 dengan Amazon Redshift
- Gunakan layanan analitik yang dibuat khusus untuk pemrosesan data dalam arsitektur Anda. Untuk detail tentang layanan analitik AWS yang dibuat khusus, lihat AWS re:Invent 2022 - Membangun arsitektur data modern di. AWS
- Gunakan mesin basis data yang paling mendukung pola-pola kueri dominan Anda. Kelola indeks basis data Anda untuk memastikan pembuatan kueri yang efisien. Untuk detail lebih lanjut, lihat Basis Data AWS dan AWS re:Invent 2022 - - Melakukan modernisasi aplikasi dengan basis data yang dibuat khusus.
- Pilihlah protokol-protokol jaringan yang dapat mengurangi jumlah kapasitas jaringan yang dipakai di arsitektur Anda.

Sumber daya

Dokumen terkait:

- COPYdari format data kolumnar dengan Amazon Redshift
- Mengonversi Format Catatan Input Anda di Firehose
- Meningkatkan kinerja kueri di Amazon Athena dengan Mengonversi ke Format Kolom
- Memantau muatan DB dengan Wawasan Performa di Amazon Aurora
- Memantau beban DB dengan Performance Insights di Amazon RDS
- Kelas penyimpanan Amazon S3 Intelligent-Tiering
- Bangun toko CQRS acara dengan Amazon DynamoDB

Video terkait:

- AWS re:invent 2022 - Membangun arsitektur data mesh di AWS
- AWS RE: invent 2023 - Selami Amazon Aurora dan inovasinya
- AWS RE: invent 2023 - Meningkatkan EBS efisiensi Amazon dan menjadi lebih hemat biaya
- AWS Re: invent 2023 - Mengoptimalkan harga dan kinerja penyimpanan dengan Amazon S3
- AWS re:invent 2023 - Membangun dan mengoptimalkan data lake di Amazon S3
- AWS Re:invent 2023 - Pola berbasis peristiwa tingkat lanjut dengan Amazon EventBridge

Contoh terkait:

- AWS Workshop Database yang Dibangun Tujuan
- AWS Hari Perendaman Arsitektur Data Modern
- Membangun Data Mesh di AWS

## Data

Pertanyaan

- SUS4 Bagaimana Anda memanfaatkan kebijakan dan pola manajemen data untuk mendukung tujuan keberlanjutan Anda?

## SUS4 Bagaimana Anda memanfaatkan kebijakan dan pola manajemen data untuk mendukung tujuan keberlanjutan Anda?

Implementasikan praktik manajemen data untuk mengurangi penyediaan penyimpanan yang diperlukan untuk mendukung beban kerja Anda, serta untuk mengurangi sumber daya yang diperlukan untuk menggunakannya. Pahami data Anda, dan gunakan konfigurasi dan teknologi penyimpanan penggunaan yang paling efektif untuk mendukung nilai bisnis data dan cara data digunakan. Buat siklus hidup data agar penyimpanan menjadi lebih efisien dan memiliki kinerja lebih rendah ketika persyaratan berkurang, dan hapus data yang tidak lagi diperlukan.

Praktik terbaik

- SUS04-BP01 Menerapkan kebijakan klasifikasi data
- SUS04-BP02 Menggunakan teknologi yang mendukung akses data dan pola penyimpanan
- SUS04-BP03 Gunakan kebijakan untuk mengelola siklus hidup kumpulan data Anda
- SUS04-BP04 Gunakan elastisitas dan otomatisasi untuk memperluas penyimpanan blok atau sistem file
- SUS04-BP05 Hapus data yang tidak dibutuhkan atau berlebihan
- SUS04-BP06 Gunakan sistem file bersama atau penyimpanan untuk mengakses data umum
- SUS04-BP07 Meminimalkan pergerakan data di seluruh jaringan
- SUS04-BP08 Cadangkan data hanya ketika sulit dibuat ulang

### SUS04-BP01 Menerapkan kebijakan klasifikasi data

Kelompokkan data untuk memahami tingkat kekritisannya terhadap hasil bisnis dan pilih tingkat penyimpanan hemat energi yang tepat untuk menyimpan data.

Anti-pola umum:

- Anda tidak mengidentifikasi aset data yang memiliki karakteristik serupa (seperti sensitivitas, kekritisan bisnis, atau persyaratan peraturan) yang diproses atau disimpan.
- Anda belum mengimplementasikan katalog data untuk menginventarisasi aset data Anda.

Manfaat menerapkan praktik terbaik ini: Menerapkan kebijakan klasifikasi data memungkinkan Anda menentukan tingkat penyimpanan data yang paling hemat energi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Klasifikasi data melibatkan identifikasi jenis-jenis data yang sedang diproses dan disimpan dalam sistem informasi yang dimiliki atau dioperasikan oleh organisasi. Klasifikasi data juga melibatkan penentuan tingkat kekritisan data dan kemungkinan dampak-dampak yang ditimbulkan saat terjadi peretasan, kehilangan, atau penyalahgunaan data.

Implementasikan kebijakan klasifikasi data dengan bekerja mundur dari penggunaan data kontekstual dan membuat sebuah skema kategorisasi yang mempertimbangkan tingkat kekritisan set data tertentu terhadap operasi organisasi.

Langkah-langkah implementasi

- Lakukan inventarisasi data: Lakukan inventarisasi atas berbagai jenis data yang ada untuk beban kerja Anda.
- Kelompokkan data: Tentukan kekritisan, kerahasiaan, integritas, dan ketersediaan data berdasarkan risiko terhadap organisasi. Gunakan persyaratan-persyaratan ini untuk mengelompokkan data ke dalam satu tingkat klasifikasi data yang Anda adopsi. Sebagai contoh, lihat Empat langkah sederhana untuk mengklasifikasikan data Anda dan mengamankan perusahaan rintisan Anda.
- Tentukan tingkat dan kebijakan klasifikasi data: Untuk masing-masing kelompok data, tentukan tingkat klasifikasi datanya (misalnya, publik atau rahasia) dan kebijakan penanganannya. Berikan tag pada data sebagaimana mestinya. Untuk detail selengkapnya tentang kategori data, lihat laporan resmi Klasifikasi Data.
- Tinjau secara berkala: Lakukan peninjauan dan audit terhadap lingkungan Anda secara berkala untuk data yang tidak diberi tag dan tidak diklasifikasikan. Gunakan otomatisasi untuk mengidentifikasi data ini, dan klasifikasikan serta berikan tag pada data dengan semestinya. Sebagai contoh, lihat Katalog Data dan perayap di AWS Glue.
- Buat katalog data: Buatlah katalog data yang menyediakan kemampuan audit dan tata kelola.
- Dokumentasi: Buatlah dokumentasi kebijakan klasifikasi data dan prosedur penanganan untuk setiap kelas data.

Sumber daya

Dokumen terkait:

- Memanfaatkan AWS Cloud untuk Mendukung Klasifikasi Data

- Tag kebijakan dari AWS Organizations

Video terkait:

- AWS re:invent 2022 - Mengaktifkan kelincahan dengan tata kelola data di AWS
- AWS RE: invent 2023 - Perlindungan dan ketahanan data dengan penyimpanan AWS

SUS04-BP02 Menggunakan teknologi yang mendukung akses data dan pola penyimpanan

Gunakan teknologi penyimpanan yang paling mendukung cara data Anda diakses dan disimpan untuk meminimalkan sumber daya yang disediakan sambil mendukung beban kerja Anda.

Anti-pola umum:

- Anda berasumsi bahwa semua beban kerja memiliki pola penyimpanan data dan akses data yang serupa.
- Anda hanya menggunakan satu tingkat penyimpanan, dengan anggapan semua beban kerja masuk dalam tingkat tersebut.
- Anda berasumsi bahwa pola akses data tidak akan berubah.

Manfaat menerapkan praktik terbaik ini: Memilih dan mengoptimalkan teknologi penyimpanan Anda berdasarkan pola akses dan penyimpanan data akan membantu Anda mengurangi sumber daya cloud yang diperlukan untuk memenuhi kebutuhan bisnis dan meningkatkan keseluruhan efisiensi beban kerja cloud.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Pilih solusi penyimpanan yang paling sesuai dengan pola akses Anda, atau pertimbangkan untuk mengubah pola akses tersebut agar sesuai dengan solusi penyimpanan untuk meraih efisiensi kinerja yang maksimal.

Langkah-langkah implementasi

- Lakukan evaluasi terhadap karakteristik data dan akses: Evaluasi karakteristik dan pola akses data Anda untuk mengumpulkan karakteristik utama kebutuhan penyimpanan Anda. Karakteristik utama yang perlu dipertimbangkan antara lain:

- Jenis data: terstruktur, semi-terstruktur, tidak terstruktur
- Pertumbuhan data: terbatas, tidak terbatas
- Ketahanan data: persisten, sementara, transien
- Pola akses baca atau tulis, frekuensi, berfluktuasi, atau konsisten

- Pilih teknologi penyimpanan yang tepat: Migrasikan data ke teknologi penyimpanan yang tepat yang mendukung karakteristik dan pola akses data Anda. Berikut adalah beberapa contoh teknologi AWS penyimpanan dan karakteristik utamanya:

| Tipe | Teknologi | Karakteristik utama |
| --- | --- | --- |
| Penyimpanan objek | Amazon S3 | Sebuah layanan penyimpan an objek yang memiliki skalabilitas tak terbatas, ketersediaan tinggi, dan berbagai opsi aksesibilitas. Mentransfer dan mengakses objek-objek yang masuk dan keluar dari Amazon S3 dapat dilakukan dengan menggunak an sebuah layanan, seperti Akselerasi Transfer atau Titik Akses, untuk mendukung lokasi, kebutuhan keamanan, dan pola akses Anda. |
| Penyimpanan pengarsipan | Amazon S3 Glacier | Kelas penyimpanan Amazon S3 yang dibuat untuk pengarsipan data. |
| Sistem file bersama | Amazon Elastic File System (AmazonEFS) | Sistem file yang dapat dipasang dan dapat diakses oleh berbagai jenis solusi komputasi. Amazon EFS secara otomatis menumbuhk an dan menyusutkan penyimpanan dan dioptimal |

| Tipe | Teknologi | Karakteristik utama |
| --- | --- | --- |
| | | kan kinerja untuk menghasil kan latensi rendah yang konsisten. |
| Sistem file bersama | Amazon FSx | Dibangun di atas solusi AWS komputasi terbaru untuk mendukung empat sistem file yang umum digunakan: NetApp ONTAP, OpenZFS, Windows File Server, dan Lustre. FSxLatensi Amazon, throughput, dan IOPS bervariasi per sistem file dan harus dipertimbangkan saat memilih sistem file yang tepat untuk kebutuhan beban kerja Anda. |
| Penyimpanan blok | Toko Blok Elastis Amazon (AmazonEBS) | Layanan penyimpanan blok berkinerja tinggi yang dapat diskalakan yang dirancang untuk Amazon Elastic Compute Cloud (Amazon). EC2 Amazon EBS menyertak an penyimpanan yang SSD didukung untuk beban kerja transaksional dan intensif serta HDD penyimpanan yang IOPS didukung untuk beban kerja yang intensif throughput. |

| Tipe | Teknologi | Karakteristik utama |
| --- | --- | --- |
| Basis data relasional | Amazon Aurora, AmazonRDS, Amazon Redshift | Dirancang untuk mendukung transaksi ACID (atomisitas, konsistensi, isolasi, daya tahan) dan menjaga integritas referensial dan konsistensi data yang kuat. Banyak aplikasi tradisional, perencanaan sumber daya perusahaan (ERP), manajemen hubungan pelanggan (CRM), dan sistem e-commerce menggunakan database relasional untuk menyimpan data mereka. |
| Basis data nilai-kunci | Amazon DynamoDB | Dioptimalkan untuk pola akses umum, biasanya digunakan untuk menyimpan dan mengambil data dalam volume besar. Aplikasi web dengan lalu lintas tinggi, sistem perdagangan elektronik, dan aplikasi gaming merupakan kasus penggunaan umum untuk basis data nilai kunci. |

- Mengotomatiskan alokasi penyimpanan: Untuk sistem penyimpanan yang berukuran tetap, seperti Amazon atau EBS AmazonFSx, pantau ruang penyimpanan yang tersedia dan otomatiskan alokasi penyimpanan saat mencapai ambang batas. Anda dapat memanfaatkan Amazon CloudWatch untuk mengumpulkan dan menganalisis berbagai metrik untuk Amazon EBS dan Amazon FSx.
- Pilih kelas penyimpanan yang tepat: Pilihlah kelas penyimpanan yang sesuai untuk data Anda.
  - Kelas penyimpanan Amazon S3 dapat dikonfigurasi pada tingkat objek. Satu bucket dapat berisi objek-objek yang disimpan di semua kelas penyimpanan.

- Anda dapat menggunakan kebijakan Siklus Hidup Amazon S3 untuk mengalihkan objek secara otomatis antar kelas-kelas penyimpanan atau menghapus data tanpa perubahan aplikasi apa pun. Secara umum, Anda harus memilih mana yang paling penting (melakukan kompromi) antara efisiensi sumber daya, latensi akses, dan keandalan saat mempertimbangkan semua mekanisme penyimpanan ini.

Sumber daya

Dokumen terkait:

- Jenis EBS volume Amazon
- Toko EC2 contoh Amazon
- Amazon S3 Intelligent-Tiering
- Karakteristik Amazon EBS I/O
- Menggunakan kelas penyimpanan Amazon S3
- Apa Itu Amazon S3 Glacier?

Video terkait:

- AWS RE: invent 2023 - Meningkatkan EBS efisiensi Amazon dan menjadi lebih hemat biaya
- AWS Re: invent 2023 - Mengoptimalkan harga dan kinerja penyimpanan dengan Amazon S3
- AWS re:invent 2023 - Membangun dan mengoptimalkan data lake di Amazon S3
- AWS re:invent 2022 - Membangun arsitektur data modern di AWS
- AWS re:invent 2022 - Modernisasi aplikasi dengan database yang dibuat khusus
- AWS re:invent 2022 - Membangun arsitektur data mesh di AWS
- AWS re: invent 2023 - Menyelam jauh ke Amazon Aurora dan inovasinya
- AWS RE: invent 2023 - Pemodelan data tingkat lanjut dengan Amazon DynamoDB

Contoh terkait:

- Contoh-contoh Amazon S3
- AWS Workshop Database yang Dibangun Tujuan
- Basis Data untuk Pengembang
- AWS Hari Perendaman Arsitektur Data Modern

- Membangun Data Mesh di AWS

SUS04-BP03 Gunakan kebijakan untuk mengelola siklus hidup kumpulan data Anda

Kelola siklus hidup semua data Anda dan terapkan penghapusan secara otomatis untuk meminimalkan total penyimpanan yang diperlukan untuk beban kerja Anda.

Anti-pola umum:

- Anda menghapus data secara manual.
- Anda tidak menghapus data beban kerja Anda sama sekali.
- Anda tidak mengalihkan data ke tingkat penyimpanan yang lebih hemat energi berdasarkan persyaratan retensi dan aksesnya.

Manfaat menerapkan praktik terbaik ini: Menggunakan kebijakan siklus hidup data memastikan akses dan retensi data yang efisien dalam beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Set data biasanya memiliki persyaratan retensi dan akses data yang berbeda-beda selama masa hidupnya. Misalnya, aplikasi Anda mungkin memerlukan akses yang sering ke sejumlah set data dalam jangka waktu terbatas. Setelah itu, set-set data tersebut jarang diakses.

Untuk mengelola set data Anda secara efisien di sepanjang siklus hidupnya, konfigurasikan kebijakan siklus hidup, yakni aturan yang menetapkan cara menangani set data.

Dengan Aturan konfigurasi siklus hidup, Anda dapat meminta layanan penyimpanan tertentu untuk mengalihkan suatu set data ke tingkat penyimpanan yang lebih hemat energi, mengarsipkannya, atau menghapusnya.

Langkah-langkah implementasi

- Klasifikasikan set data dalam beban kerja Anda.
- Tetapkan prosedur penanganan untuk setiap kelas data.
- Atur kebijakan siklus hidup otomatis untuk menegakkan aturan siklus hidup. Berikut adalah beberapa contoh cara mengatur kebijakan siklus hidup otomatis untuk berbagai layanan AWS penyimpanan:

| Layanan penyimpanan | Cara menyetel kebijakan siklus hidup otomatis |
| --- | --- |
| Amazon S3 | Anda dapat menggunakan Siklus Hidup Amazon S3 untuk mengelola objek-objek Anda di sepanjang siklus hidupnya. Jika pola akses Anda tidak diketahui, berubah, atau tidak dapat diprediksi, Anda dapat menggunak an Amazon S3 Intelligent-Tiering, yang akan memantau pola akses dan secara otomatis memindahkan objek yang belum diakses ke tingkat akses yang berbiaya lebih rendah. Anda dapat memanfaatkan metrik Lensa Penyimpanan Amazon S3 untuk melakukan identifikasi terhadap peluang dan celah pengoptimalan dalam manajemen siklus hidup. |
| Amazon Elastic Block Store | Anda dapat menggunakan Amazon Data Lifecycle Manager untuk mengotomatiskan pembuatan, penyimpanan, dan penghapus an snapshot Amazon dan Amazon yang didukung. EBS EBS AMIs |
| Sistem File Elastis Amazon | Manajemen EFS siklus hidup Amazon secara otomatis mengelola penyimpanan file untuk sistem file Anda. |
| Amazon Elastic Container Registry | Kebijakan ECR siklus hidup Amazon mengotomatiskan pembersihan gambar kontainer Anda dengan kedaluwarsa gambar berdasarkan usia atau hitungan. |
| AWS Elemental MediaStore | Anda dapat menggunakan kebijakan siklus hidup objek yang mengatur berapa lama objek harus disimpan dalam wadah. MediaStore |

- Hapus volume, snapshot, dan data yang tidak digunakan yang sudah melebihi masa retensinya. Manfaatkan fitur layanan asli seperti Amazon DynamoDB Time To Live atau retensi log CloudWatch Amazon untuk dihapus.
- Lakukan agregasi dan kompresi data, jika memungkinkan, berdasarkan aturan siklus hidup.

Sumber daya

Dokumen terkait:

- Optimalkan aturan Siklus Hidup Amazon S3 Anda dengan Analisis Kelas Penyimpanan Amazon S3
- Mengevaluasi Sumber Daya dengan Aturan AWS Config

Video terkait:

- AWS re:invent 2021 - Praktik terbaik Siklus Hidup Amazon S3 untuk mengoptimalkan pengeluaran penyimpanan Anda
- AWS Re: invent 2023 - Mengoptimalkan harga dan kinerja penyimpanan dengan Amazon S3
- Sederhanakan Siklus Hidup Data Anda dan Optimalkan Biaya Penyimpanan Dengan Siklus Hidup Amazon S3
- Menggunakan Lensa Penyimpanan Amazon S3 untuk Mengurangi Biaya Penyimpanan Anda

SUS04-BP04 Gunakan elastisitas dan otomatisasi untuk memperluas penyimpanan blok atau sistem file

Gunakan elastisitas dan otomatisasi untuk memperluas sistem file atau penyimpanan blok seiring pertumbuhan data untuk meminimalkan total penyimpanan yang disediakan.

Anti-pola umum:

- Anda membeli sistem file atau penyimpanan blok besar untuk keperluan di waktu mendatang.
- Anda menyediakan lebih banyak operasi input dan output per detik (IOPS) dari sistem file Anda.
- Anda tidak memantau pemanfaatan volume data Anda.

Manfaat menerapkan praktik terbaik ini: Meminimalkan penyediaan berlebih untuk sistem penyimpanan akan mengurangi sumber daya yang tidak digunakan dan meningkatkan efisiensi keseluruhan beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Buat sistem file dan penyimpanan blok dengan alokasi ukuran, throughput, dan latensi yang sesuai untuk beban kerja Anda. Gunakan elastisitas dan otomatisasi untuk memperluas sistem file atau penyimpanan blok seiring pertumbuhan data tanpa perlu melakukan penyediaan layanan penyimpanan yang berlebihan.

Langkah-langkah implementasi

- Untuk penyimpanan ukuran tetap seperti Amazon EBS, verifikasi bahwa Anda memantau jumlah penyimpanan yang digunakan versus ukuran penyimpanan keseluruhan dan buat otomatisasi, jika mungkin, untuk meningkatkan ukuran penyimpanan saat mencapai ambang batas.
- Gunakan volume elastis dan layanan data blok terkelola untuk mengotomatisasi alokasi penyimpanan tambahan seiring tumbuhnya data persisten Anda. Sebagai contoh, Anda dapat menggunakan Volume EBS Elastis Amazon untuk mengubah ukuran volume, jenis volume, atau menyesuaikan kinerja EBS volume Amazon Anda.
- Pilih kelas penyimpanan, mode performa, dan mode throughput yang tepat untuk sistem file Anda guna memenuhi kebutuhan bisnis, tidak melebihinya.
  - EFSKinerja Amazon
  - Kinerja EBS volume Amazon pada instance Linux
- Atur target tingkat pemanfaatan untuk volume data Anda, dan ubah ukuran volume di luar rentang yang diperkirakan.
- Sesuaikan ukuran volume hanya-baca agar sesuai dengan data.
- Migrasikan data ke penyimpanan objek untuk menghindari penyediaan kapasitas yang berlebihan dari ukuran volume tetap di penyimpanan blok.
- Secara rutin tinjau volume elastis dan sistem file untuk menghentikan volume yang tidak aktif dan memperkecil sumber daya dengan penyediaan berlebihan agar sesuai dengan ukuran data saat ini.

Sumber daya

Dokumen terkait:

- Perluas sistem file setelah mengubah ukuran volume EBS

- Ubah volume menggunakan Amazon EBS Elastic Volumes
- FSxDokumentasi Amazon
- Apa itu Sistem File Elastis Amazon?

Video terkait:

- Menyelam Jauh di Volume EBS Elastis Amazon
- Amazon EBS dan Strategi Optimasi Snapshot untuk Kinerja yang Lebih Baik dan Penghematan Biaya
- Mengoptimalkan Amazon EFS untuk biaya dan kinerja, menggunakan praktik terbaik

SUS04-BP05 Hapus data yang tidak dibutuhkan atau berlebihan

Hapus data yang tidak diperlukan atau redundan untuk meminimalkan sumber daya penyimpanan yang diperlukan untuk menyimpan set data Anda.

Anti-pola umum:

- Anda menduplikasi data yang dapat diperoleh atau dibuat ulang dengan mudah.
- Anda mencadangkan semua data tanpa mempertimbangkan tingkat kekritisannya.
- Anda menghapus data tidak rutin, hanya pada peristiwa operasional, atau tidak menghapusnya sama sekali.
- Anda menyimpan data secara redundan dengan mengabaikan durabilitas layanan penyimpanan.
- Anda mengaktifkan penentuan versi Amazon S3 tanpa alasan bisnis apa pun.

Manfaat menerapkan praktik terbaik ini: Menghapus data yang tidak dibutuhkan akan mengurangi ukuran penyimpanan yang diperlukan untuk beban kerja Anda dan dampak lingkungan yang ditimbulkan beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jangan menyimpan data yang tidak Anda perlukan. Otomatiskan penghapusan data yang tidak diperlukan. Gunakan teknologi yang menghilangkan data ganda pada tingkat file dan blok. Manfaatkan fitur replikasi dan redundansi data native dari layanan.

Langkah-langkah implementasi

- Evaluasi apakah Anda dapat menghindari penyimpanan data dengan menggunakan set data yang tersedia untuk umum yang ada di dalam AWS Data Exchange dan Data Terbuka di AWS.
- Gunakan mekanisme yang dapat membatalkan duplikasi data pada tingkat blok dan objek. Berikut adalah beberapa contoh cara menghapus duplikat data pada: AWS

| Layanan penyimpanan | Mekanisme deduplikasi |
| --- | --- |
| Amazon S3 | Gunakan AWS Lake Formation FindMatch esuntuk menemukan catatan yang cocok di seluruh kumpulan data (termasuk yang tanpa pengenal) dengan menggunakan Transform FindMatches ML baru. |
| Amazon FSx | Gunakan deduplikasi data di Amazon FSx untuk Windows. |
| Snapshot Amazon Elastic Block Store | Snapshot bertahap pencadangan, yang berarti bahwa hanya blok di perangkat yang diubah setelah snapshot terbaru Anda disimpan. |

- Analisis akses data untuk mengidentifikasi data yang tidak diperlukan. Otomatiskan kebijakan siklus hidup. Manfaatkan fitur layanan asli seperti Amazon DynamoDB Time To Live, Siklus Hidup Amazon S3, atau retensi log Amazon untuk dihapus. CloudWatch
- Gunakan kemampuan virtualisasi data AWS untuk mempertahankan data pada sumbernya dan menghindari duplikasi data.
  - Virtualisasi Data Asli Cloud di AWS
  - Optimalkan Pola Data menggunakan Pembagian Data Amazon Redshift
- Gunakan teknologi pencadangan yang dapat membuat cadangan bertahap.
- Manfaatkan daya tahan Amazon S3 dan replikasi Amazon EBS untuk memenuhi tujuan daya tahan Anda alih-alih teknologi yang dikelola sendiri (seperti susunan disk independen () yang berlebihan). RAID
- Pusatkan log dan lacak data, batalkan duplikasi entri log yang identik, dan buat mekanisme untuk menyesuaikan verbositas saat diperlukan.
- Pra-isi cache hanya saat ada alasan yang dibenarkan.

- Lakukan pemantauan dan otomatisasi cache untuk menyesuaikan ukuran cache dengan tepat.
- Hapus out-of-date penerapan dan aset dari penyimpanan objek dan cache tepi saat mendorong versi baru beban kerja Anda.

Sumber daya

Dokumen terkait:

- Ubah penyimpanan data log di CloudWatch Log
- Deduplikasi data di Amazon FSx untuk Windows File Server
- Fitur Amazon FSx untuk ONTAP menyertakan deduplikasi data
- Membatalkan File di Amazon CloudFront
- Menggunakan AWS Backup untuk mencadangkan dan memulihkan sistem EFS file Amazon
- Apa itu Amazon CloudWatch Logs?
- Bekerja dengan cadangan di Amazon RDS
- Integrasikan dan deduplikat kumpulan data menggunakan AWS Lake Formation

Video terkait:

- Kasus Penggunaan Berbagi Data Amazon Redshift

Contoh terkait:

- Bagaimana cara menganalisis log akses server Amazon S3 menggunakan Amazon Athena?

SUS04-BP06 Gunakan sistem file bersama atau penyimpanan untuk mengakses data umum

Adopsi sistem file atau penyimpanan bersama untuk menghindari duplikasi data dan memungkinkan infrastruktur yang lebih efisien untuk beban kerja Anda.

Anti-pola umum:

- Anda menyediakan penyimpanan untuk setiap klien secara individu.
- Anda tidak melepaskan volume data dari klien yang tidak aktif.
- Anda tidak memberikan akses ke penyimpanan di semua platform dan sistem.

Manfaat menerapkan praktik terbaik ini: Menggunakan sistem file atau penyimpanan bersama akan memungkinkan Anda untuk berbagi data ke satu atau beberapa konsumen tanpa harus menyalin data. Hal ini membantu mengurangi sumber daya penyimpanan yang diperlukan untuk beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jika Anda memiliki beberapa pengguna atau aplikasi yang mengakses set data yang sama, penggunaan teknologi penyimpanan bersama sangatlah penting untuk menggunakan infrastruktur yang efisien untuk beban kerja Anda. Teknologi penyimpanan bersama memberikan lokasi sentral untuk menyimpan dan mengelola set data dan menghindari duplikasi data. Teknologi ini juga memastikan konsistensi data di berbagai sistem yang berlainan. Lebih lanjut, teknologi penyimpanan bersama memungkinkan penggunaan daya komputasi yang lebih efisien, karena beberapa sumber daya komputasi dapat mengakses dan memproses data pada saat yang sama secara paralel.

Hanya ambil data dari layanan penyimpanan bersama ini sesuai kebutuhan dan lepaskan volume yang tidak digunakan untuk membebaskan sumber daya.

Langkah-langkah implementasi

- Migrasikan data ke penyimpanan bersama ketika data memiliki beberapa pemakai. Berikut adalah beberapa contoh teknologi penyimpanan bersama di AWS:

| Opsi penyimpanan | Kapan harus digunakan |
| --- | --- |
| Amazon EBS Multi-Lampirkan | Amazon EBS Multi-Attach memungkinkan Anda melampirkan satu volume Provisioned IOPS SSD (io1 atau io2) ke beberapa instance yang berada di Availability Zone yang sama. |
| Amazon EFS | Lihat Kapan Memilih Amazon EFS. |
| Amazon FSx | Lihat Memilih Sistem FSx File Amazon. |
| Amazon S3 | Aplikasi yang tidak memerlukan struktur sistem file dan didesain untuk berfungsi dengan penyimpanan objek dapat menggunakan Amazon S3 sebagai solusi penyimpanan |

| Opsi penyimpanan | Kapan harus digunakan |
| --- | --- |
| | objek yang dapat diskalakan besar-besaran, tahan lama, dan murah. |

- Salin data ke sistem file atau ambil data dari sistem file bersama hanya jika dibutuhkan. Sebagai contoh, Anda dapat membuat sistem file Amazon FSx untuk Lustre yang didukung oleh Amazon S3 dan hanya memuat subset data yang diperlukan untuk memproses pekerjaan ke Amazon. FSx
- Hapus data yang sesuai untuk pola penggunaan Anda seperti yang diuraikan dalam SUS04-BP03 Gunakan kebijakan untuk mengelola siklus hidup kumpulan data Anda.
- Lepaskan volume dari klien yang tidak menggunakannya secara aktif.

Sumber daya

Dokumen terkait:

- Menautkan sistem file Anda ke bucket Amazon S3
- Menggunakan Amazon EFS untuk AWS Lambda aplikasi tanpa server Anda
- Amazon EFS Intelligent-Tiering Mengoptimalkan Biaya untuk Beban Kerja dengan Mengubah Pola Akses
- Menggunakan Amazon FSx dengan repositori data lokal

video terkait:

- Optimalisasi biaya penyimpanan dengan Amazon EFS
- AWS re:invent 2023 - Apa yang baru dengan penyimpanan file AWS
- AWS RE: invent 2023 - Penyimpanan file untuk pembangun dan ilmuwan data di Amazon Elastic File System

SUS04-BP07 Meminimalkan pergerakan data di seluruh jaringan

Gunakan sistem file atau penyimpanan objek bersama untuk mengakses data umum dan meminimalkan total sumber daya jaringan yang diperlukan untuk mendukung perpindahan data beban kerja Anda.

Anti-pola umum:

- Anda menyimpan semua data di tempat yang sama Wilayah AWS terlepas dari tempat pengguna data berada.
- Anda tidak mengoptimalkan format dan ukuran data sebelum memindahkannya melalui jaringan.

Manfaat menerapkan praktik terbaik ini: Mengoptimalkan perpindahan data di seluruh jaringan mengurangi total sumber daya jaringan yang diperlukan untuk beban kerja dan memperkecil dampaknya pada lingkungan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Memindahkan data ke berbagai bagian dalam organisasi memerlukan sumber daya komputasi, jaringan, dan penyimpanan. Gunakan teknik untuk meminimalkan perpindahan data dan meningkatkan efisiensi beban kerja Anda secara keseluruhan.

Langkah-langkah implementasi

- Pertimbangkan kedekatan dengan data atau pengguna sebagai faktor dalam pengambilan keputusan saat memilih Wilayah untuk beban kerja Anda.
- Partisi layanan yang digunakan secara Regional sehingga data khusus Wilayahnya disimpan di Wilayah tempat penggunaan data.
- Gunakan format file yang efisien (seperti Parket atauORC) dan kompres data sebelum memindahkannya melalui jaringan.
- Jangan pindahkan data yang tidak digunakan. Beberapa contoh tindakan yang dapat membantu Anda menghindari pemindahan data yang tidak digunakan:
  - Kurangi API respons hanya pada data yang relevan.
  - Kumpulkan data apabila terperinci (informasi tingkat catatan tidak diperlukan).
  - Lihat Lab Well-Architected - Mengoptimalkan Pola Data Menggunakan Fitur Berbagi Data Amazon Redshift.
  - Pertimbangkan berbagi data lintas akun di AWS Lake Formation.
- Gunakan layanan yang dapat membantu Anda menjalankan kode lebih dekat dengan pengguna beban kerja Anda.

| Layanan | Kapan harus digunakan |
| --- | --- |
| Lambda@Edge | Gunakan untuk operasi dengan banyak komputasi yang dijalankan saat objek tidak ada dalam cache. |
| CloudFrontFungsi | Gunakan untuk kasus penggunaan sederhana seperti HTTP manipulasi permintaan/respons yang dapat dimulai oleh fungsi berumur pendek. |
| AWS IoT Greengrass | Jalankan komputasi lokal, olahpesan, dan caching data untuk perangkat yang terhubung. |

Sumber daya

Dokumen terkait:

- Mengoptimalkan AWS Infrastruktur Anda untuk Keberlanjutan, BagianIII: Jaringan
- AWS Infrastruktur Global
- Fitur CloudFront Utama Amazon termasuk CloudFront Global Edge Network
- Mengompresi HTTP permintaan di Layanan Amazon OpenSearch
- Kompresi data menengah dengan Amazon EMR
- Memuat file data terkompresi dari Amazon S3 ke Amazon Redshift
- Melayani file terkompresi dengan Amazon CloudFront

Video terkait:

- Mengungkap transfer data pada AWS

Contoh terkait:

- Merancang arsitektur untuk keberlanjutan - Meminimalkan pergerakan data lintas jaringan

SUS04-BP08 Cadangkan data hanya ketika sulit dibuat ulang

Hindari mencadangkan data yang tidak memiliki nilai bisnis untuk meminimalkan persyaratan sumber daya penyimpanan untuk beban kerja Anda.

Anti-pola umum:

- Anda tidak memiliki strategi cadangan untuk data Anda.
- Anda mencadangkan data yang dapat dibuat ulang dengan mudah.

Manfaat menerapkan praktik terbaik ini: Menghindari cadangan data non-kritis dapat mengurangi sumber daya penyimpanan yang diperlukan untuk beban kerja dan menurunkan dampak lingkungan yang ditimbulkannya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Menghindari pencadangan data yang tidak perlu dapat membantu menurunkan biaya dan mengurangi sumber daya penyimpanan yang digunakan oleh beban kerja. Hanya cadangkan data yang memiliki nilai bisnis atau yang diperlukan untuk memenuhi persyaratan kepatuhan. Periksa kebijakan pencadangan dan jangan sertakan penyimpanan sementara yang tidak memberikan nilai dalam skenario pemulihan.

Langkah-langkah implementasi

- Menerapkan kebijakan klasifikasi data sebagaimana diuraikan dalam SUS04-BP01 Menerapkan kebijakan klasifikasi data.
- Gunakan kekritisan klasifikasi data Anda dan rancang strategi cadangan berdasarkan tujuan waktu pemulihan Anda (RTO) dan tujuan titik pemulihan (RPO). Hindari mencadangkan data yang tidak penting.
  - Jangan sertakan data yang dapat dibuat ulang dengan mudah.
  - Jangan sertakan data sementara dari cadangan Anda.
  - Kecualikan salinan data lokal, kecuali waktu yang diperlukan untuk memulihkan data tersebut dari lokasi umum melebihi perjanjian tingkat layanan Anda ()SLAs.
- Gunakan solusi otomatis atau layanan terkelola untuk mencadangkan data yang penting bagi bisnis.

- AWS Backupadalah layanan yang dikelola sepenuhnya yang memudahkan untuk memusatkan dan mengotomatiskan perlindungan data di seluruh AWS layanan, di cloud, dan di tempat. Untuk panduan langsung tentang cara membuat backup otomatis menggunakan, AWS Backup lihat Well-Architected Labs - Testing Backup and Restore of Data.
- Otomatiskan pencadangan dan optimalkan biaya pencadangan untuk Amazon. EFS AWS Backup

Sumber daya

Praktik-praktik terbaik terkait:

- REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau mereproduksi data dari sumber
- REL09-BP03 Lakukan pencadangan data secara otomatis
- REL13-BP02 Gunakan strategi pemulihan yang ditentukan untuk memenuhi tujuan pemulihan

Dokumen terkait:

- Menggunakan AWS Backup untuk mencadangkan dan memulihkan sistem EFS file Amazon
- EBSCuplikan Amazon
- Bekerja dengan backup di Amazon Relational Database Service
- APNMitra: mitra yang dapat membantu dengan cadangan
- AWS Marketplace: produk yang dapat digunakan untuk pencadangan
- Mencadangkan Amazon EFS
- Mencadangkan Amazon FSx untuk Server File Windows
- Backup dan Restore untuk Amazon ElastiCache (RedisOSS)

Video terkait:

- AWS re:invent 2023 - Strategi backup dan pemulihan bencana untuk meningkatkan ketahanan
- AWS Re:invent 2023 - Apa yang baru AWS Backup
- AWS re:invent 2021 - Backup, pemulihan bencana, dan perlindungan ransomware dengan AWS

Contoh terkait:

- Lab Well-Architected - Data cadangan

## Perangkat keras dan layanan

Pertanyaan

- SUS5 Bagaimana Anda memilih dan menggunakan perangkat keras dan layanan cloud dalam arsitektur Anda untuk mendukung tujuan keberlanjutan Anda?

### SUS5 Bagaimana Anda memilih dan menggunakan perangkat keras dan layanan cloud dalam arsitektur Anda untuk mendukung tujuan keberlanjutan Anda?

Cari peluang untuk mengurangi dampak beban kerja terhadap keberlanjutan dengan membuat perubahan pada praktik manajemen perangkat keras Anda. Minimalkan jumlah perangkat keras yang perlu disediakan dan di-deploy, serta pilih perangkat keras dan layanan yang paling efisien untuk setiap beban kerja Anda.

Praktik terbaik

- SUS05-BP01 Gunakan jumlah minimum perangkat keras untuk memenuhi kebutuhan Anda
- SUS05-BP02 Gunakan tipe instans dengan dampak paling kecil
- SUS05-BP03 Gunakan layanan terkelola
- SUS05-BP04 Optimalkan penggunaan akselerator komputasi berbasis perangkat keras

SUS05-BP01 Gunakan jumlah minimum perangkat keras untuk memenuhi kebutuhan Anda

Gunakan perangkat keras dalam jumlah minimum untuk beban kerja Anda guna memenuhi kebutuhan-kebutuhan bisnis secara efisien.

Anti-pola umum:

- Anda tidak memantau pemanfaatan sumber daya.
- Anda memiliki sumber daya dengan tingkat pemanfaatan rendah di arsitektur Anda.
- Anda tidak meninjau pemanfaatan perangkat keras statis untuk menentukan apakah ukurannya harus diubah atau tidak.
- Anda tidak menetapkan tujuan pemanfaatan perangkat keras untuk infrastruktur komputasi Anda berdasarkan bisnis. KPIs

Manfaat menerapkan praktik terbaik ini: Mengatur ukuran sumber daya cloud Anda dengan tepat akan membantu mengurangi dampak lingkungan beban kerja, menghemat uang, dan mempertahankan tolok ukur kinerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Pilih secara optimal jumlah total perangkat keras yang diperlukan untuk beban kerja Anda guna meningkatkan efisiensi beban kerja secara keseluruhan. AWS Cloud Ini memberikan fleksibilitas untuk memperluas atau mengurangi jumlah sumber daya secara dinamis melalui berbagai mekanisme, seperti AWS Auto Scaling, dan memenuhi perubahan permintaan. Ini juga menyediakan APIsdan SDKs yang memungkinkan sumber daya untuk dimodifikasi dengan sedikit usaha. Gunakan kemampuan ini untuk membuat perubahan pada implementasi beban kerja Anda dengan interval waktu yang rapat. Selain itu, gunakan pedoman rightsizing dari AWS alat untuk mengoperasikan sumber daya cloud Anda secara efisien dan memenuhi kebutuhan bisnis Anda.

Langkah-langkah implementasi

- Pilih jenis instans: Pilih jenis instans yang tepat yang paling sesuai dengan kebutuhan Anda. Untuk mempelajari cara memilih instans Amazon Elastic Compute Cloud dan menggunakan mekanisme seperti pemilihan instans berbasis atribut, lihat dokumentasi berikut ini:
  - Bagaimana cara memilih jenis EC2 instans Amazon yang sesuai untuk beban kerja saya?
  - Pemilihan jenis instans berbasis atribut untuk Amazon EC2 Fleet.
  - Membuat grup Auto Scaling menggunakan pemilihan jenis instans berbasis atribut.
- Skalakan: Gunakan peningkatan kecil untuk menskalakan beban kerja variabel.
- Gunakan beberapa opsi pembelian komputasi: Seimbangkan fleksibilitas instans, skalabilitas, dan penghematan biayanya dengan menggunakan beberapa opsi pembelian komputasi.
  - Instans EC2 On-Demand Amazon paling cocok untuk beban kerja baru, stateful, dan runcing yang tidak dapat berupa jenis instans, lokasi, atau waktu yang fleksibel.
  - Amazon EC2 Spot Instances adalah cara yang bagus untuk melengkapi opsi lain untuk aplikasi yang toleran terhadap kesalahan dan fleksibel.
  - Manfaatkan Compute Savings Plans untuk beban kerja steady state yang memungkinkan fleksibilitas jika kebutuhan Anda (seperti AZ, Region, keluarga instans, atau jenis instans) berubah.
- Gunakan keragaman instans dan Zona Ketersediaan: Maksimalkan ketersediaan aplikasi dan manfaatkan kelebihan kapasitas dengan mendiversifikasi instans dan Zona Ketersediaan Anda.

- Instans Rightsize: Gunakan rekomendasi rightsizing dari AWS alat untuk membuat penyesuaian pada beban kerja Anda. Untuk informasi selengkapnya, lihat Mengoptimalkan biaya Anda dengan Rekomendasi Penentuan Ukuran yang Tepat dan Penentuan Ukuran yang Tepat: Menyediakan Instans yang Sesuai dengan Beban Kerja
  - Gunakan rekomendasi rightsizing dalam AWS Cost Explorer atau AWS Compute Optimizeruntuk mengidentifikasi peluang hak.
- Negosiasikan perjanjian tingkat layanan (SLAs): Negosiasikan SLAs yang memungkinkan pengurangan kapasitas sementara sementara sementara otomatisasi menyebarkan sumber daya pengganti.

Sumber daya

Dokumen terkait:

- Mengoptimalkan AWS Infrastruktur Anda untuk Keberlanjutan, Bagian I: Komputasi
- Pemilihan Jenis Instance berbasis Attirbute untuk Auto Scaling untuk Amazon Fleet EC2
- AWS Compute Optimizer Dokumentasi
- Mengoperasikan: Optimasi kinerja
- Dokumentasi Penskalaan Otomatis

Video terkait:

- AWS re:invent 2023 - Apa yang baru dengan Amazon EC2
- AWS re: invent 2023 - Penghematan cerdas: Strategi pengoptimalan biaya Amazon Elastic Compute Cloud
- AWS re:invent 2022 - Mengoptimalkan Amazon Elastic Kubernetes Service untuk kinerja dan biaya AWS
- AWS Re:invent 2023 - Komputasi berkelanjutan: mengurangi biaya dan emisi karbon dengan AWS

SUS05-BP02 Gunakan tipe instans dengan dampak paling kecil

Terus pantau dan gunakan jenis instans baru untuk memanfaatkan peningkatan penghematan energi.

Anti-pola umum:

- Anda hanya menggunakan satu keluarga instans.
- Anda hanya menggunakan instans x86.
- Anda menentukan satu jenis instans dalam konfigurasi Amazon EC2 Auto Scaling.
- Anda menggunakan AWS instance dengan cara yang tidak dirancang (misalnya, Anda menggunakan instance yang dioptimalkan komputasi untuk beban kerja intensif memori).
- Anda tidak mengevaluasi jenis instans baru secara teratur.
- Anda tidak memeriksa rekomendasi dari alat AWS hak cipta seperti.AWS Compute Optimizer

Manfaat menerapkan praktik terbaik ini: Dengan memanfaatkan instans hemat energi dan berukuran tepat, Anda dapat jauh mengurangi dampak lingkungan dan biaya beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Menggunakan instans yang efisien dalam beban kerja cloud sangat penting untuk menurunkan penggunaan sumber daya dan menghemat biaya. Terus lakukan pemantauan terhadap rilis instans jenis baru dan manfaatkan peningkatan penghematan energi, termasuk jenis instans yang dirancang untuk mendukung beban kerja spesifik seperti pelatihan dan inferensi machine learning, serta transkode video.

Langkah-langkah implementasi

- Pelajari dan jelajahi jenis instans: Temukan jenis instans yang dapat menurunkan dampak lingkungan beban kerja Anda.
  - Berlangganan Apa yang Baru AWS untuk tetap up-to-date AWS menggunakan teknologi dan contoh terbaru.
  - Pelajari tentang berbagai jenis AWS instance.
  - Pelajari tentang instans AWS berbasis Graviton yang menawarkan kinerja terbaik per watt penggunaan energi di Amazon EC2 dengan menonton Re:Invent 2020 - Menyelam jauh pada instans Amazon yang AWS didukung prosesor Graviton2 dan menyelam jauh ke instans Graviton3 dan Amazon EC2 C7g. AWS EC2
- Gunakan jenis instans dengan dampak paling kecil: Rencanakan dan transisikan beban kerja Anda ke jenis instans dengan dampak paling kecil.
  - Tentukan sebuah proses untuk mengevaluasi fitur atau instans baru untuk beban kerja Anda. Manfaatkan ketangkasan di cloud untuk menguji dengan cepat bagaimana jenis instans baru

dapat meningkatkan keberlanjutan beban kerja Anda. Gunakan metrik-metrik proksi untuk mengukur berapa banyak sumber daya yang Anda perlukan untuk menyelesaikan satu unit pekerjaan.

- Jika memungkinkan, ubah beban kerja Anda agar bekerja dengan jumlah memori yang berbeda vCPUs dan jumlah memori yang berbeda untuk memaksimalkan pilihan jenis instans Anda.
- Pertimbangkan untuk mengalihkan beban kerja Anda ke instans berbasis Graviton guna meningkatkan efisiensi performa beban kerja Anda. Untuk informasi selengkapnya tentang memindahkan beban kerja ke AWS Graviton, lihat Mulai Cepat Graviton dan Pertimbangan saat mentransisikan beban kerja ke instans Amazon Elastic Compute Cloud berbasis AWS Graviton. AWS
- Pertimbangkan untuk memilih opsi AWS Graviton dalam penggunaan layanan AWS terkelola Anda.
- Migrasikan beban kerja ke Wilayah yang menawarkan instans dengan dampak paling sedikit terhadap keberlanjutan dan masih memenuhi kebutuhan bisnis Anda.
- Untuk beban kerja pembelajaran mesin, manfaatkan perangkat keras yang dibuat khusus untuk beban kerja Anda seperti AWS Trainium, Inferentia, dan Amazon.AWS EC2 DL1 AWS Instans Inferentia seperti instans Inf2 menawarkan kinerja per watt hingga 50% lebih baik dibandingkan instans Amazon yang sebanding. EC2
- Gunakan Amazon SageMaker Inference Recommender untuk mengukur titik akhir inferensi ML yang tepat.
- Untuk beban kerja yang berfluktuasi (beban kerja yang jarang memerlukan kapasitas tambahan), gunakan instans performa yang dapat melonjak.
- Untuk beban kerja tanpa kewarganegaraan dan toleran terhadap kesalahan, gunakan Instans EC2 Spot Amazon untuk meningkatkan pemanfaatan cloud secara keseluruhan, dan mengurangi dampak keberlanjutan sumber daya yang tidak digunakan.

- Operasikan dan optimalkan: Operasikan dan optimalkan instans beban kerja Anda.
  - Untuk beban kerja sementara, evaluasi CloudWatch metrik Amazon instance seperti `CPUUtilization` untuk mengidentifikasi apakah instans tidak digunakan atau kurang dimanfaatkan.
  - Untuk beban kerja yang stabil, periksa alat AWS pengatur ukuran seperti secara AWS Compute Optimizerberkala untuk mengidentifikasi peluang untuk mengoptimalkan dan mengukur instans dengan tepat. Untuk contoh dan rekomendasi lebih lanjut, silakan lihat lab berikut:
    - Lab Well-Architected - Rekomendasi Penyesuaian Ukuran
    - Lab Well-Architected - Penyesuaian Ukuran dengan Pengoptimal Komputasi

- Well-Architected Lab - Optimalkan Pola Perangkat Keras dan Keberlanjutan Observice KPIs

Sumber daya

Dokumen terkait:

- Mengoptimalkan AWS Infrastruktur Anda untuk Keberlanjutan, Bagian I: Komputasi
- AWS Graviton
- Amazon EC2 DL1
- Armada Reservasi EC2 Kapasitas Amazon
- Armada EC2 Spot Amazon
- Fungsi: Konfigurasi Fungsi Lambda
- Pemilihan jenis instans berbasis atribut untuk Amazon Fleet EC2
- Membangun Aplikasi yang Berkelanjutan, Efisien, dan Dioptimalkan untuk Biaya di AWS
- Bagaimana Dasbor Keberlanjutan Contino Membantu Pelanggan Mengoptimalkan Jejak Karbon Mereka

Video terkait:

- AWS Re: invent 2023 - AWS Graviton: Kinerja harga terbaik untuk beban kerja Anda AWS
- AWS re: invent 2023 - Kemampuan AI generatif Amazon Elastic Compute Cloud baru di AWS Management Console
- AWS re:invent 2023 = Apa yang baru dengan Amazon Elastic Compute Cloud
- AWS re: invent 2023 - Penghematan cerdas: Strategi pengoptimalan biaya Amazon Elastic Compute Cloud
- AWS re:invent 2021 - Selami instans AWS Graviton3 dan Amazon C7g EC2
- AWS re:invent 2022 - Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya

Contoh terkait:

- Solusi: Panduan untuk Mengoptimalkan Beban Kerja Deep Learning untuk Keberlanjutan AWS
- Migrasi Basis Data Amazon Relational Database Service ke Graviton

SUS05-BP03 Gunakan layanan terkelola

Gunakan layanan-layanan terkelola untuk beroperasi dengan lebih efisien di cloud.

Anti-pola umum:

- Anda menggunakan EC2 instans Amazon dengan pemanfaatan rendah untuk menjalankan aplikasi Anda.
- Tim internal Anda hanya mengelola beban kerja, tanpa ada waktu untuk berfokus melakukan inovasi atau simplifikasi.
- Anda melakukan deployment dan memelihara teknologi untuk tugas-tugas yang dapat dijalankan dengan lebih efisien di layanan-layanan terkelola.

Manfaat menjalankan praktik terbaik ini:

- Menggunakan layanan terkelola mengalihkan tanggung jawab AWS, yang memiliki wawasan di jutaan pelanggan yang dapat membantu mendorong inovasi dan efisiensi baru.
- Layanan terkelola mendistribusikan dampak lingkungan dari layanan ke banyak pengguna karena bidang kontrol multi-prinsip.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Layanan terkelola mengalihkan tanggung jawab AWS untuk mempertahankan pemanfaatan tinggi dan optimalisasi keberlanjutan perangkat keras yang digunakan. Layanan-layanan terkelola juga dapat menghilangkan beban administratif dan operasional yang muncul dalam pemeliharaan layanan, sehingga tim Anda dapat memiliki lebih banyak waktu dan berfokus untuk membuat inovasi.

Tinjau beban kerja Anda untuk mengidentifikasi komponen yang dapat diganti dengan layanan AWS terkelola. Misalnya, AmazonRDS, Amazon Redshift, dan Amazon ElastiCache menyediakan layanan database terkelola. Amazon Athena, AmazonEMR, dan Amazon OpenSearch Service menyediakan layanan analitik terkelola.

Langkah-langkah implementasi

1. Buat inventarisasi beban kerja Anda: Inventarisasikan beban kerja Anda untuk layanan dan komponen.

2. Identifikasi kandidat: Nilai dan identifikasi komponen yang dapat digantikan oleh layanan terkelola. Berikut ini adalah beberapa contoh kapan Anda mungkin harus mempertimbangkan untuk menggunakan layanan terkelola:

| Tugas | Apa yang harus digunakan pada AWS |
| --- | --- |
| Meng-hosting basis data | Gunakan instans Amazon Relational Database Service (RDSAmazon) yang dikelola alih-alih memelihara instans Amazon Anda sendiri di RDS Amazon Elastic Compute Cloud EC2 (Amazon). |
| Hosting beban kerja kontainer | Gunakan AWS Fargate, alih-alih mengimple mentasikan infrastruktur kontainer Anda sendiri. |
| Hosting aplikasi web | Gunakan AWS Amplify Hosting sebagai CI/ CD dan layanan hosting terkelola sepenuhny a untuk situs web statis dan aplikasi web yang dirender sisi server. |

3. Buat rencana migrasi: Identifikasi dependensi dan buatlah sebuah rencana migrasi. Perbarui runbook dan playbook sesuai dengannya.
    - AWS Application Discovery Service secara otomatis mengumpulkan dan menyajikan informasi terperinci tentang dependensi dan pemanfaatan aplikasi untuk membantu Anda membuat keputusan yang lebih tepat saat merencanakan migrasi Anda
4. Lakukan pengujian Uji layanan sebelum migrasi ke layanan terkelola.
5. Ganti layanan yang dihosting sendiri: Gunakan paket migrasi Anda untuk mengganti layanan-layanan yang dihosting sendiri dengan layanan terkelola.
6. Pantau dan sesuaikan: Terus pantau layanan setelah migrasi selesai untuk membuat penyesuaian sebagaimana diperlukan dan optimalkan layanan.

Sumber daya

Dokumen terkait:

- AWS Cloud Produk

- AWS Total Biaya Kepemilikan (TCO) Kalkulator
- Amazon DocumentDB
- Layanan Amazon Elastic Kubernetes () EKS
- Amazon Managed Streaming for Apache Kafka (Amazon) MSK

Video terkait:

- AWS re:invent 2021 - Operasi cloud dalam skala besar dengan AWS Managed Services
- AWS Re:invent 2023 - Praktik terbaik untuk beroperasi di AWS

SUS05-BP04 Optimalkan penggunaan akselerator komputasi berbasis perangkat keras

Optimalkan penggunaan instans komputasi terakselerasi Anda untuk mengurangi permintaan-permintaan infrastruktur fisik beban kerja Anda.

Anti-pola umum:

- Anda tidak memantau GPU penggunaan.
- Anda menggunakan instans tujuan umum untuk beban kerja, padahal instans yang dibuat khusus dapat menghadirkan kinerja yang lebih tinggi, biaya yang lebih rendah, dan kinerja per watt yang lebih baik.
- Anda menggunakan akselerator komputasi berbasis perangkat keras untuk tugas-tugas di mana mereka lebih efisien menggunakan alternatif berbasis. CPU

Manfaat menerapkan praktik terbaik ini: Dengan mengoptimalkan penggunaan akselerator berbasis perangkat keras, Anda dapat mengurangi permintaan infrastruktur fisik untuk beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Jika Anda memerlukan kemampuan pemrosesan yang tinggi, Anda bisa mendapatkan keuntungan dari menggunakan instans komputasi yang dipercepat, yang menyediakan akses ke akselerator komputasi berbasis perangkat keras seperti unit pemrosesan grafis (GPUs) dan array gerbang yang dapat diprogram lapangan (). FPGAs Akselerator perangkat keras ini melakukan fungsi tertentu seperti pemrosesan grafis atau pencocokan pola data lebih efisien daripada alternatif CPU berbasis.

Banyak beban kerja terakselerasi, seperti perenderan, transkode, dan machine learning, yang memiliki variabel tinggi sehubungan dengan penggunaan sumber daya. Jalankan perangkat keras ini hanya ketika diperlukan, dan non-aktifkan instans GPU secara otomatis saat tidak diperlukan, guna meminimalkan sumber daya yang digunakan.

Langkah-langkah implementasi

- Identifikasi instans komputasi terakselerasi yang dapat memenuhi kebutuhan Anda.
- Untuk beban kerja pembelajaran mesin, manfaatkan perangkat keras yang dibuat khusus untuk beban kerja Anda, seperti AWS Trainium, Inferentia, dan Amazon.AWS EC2 DL1 AWS Instans Inferentia seperti instans Inf2 menawarkan kinerja per watt hingga 50% lebih baik dibandingkan instans Amazon yang sebanding. EC2
- Kumpulkan metrik penggunaan untuk instans komputasi terakselerasi Anda. Misalnya, Anda dapat menggunakan CloudWatch agen untuk mengumpulkan metrik seperti `utilization_gpu` dan `utilization_memory` untuk Anda GPUs seperti yang ditunjukkan dalam Kumpulkan NVIDIA GPU metrik dengan Amazon. CloudWatch
- Optimalkan kode, operasi jaringan, dan pengaturan akselerator perangkat keras untuk memastikan perangkat keras yang mendasarinya dimanfaatkan sepenuhnya.
  - Optimalkan GPU pengaturan
  - GPUPemantauan dan Optimalisasi dalam Pembelajaran Mendalam AMI
  - Mengoptimalkan I/O untuk penyetelan GPU kinerja pelatihan pembelajaran mendalam di Amazon SageMaker
- Gunakan pustaka dan driver berkinerja tinggi terbaru. GPU
- Gunakan otomatisasi untuk merilis GPU instance saat tidak digunakan.

Sumber daya

Dokumen terkait:

- Komputasi yang Dipercepat
- Mari Merancang! Merancang arsitektur dengan chip dan akselerator kustom
- Bagaimana cara memilih jenis EC2 instans Amazon yang sesuai untuk beban kerja saya?
- EC2VT1Contoh Amazon
- Pilih akselerator AI terbaik dan kompilasi model untuk inferensi visi komputer dengan Amazon SageMaker

Video terkait:

- AWS re:invent 2021 - Cara memilih EC2 GPU instans Amazon untuk pembelajaran mendalam
- AWS Pembicaraan Teknologi Online - Menyebarkan Inferensi Pembelajaran Mendalam yang Hemat Biaya
- AWS Re:invent 2023 - AI mutakhir dengan dan AWS NVIDIA
- AWS Re: ciptakan 2022 - [!] NEW LAUNCH Memperkenalkan instans AWS Amazon Inf2 berbasis Inferensia2 EC2
- AWS Re:invent 2022 - Mempercepat pembelajaran mendalam dan berinovasi lebih cepat dengan AWS Trainium
- AWS re:invent 2022 - Pembelajaran mendalam AWS denganNVIDIA: Dari pelatihan hingga penerapan

## Proses dan budaya

Pertanyaan

- SUS6 Bagaimana proses organisasi Anda mendukung tujuan keberlanjutan Anda?

### SUS6 Bagaimana proses organisasi Anda mendukung tujuan keberlanjutan Anda?

Cari peluang untuk mengurangi dampak operasi Anda terhadap keberlanjutan dengan membuat perubahan pada praktik deployment, pengujian, dan pengembangan.

Praktik terbaik

- SUS06-BP01 Mengadopsi metode yang dapat dengan cepat memperkenalkan peningkatan keberlanjutan
- SUS06-BP02 Jaga beban kerja Anda up-to-date
- SUS06-BP03 Meningkatkan pemanfaatan lingkungan build
- SUS06-BP04 Gunakan peternakan perangkat terkelola untuk pengujian

#### SUS06-BP01 Mengadopsi metode yang dapat dengan cepat memperkenalkan peningkatan keberlanjutan

Adopsi metode dan proses untuk memvalidasi potensi peningkatan, meminimalkan biaya pengujian, dan memberikan peningkatan-peningkatan kecil.

Anti-pola umum:

- Meninjau aplikasi Anda untuk keberlanjutan adalah tugas yang dilakukan hanya satu kali pada awal proyek.
- Beban kerja Anda telah kedaluwarsa, karena proses rilis terlalu merepotkan guna melakukan perubahan kecil untuk efisiensi sumber daya.
- Anda tidak memiliki mekanisme untuk meningkatkan beban kerja untuk keberlanjutan.

Manfaat menerapkan praktik terbaik ini: Dengan membangun proses untuk memperkenalkan dan melacak peningkatan keberlanjutan, Anda akan dapat terus mengadopsi fitur dan kemampuan baru, menghilangkan masalah, dan meningkatkan efisiensi beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Uji dan validasi potensi peningkatan keberlanjutan sebelum melakukan deployment peningkatan ini ke produksi. Pertimbangkan biaya pengujian saat menghitung potensi manfaat dari sebuah peningkatan di masa mendatang. Kembangkan metode-metode pengujian berbiaya rendah untuk memberikan peningkatan kecil.

Langkah-langkah implementasi

- Memahami dan mengkomunikasikan tujuan keberlanjutan organisasi Anda: Memahami tujuan keberlanjutan organisasi Anda, seperti pengurangan karbon atau pengelolaan air. Terjemahkan tujuan tersebut menjadi persyaratan keberlanjutan untuk beban kerja cloud Anda. Komunikasikan persyaratan-persyaratan tersebut kepada para pemangku kepentingan utama.
- Tambahkan persyaratan keberlanjutan ke backlog Anda: Tambahkan persyaratan untuk melakukan peningkatan keberlanjutan ke backlog pengembangan Anda.
- Ulang-ulang dan tingkatkan: Gunakan proses perbaikan berulang untuk mengidentifikasi, mengevaluasi, memprioritaskan, menguji, dan men-deploy peningkatan ini.
- Uji menggunakan produk minimum yang layak (MVP): Mengembangkan dan menguji potensi perbaikan menggunakan komponen perwakilan minimum yang layak untuk mengurangi biaya dan dampak lingkungan dari pengujian.
- Sederhanakan prosesnya: Terus tingkatkan dan sederhanakan proses pengembangan Anda. Sebagai contoh, Lakukan otomastisasi terhadap proses pengiriman perangkat lunak Anda menggunakan pipeline integrasi dan pengiriman berkelanjutan (CI/CD) untuk menguji dan

melakukan deployment pada peningkatan-peningkatan potensial untuk mengurangi tingkat upaya dan membatasi kesalahan yang disebabkan oleh proses manual.

- Pelatihan dan kesadaran: Jalankan program pelatihan untuk para anggota tim Anda untuk mendidik mereka tentang keberlanjutan dan bagaimana aktivitas mereka memengaruhi tujuan keberlanjutan organisasi Anda.
- Nilai dan sesuaikan: Terus nilai dampak peningkatan dan buat penyesuaian jika diperlukan.

Sumber daya

Dokumen terkait:

- AWS memungkinkan solusi keberlanjutan
- Praktik pengembangan tangkas yang dapat diskalakan berdasarkan AWS CodeCommit

Video terkait:

- AWS Re:invent 2023 - Arsitektur berkelanjutan: Masa lalu, sekarang, dan masa depan
- AWS re:invent 2022 - Memberikan arsitektur yang berkelanjutan dan berkinerja tinggi
- AWS re:invent 2022 - Merancang secara berkelanjutan dan mengurangi jejak karbon Anda AWS
- AWS re:invent 2022 - Keberlanjutan dalam infrastruktur global AWS
- AWS re:invent 2023 - Apa yang baru dengan observabilitas dan operasi AWS

Contoh terkait:

- Lab Well-Architected - Mengubah laporan biaya & penggunaan menjadi laporan efisiensi

SUS06-BP02 Jaga beban kerja Anda up-to-date

Pertahankan beban kerja Anda up-to-date untuk mengadopsi fitur yang efisien, menghapus masalah, dan meningkatkan efisiensi keseluruhan beban kerja Anda.

Anti-pola umum:

- Anda berasumsi bahwa arsitektur Anda saat ini adalah arsitektur statis dan tidak akan diperbarui seiring waktu.

- Anda tidak memiliki sistem atau koordinasi rutin untuk mengevaluasi apakah perangkat lunak dan paket-paket yang diperbarui kompatibel dengan beban kerja Anda.

Manfaat menerapkan praktik terbaik ini: Dengan menetapkan proses untuk menjaga agar beban kerja Anda tetap yang terbaru, Anda akan dapat menerapkan fitur dan kemampuan baru, menyelesaikan masalah, dan meningkatkan efisiensi beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Sistem operasi, runtime, perangkat lunak perantara (middleware), pustaka, dan aplikasi yang mutakhir dapat meningkatkan efisiensi beban kerja serta memudahkan Anda melakukan adopsi teknologi yang lebih efisien. Perangkat lunak yang mutakhir juga dapat menyertakan fitur-fitur untuk mengukur dampak beban kerja terhadap keberlanjutan secara lebih akurat, mengingat vendor juga menghadirkan fitur-fitur untuk memenuhi tujuan keberlanjutan mereka sendiri. Secara teratur jaga agar beban kerja Anda diperbarui dengan fitur dan rilis terbaru.

Langkah-langkah implementasi

- Tentukan sebuah proses: Gunakan sebuah proses dan jadwal untuk mengevaluasi fitur atau instans baru untuk beban kerja Anda. Manfaatkan ketangkasan di cloud untuk menguji dengan cepat bagaimana fitur-fitur baru dapat meningkatkan beban kerja Anda untuk:
  - Mengurangi dampak keberlanjutan.
  - Memperoleh efisiensi performa.
  - Menghilangkan penghalang untuk peningkatan terencana.
  - Meningkatkan kemampuan Anda dalam mengukur dan mengelola dampak terhadap keberlanjutan.
- Buat inventaris beban kerja: Buatlah inventaris perangkat lunak dan arsitektur beban kerja Anda dan identifikasi komponen yang perlu diperbarui.
  - Anda dapat menggunakan AWS Systems Manager Inventaris untuk mengumpulkan sistem operasi (OS), aplikasi, dan metadata instans dari instans Amazon Anda dan dengan cepat memahami EC2 instans mana yang menjalankan perangkat lunak dan konfigurasi yang diperlukan oleh kebijakan perangkat lunak Anda dan instans mana yang perlu diperbarui.
- Pelajari prosedur pembaruan: Pahami cara memperbarui komponen beban kerja Anda.

| Komponen Beban Kerja | Cara memperbarui |
| --- | --- |
| Citra mesin | Gunakan EC2Image Builder untuk mengelola pembaruan Amazon Machine Images (AMIs) untuk gambar server Linux atau Windows. |
| Gambar kontainer | Gunakan Amazon Elastic Container Registry (AmazonECR) dengan pipeline yang ada untuk mengelola gambar Amazon Elastic Container Service (AmazonECS). |
| AWS Lambda | AWS Lambda termasuk fitur manajemen versi. |

- Gunakan otomatisasi: Lakukan otomatisasi terhadap proses pembaruan guna mengurangi tingkat upaya dalam melakukan deployment fitur baru dan membatasi kesalahan yang disebabkan oleh proses manual.
  - Anda dapat menggunakan CI/CD untuk memperbaruiAMIs, gambar kontainer, dan artefak lain yang terkait dengan aplikasi cloud Anda secara otomatis.
  - Anda dapat menggunakan alat seperti AWS Systems Manager Patch Manager untuk melakukan otomatisasi proses pembaruan sistem, dan menjadwalkan aktivitas menggunakan AWS Systems Manager Windows Maintenance.

Sumber daya

Dokumen terkait:

- AWS Pusat Arsitektur
- Apa yang baru dengan AWS
- AWS Alat Pengembang

Video terkait:

- AWS Re:invent 2022 - Optimalkan beban AWS kerja Anda dengan panduan praktik terbaik
- All Things Patch: AWS Systems Manager

Contoh terkait:

- Lab Well-Architected - Manajemen Inventaris dan Patch
- Lab: AWS Systems Manager

SUS06-BP03 Meningkatkan pemanfaatan lingkungan build

Tingkatkan pemanfaatan sumber daya untuk mengembangkan, menguji, dan membangun beban kerja Anda.

Anti-pola umum:

- Anda menyediakan atau menghentikan lingkungan build Anda secara manual.
- Anda mempertahankan lingkungan-lingkungan build terus berjalan terlepas dari aktivitas pengujian, build, atau rilis (misalnya, menjalankan lingkungan di luar jam kerja anggota tim pengembangan Anda).
- Anda menyediakan terlalu banyak sumber daya untuk lingkungan build Anda.

Manfaat menerapkan praktik terbaik ini: Dengan meningkatkan pemanfaatan lingkungan build, Anda dapat meningkatkan efisiensi keseluruhan beban kerja cloud Anda sekaligus mengalokasikan sumber daya untuk kepada builder untuk melakukan pengembangan, pengujian, dan pembangunan secara efisien.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Gunakan otomatisasi dan infrastructure-as-code untuk meningkatkan lingkungan build saat dibutuhkan dan menurunkannya saat tidak digunakan. Hal yang umum dilakukan adalah menjadwalkan periode ketersediaan yang bertepatan dengan jam kerja anggota tim pengembangan. Lingkungan pengujian Anda harus sangat mirip dengan konfigurasi lingkungan produksi. Namun, cari peluang untuk menggunakan jenis instans dengan kapasitas burst, Instans EC2 Spot Amazon, layanan database penskalaan otomatis, wadah, dan teknologi tanpa server untuk menyelaraskan pengembangan dan kapasitas pengujian dengan penggunaan. Batasi volume data untuk tepat memenuhi persyaratan pengujian. Jika Anda menggunakan data produksi dalam pengujian, jelajahi kemungkinan berbagi data dari produksi dan tidak memindahkan data ke mana-mana.

Langkah-langkah implementasi

- Menggunakan infrastruktur sebagai kode: Gunakan infrastruktur sebagai kode untuk menyediakan lingkungan build Anda.

- Gunakan otomatisasi: Gunakan otomatisasi untuk mengelola siklus hidup pengembangan dan menguji lingkungan serta memaksimalkan efisiensi sumber daya build Anda.
- Maksimalkan pemanfaatan: Gunakan strategi untuk memaksimalkan pemanfaatan lingkungan pengembangan dan pengujian.
  - Gunakan lingkungan representatif yang dapat digunakan pada tingkat minimum untuk mengembangkan dan menguji potensi peningkatan yang mungkin dilakukan.
  - Gunakan teknologi nirserver jika mungkin.
  - Gunakan Instans Sesuai Permintaan untuk membantu perangkat-perangkat pengembang Anda.
  - Gunakan jenis instans dengan kapasitas lonjakan, Instans Spot, dan teknologi-teknologi lainnya untuk menyesuaikan kapasitas build dengan penggunaan.
  - Adopsi layanan-layanan cloud native untuk akses shell instans yang aman daripada melakukan deployment armada host bastion.
  - Menskalakan secara otomatis sumber daya build Anda sesuai dengan tugas build Anda.

Sumber daya

Dokumen terkait:

- AWS Manajer Sesi Systems Manager
- Contoh kinerja Amazon EC2 Burstable
- Apa itu AWS CloudFormation?
- Apa itu AWS CodeBuild?
- Penjadwal Instance aktif AWS

Video terkait:

- AWS RE: invent 2023 - Integrasi dan pengiriman berkelanjutan untuk AWS

SUS06-BP04 Gunakan peternakan perangkat terkelola untuk pengujian

Gunakan device farm terkelola untuk secara efisien menguji fitur baru pada serangkaian perangkat keras representatif.

Anti-pola umum:

- Anda menguji dan melakukan deployment terhadap aplikasi Anda di masing-masing perangkat fisik secara manual.
- Anda tidak menggunakan layanan pengujian aplikasi untuk menguji dan berinteraksi dengan aplikasi Anda (contohnya, Android, iOS, dan aplikasi web) pada perangkat fisik dan nyata.

Manfaat menerapkan praktik terbaik ini: Menggunakan device farm terkelola untuk menguji aplikasi berkemampuan cloud akan memberikan Anda sejumlah manfaat:

- Device farm terkelola disertai dengan fitur yang lebih efisien untuk menguji aplikasi di berbagai macam perangkat.
- Device farm terkelola menghilangkan kebutuhan akan infrastruktur internal untuk melakukan pengujian.
- Device farm terkelola menawarkan berbagai macam jenis perangkat, termasuk perangkat keras yang lebih lama dan kurang populer, yang menghilangkan kebutuhan akan pemutakhiran perangkat yang tidak perlu.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

Menggunakan device farm terkelola dapat membantu Anda untuk menyederhanakan proses pengujian untuk fitur baru di serangkaian perangkat keras representatif. Device farm terkelola menawarkan berbagai jenis perangkat, termasuk perangkat keras yang lebih lama dan kurang populer, serta menghindari dampak keberlanjutan pelanggan akibat pemutakhiran perangkat yang tidak perlu.

Langkah-langkah implementasi

- Tentukan persyaratan pengujian: Tetapkan persyaratan pengujian Anda dan rencanakan (seperti jenis pengujian, sistem operasi, dan jadwal pengujian).
  - Anda dapat menggunakan Amazon CloudWatch RUM untuk mengumpulkan dan menganalisis data sisi klien dan membentuk rencana pengujian Anda.
- Pilih device farm terkelola: Pilih device farm terkelola yang dapat mendukung persyaratan pengujian Anda. Misalnya, Anda dapat menggunakan AWS Device Farm untuk menguji dan memahami dampak perubahan yang terjadi pada perangkat keras yang representatif.
- Gunakan otomatisasi: Gunakan otomatisasi dan integrasi berkelanjutan/deployment berkelanjutan (CI/CD) untuk menjadwalkan dan menjalankan pengujian Anda.

  - Mengintegrasikan AWS Device Farm dengan pipeline CI/CD Anda untuk menjalankan pengujian Selenium lintas-browser
  - Membangun dan menguji aplikasi iOS dan iPad OS dengan AWS DevOps dan layanan seluler
- Tinjau dan sesuaikan: Terus tinjau hasil pengujian Anda dan buat peningkatan yang perlu.

Sumber daya

Dokumen terkait:

- AWS Device Farm daftar perangkat
- Melihat CloudWatch RUM dasbor

Video terkait:

- AWS re:invent 2023 - Tingkatkan kualitas aplikasi seluler dan web Anda menggunakan Device Farm AWS
- AWS re:invent 2021 - Optimalkan aplikasi melalui wawasan pengguna akhir dengan Amazon CloudWatch RUM

Contoh terkait:

- AWS Device Farm Contoh Aplikasi untuk Android
- AWS Device Farm Contoh Aplikasi untuk iOS
- Tes Appium Web untuk AWS Device Farm

# Pemberitahuan

Pelanggan bertanggung jawab untuk membuat penilaian independen mereka sendiri atas informasi dalam dokumen ini. Dokumen ini: (a) hanya untuk tujuan informasi, (b) mewakili penawaran dan praktik AWS produk saat ini, yang dapat berubah tanpa pemberitahuan, dan (c) tidak membuat komitmen atau jaminan apa pun dari AWS dan afiliasinya, pemasok, atau pemberi lisensinya. AWS produk atau layanan disediakan “sebagaimana adanya” tanpa jaminan, representasi, atau kondisi apa pun, baik tersurat maupun tersirat. Tanggung jawab dan kewajiban AWS kepada pelanggannya dikendalikan oleh AWS perjanjian, dan dokumen ini bukan bagian dari, juga tidak mengubah, perjanjian apa pun antara AWS dan pelanggannya.

# AWS Glosarium

Untuk AWS terminologi terbaru, lihat AWS glosarium di Referensi.Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.