Laporan Resmi AWS

Pengenalan DevOps di AWS

Pengenalan DevOps di AWS: Laporan Resmi AWS

Table of Contents

Pengenalan DevOps di AWS

Tanggal publikasi: 16 Oktober 2020 (Revisi Dokumen)

Abstrak

Saat ini lebih dari sebelumnya, korporasi memulai perjalanan transformasi digital mereka untuk membangun hubungan yang lebih mendalam dengan pelanggan guna mencapai nilai bisnis yang berkelanjutan dan tak lekang waktu. Organisasi dari segala bentuk dan ukuran menimbulkan disrupsi bagi pesaing mereka dan memasuki pasar baru dengan berinovasi lebih cepat daripada sebelumnya. Untuk organisasi ini, penting untuk fokus pada disrupsi inovasi dan perangkat lunak, sehingga sangat penting untuk menyederhanakan pengiriman perangkat lunak mereka. Organisasi yang mempersingkat waktu mereka dari ide ke produksi dengan memprioritaskan kecepatan dan ketangkasan dapat menjadi pendisrupsi di masa mendatang.

Meskipun ada beberapa faktor yang perlu dipertimbangkan dalam menjadi pendisrupsi digital berikutnya, laporan resmi ini fokus pada DevOps, serta layanan dan fitur di platform AWS yang akan membantu meningkatkan kemampuan organisasi untuk menghadirkan aplikasi dan layanan dengan kecepatan tinggi.

Pengantar

DevOps adalah kombinasi budaya, praktik dan pola rekayasa, serta alat yang meningkatkan kemampuan organisasi untuk memberikan aplikasi dan layanan dengan kecepatan tinggi dan kualitas yang lebih baik. Seiring waktu, beberapa praktik penting telah muncul saat mengadopsi DevOps: Integrasi Berkelanjutan, Pengiriman Berkelanjutan, Infrastruktur sebagai Kode (IaC), serta Pemantauan dan Pencatatan Log.

Dokumen ini menyoroti kemampuan AWS yang membantu Anda mempercepat perjalanan DevOps Anda, dan bagaimana layanan AWS dapat membantu menghilangkan pekerjaan berat yang tidak terdiferensiasi terkait adaptasi DevOps. Kami juga menyoroti cara membangun kemampuan integrasi dan pengiriman berkelanjutan tanpa mengelola server atau membangun node, dan cara memanfaatkan Infrastruktur sebagai Kode (IaC) untuk menyediakan dan mengelola sumber daya cloud Anda dalam cara yang konsisten dan dapat diulang.

- Integrasi Berkelanjutan: adalah praktik pengembangan software yang memungkinkan developer secara berkala menggabungkan perubahan kode mereka ke repositori pusat, yang setelahnya dilanjutkan dengan pembangunan otomatis dan pengujian.
- Pengiriman Berkelanjutan: adalah praktik pengembangan perangkat lunak yang memungkinkan perubahan kode dibangun, diuji, dan dipersiapkan secara otomatis untuk dirilis ke tahap produksi.
- Infrastruktur sebagai Kode (IaC): adalah praktik yang memungkinkan infrastruktur disediakan dan dikelola menggunakan kode dan teknik pengembangan perangkat lunak, seperti kontrol versi dan integrasi berkelanjutan.
- Pemantauan dan Pencatatan Log: memungkinkan organisasi melihat bagaimana performa aplikasi dan infrastruktur memengaruhi pengalaman pengguna akhir produk mereka.
- Komunikasi dan Kolaborasi: Praktik dibuat untuk mempererat tim dan dengan membangun alur kerja dan mendistribusikan tanggung jawab untuk DevOps.
- Keamanan: harus memengaruhi banyak aspek. Alur integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) dan layanan terkait harus dilindungi dan izin kontrol akses yang tepat harus disiapkan.

Pemeriksaan terhadap masing-masing prinsip ini mengungkapkan hubungan yang dekat dengan penawaran yang tersedia dari Amazon Web Services (AWS).

Integrasi Berkelanjutan

Integrasi berkelanjutan (CI) adalah praktik pengembangan perangkat lunak yang memungkinkan developer menggabungkan perubahan kode mereka ke dalam repositori pusat secara rutin, lalu menjalankan pembangunan dan pengujian terotomatisasi. CI membantu menemukan dan mengatasi bug lebih cepat, meningkatkan kualitas perangkat lunak, dan mengurangi waktu yang dibutuhkan untuk melakukan validasi dan meluncurkan pembaruan perangkat lunak terbaru.

AWS menawarkan layanan berikut untuk integrasi berkelanjutan:

Topik

- AWS CodeCommit
- AWS CodeBuild
- AWS CodeArtifact

AWS CodeCommit

AWS CodeCommit adalah layanan kontrol sumber yang aman, dapat diskalakan, dan terkelola penuh yang meng-host repositori git privat. CodeCommit menghilangkan kebutuhan bagi Anda untuk mengoperasikan sistem kontrol sumber Anda sendiri dan tidak ada perangkat keras yang harus disediakan dan diskalakan atau perangkat lunak untuk diinstal, dikonfigurasi, dan dioperasikan. Anda dapat menggunakan CodeCommit untuk menyimpan apa pun mulai dari kode hingga biner, dan mendukung fungsionalitas standar Git, sehingga memungkinkannya bekerja secara lancar dengan alat berbasis Git yang ada. Tim Anda juga dapat menggunakan alat kode online CodeCommit untuk menelusuri, mengedit, dan berkolaborasi dalam suatu proyek. AWS CodeCommit memiliki beberapa manfaat:

Kolaborasi - AWS CodeCommit dirancang untuk pengembangan perangkat lunak kolaboratif. Anda dapat dengan mudah melakukan, membuat cabang, dan menggabungkan kode Anda yang memungkinkan Anda dengan mudah mempertahankan kontrol atas proyek tim Anda. CodeCommit juga mendukung permintaan pull, yang menyediakan mekanisme untuk meminta tinjauan kode dan mendiskusikan kode dengan kolaborator.

Enkripsi - Anda dapat mentransfer file Anda ke dan dari AWS CodeCommit menggunakan HTTPS atau SSH, seperti yang Anda inginkan. Repositori Anda juga secara otomatis dienkripsi saat at rest melalui AWS Key Management Service (AWS KMS) menggunakan kunci khusus pelanggan.

Kontrol Akses - AWS CodeCommit menggunakan AWS Identity and Access Management (IAM) untuk mengontrol dan memantau siapa yang dapat mengakses data Anda selain bagaimana, kapan, dan di mana mereka dapat mengaksesnya. CodeCommit juga membantu Anda memantau repositori Anda melalui AWS CloudTrail dan Amazon CloudWatch.

Ketersediaan dan Daya Tahan Tinggi- AWS CodeCommit menyimpan repositori Anda di Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB. Data terenkripsi Anda disimpan secara redundan di berbagai fasilitas. Arsitektur ini meningkatkan ketersediaan dan daya tahan data repositori Anda.

Notifikasi dan Skrip Kustom - Anda sekarang dapat menerima notifikasi untuk peristiwa yang memengaruhi repositori Anda. Notifikasi akan muncul sebagai notifikasi Amazon Simple Notification Service (Amazon SNS). Setiap notifikasi akan menyertakan pesan status serta tautan ke sumber daya yang peristiwanya menghasilkan notifikasi tersebut. Selain itu, menggunakan pemicu repositori AWS CodeCommit, Anda dapat mengirim notifikasi dan membuat webhook HTTP dengan Amazon SNS atau memanggil fungsi AWS Lambda sebagai respons terhadap peristiwa repositori yang Anda pilih.

AWS CodeBuild

AWS CodeBuild merupakan layanan integrasi berkelanjutan yang terkelola penuh untuk menyusun kode sumber, menjalankan pengujian, dan menghasilkan paket perangkat lunak yang siap di-deploy. Dengan CodeBuild, Anda tidak perlu menyediakan, mengelola, dan menskalakan server pembuatan Anda sendiri. CodeBuild dapat menggunakan salah satu dari GitHub, GitHub Enterprise, BitBucket, AWS CodeCommit, atau Amazon S3 sebagai penyedia sumber.

CodeBuild menskalakan secara terus-menerus dan dapat memproses beberapa build secara bersamaan. CodeBuild menawarkan berbagai lingkungan yang dikonfigurasi sebelumnya untuk berbagai versi Microsoft Windows dan Linux. Pelanggan juga dapat menjadikan lingkungan build yang disesuaikan sebagai kontainer Docker. CodeBuild juga terintegrasi dengan alat sumber terbuka seperti Jenkins dan Spinnaker.

CodeBuild juga dapat membuat laporan untuk pengujian unit, fungsi, atau integrasi. Laporan-laporan ini memberikan pandangan visual tentang berapa banyak kasus pengujian dijalankan dan berapa banyak yang lulus atau gagal. Proses build ini juga dapat dijalankan di dalam Amazon Virtual Private Cloud (Amazon VPC) yang dapat membantu jika layanan integrasi atau basis data Anda di-deploy di dalam VPC.

AWS CodeArtifact

AWS CodeArtifact adalah layanan repositori artefak terkelola penuh yang memudahkan organisasi dari berbagai ukuran untuk menyimpan, memublikasikan, dan berbagi paket perangkat lunak yang digunakan dalam proses pengembangan perangkat lunak mereka dengan aman. CodeArtifact dapat dikonfigurasikan untuk secara otomatis mengambil paket perangkat lunak dan dependensi dari repositori artefak publik sehingga developer memiliki akses ke versi terbaru.

Tim pengembangan perangkat lunak semakin mengandalkan paket sumber terbuka untuk melakukan tugas umum dalam paket aplikasi mereka. Sekarang penting bagi tim pengembangan perangkat lunak untuk mempertahankan kontrol pada versi tertentu dari perangkat lunak sumber terbuka agar bebas kelemahan. Dengan CodeArtifact, Anda dapat menyiapkan kontrol untuk menerapkan hal tersebut.

CodeArtifact beroperasi dengan package manager yang umum digunakan dan alat build, seperti Maven, Gradle, npm, yarn, twine, dan pip sehingga mudah untuk diintegrasikan ke dalam alur kerja pengembangan yang ada.

Pengiriman Berkelanjutan

Pengiriman berkelanjutan adalah praktik pengembangan perangkat lunak yang memungkinkan perubahan kode disiapkan secara otomatis untuk rilis ke tahap produksi. Pilar dari pengembangan aplikasi modern, pengiriman berkelanjutan mengembangkan integrasi berkelanjutan dengan men-deploy semua perubahan kode ke lingkungan pengujian dan/atau lingkungan produksi setelah tahap pembangunan. Apabila pengiriman berkelanjutan dilakukan dengan benar, developer akan selalu memiliki artefak pembuatan yang siap di-deploy yang telah melewati proses pengujian terstandardisasi.

Pengiriman berkelanjutan memungkinkan developer mengotomatisasi pengujian lebih dari hanya pengujian unit sehingga mereka dapat memverifikasi pembaruan aplikasi di beberapa dimensi sebelum men-deploy ke pelanggan. Pengujian ini mungkin termasuk pengujian UI, pengujian beban, pengujian integrasi, pengujian keandalan API, dll. Ini membantu developer lebih teliti memvalidasi pembaruan dan secara preemtif menemukan masalah. Dengan cloud, mengotomatisasi pembuatan dan replikasi beberapa lingkungan untuk pengujian menjadi mudah dan hemat biaya, yang sebelumnya sulit dilakukan on-premise.

AWS menawarkan layanan berikut untuk pengiriman berkelanjutan:

- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline

Topik

- AWS CodeDeploy
- AWS CodePipeline

AWS CodeDeploy

AWS CodeDeploy adalah layanan deployment yang terkelola penuh yang mengotomatisasi deployment perangkat lunak ke berbagai layanan komputasi seperti Amazon Elastic Compute Cloud (Amazon EC2) AWS Fargate, AWS Lambda, dan server on-premise Anda. AWS CodeDeploy memudahkan Anda untuk merilis fitur baru dengan cepat, membantu Anda menghindari waktu henti selama deployment aplikasi, dan menangani kompleksitas dalam memperbarui aplikasi Anda. Anda

dapat menggunakan CodeDeploy untuk men-deploy perangkat lunak secara otomatis, sehingga mengurangi operasi manual yang rawan kesalahan. Skala layanannya menyesuaikan dengan kebutuhan deployment Anda.

CodeDeploy memiliki beberapa manfaat yang sesuai dengan prinsip DevOps untuk deployment berkelanjutan:

Deployment Otomatis: CodeDeploy men-deploy perangkat lunak Anda secara otomatis sepenuhnya, sehingga memungkinkan Anda men-deploy secara andal dan cepat.

Kontrol terpusat: CodeDeploy memungkinkan Anda meluncurkan dan melacak status deployment aplikasi Anda dengan mudah melalui AWS Management Console atau AWS CLI. CodeDeploy memberi laporan mendetail yang memungkinkan Anda melihat kapan dan ke mana setiap revisi aplikasi di-deploy. Anda juga dapat membuat notifikasi push untuk menerima pembaruan secara langsung tentang deployment yang Anda lakukan.

Meminimalkan waktu henti: CodeDeploy membantu memaksimalkan ketersediaan aplikasi Anda selama proses deployment perangkat lunak. CodeDeploy memperkenalkan perubahan secara bertahap dan melacak kondisi aplikasi sesuai aturan yang dapat dikonfigurasi. Deployment perangkat lunak dapat dihentikan dan dijalankan kembali dengan mudah jika terdapat kesalahan.

Mudah diadopsi: CodeDeploy bekerja dengan aplikasi apa pun, dan memberikan pengalaman yang sama di berbagai platform dan bahasa. Anda dapat dengan mudah menggunakan kembali kode penyiapan yang sudah ada. CodeDeploy juga dapat berintegrasi dengan proses rilis perangkat lunak Anda yang sudah ada atau toolchain pengiriman berkelanjutan (misalnya, AWS CodePipeline, GitHub, Jenkins).

AWS CodeDeploy mendukung beberapa opsi deployment. Untuk informasi selengkapnya, lihat Strategi Deployment.

AWS CodePipeline

AWS CodePipeline adalah layanan pengiriman berkelanjutan yang memungkinkan Anda memodelkan, memvisualisasikan, dan mengotomatiskan langkah-langkah yang diperlukan untuk merilis perangkat lunak. Dengan AWS CodePipeline, Anda memodelkan proses rilis lengkap untuk membangun kode Anda, men-deploy ke lingkungan pra-produksi, menguji aplikasi Anda, dan merilisnya ke produksi. AWS CodePipeline kemudian membangun, menguji, dan men-deploy aplikasi Anda sesuai dengan alur kerja yang ditentukan setiap kali ada perubahan kode. Anda dapat

mengintegrasikan alat Partner AWS dan alat kustom milik Anda sendiri ke dalam tahap mana pun dalam proses rilis untuk membentuk solusi pengiriman berkelanjutan secara menyeluruh.

AWS CodePipeline memiliki beberapa manfaat yang sesuai dengan prinsip DevOps dari deployment berkelanjutan:

Pengiriman Cepat: AWS CodePipeline mengotomatiskan proses rilis perangkat lunak Anda, sehingga memungkinkan Anda merilis fitur baru kepada pengguna dengan cepat. Dengan CodePipeline, Anda dapat mempelajari umpan balik dengan cepat dan mengirimkan fitur baru kepada pelanggan secara lebih cepat.

Peningkatan Kualitas: Dengan mengotomatiskan proses membangun, menguji, dan merilis, AWS CodePipeline memungkinkan Anda meningkatkan kecepatan dan kualitas pembaruan perangkat lunak dengan memproses semua perubahan baru melalui serangkaian pemeriksaan kualitas yang konsisten.

Mudah Diintegrasikan: AWS CodePipeline dapat diperluas dengan mudah untuk menyesuaikan dengan kebutuhan spesifik Anda. Anda dapat menggunakan plugin bawaan atau plugin kustom Anda sendiri dalam setiap tahap proses rilis. Misalnya, Anda dapat menarik kode sumber dari GitHub, menggunakan server pembuatan Jenkins on-premise Anda, menjalankan uji pemuatan menggunakan layanan pihak ketiga, atau meneruskan informasi deployment ke dasbor operasi kustom Anda.

Alur Kerja yang Dapat Dikonfigurasi: AWS CodePipeline memungkinkan Anda memodelkan berbagai tahapan proses rilis perangkat lunak Anda menggunakan antarmuka konsol, AWS CLI AWS CloudFormation, atau SDK AWS. Anda dapat dengan mudah menentukan pengujian yang akan dijalankan dan menyesuaikan langkah-langkah deployment aplikasi Anda dan dependensinya.

Strategi Deployment

Strategi deployment menentukan bagaimana Anda ingin mengirimkan perangkat lunak Anda. Organisasi mengikuti strategi deployment yang berbeda berdasarkan model bisnis mereka. Beberapa orang mungkin memilih untuk mengirimkan perangkat lunak yang sepenuhnya diuji, dan yang lain mungkin ingin pengguna mereka memberikan umpan balik dan membiarkan pengguna mereka mengevaluasi dalam fitur pengembangan (misalnya, rilis beta). Pada bagian berikut, kita akan berbicara tentang berbagai strategi deployment.

Topik

Deployment Di Tempat

Dalam strategi ini, deployment dilakukan dengan aplikasi pada setiap instans dalam grup deployment dihentikan, revisi aplikasi terbaru diinstal, dan versi baru aplikasi dimulai dan divalidasi. Anda dapat menggunakan penyeimbang beban sehingga setiap instans dibatalkan registrasinya selama deployment dan kemudian dikembalikan ke layanan setelah deployment tersebut selesai. Deployment Di Tempat dapat diterapkan secara sekaligus, dengan mengasumsikan pemadaman layanan, atau dilakukan sebagai pembaruan bergulir. AWS CodeDeploy dan AWS Elastic Beanstalk menawarkan konfigurasi deployment untuk satu per satu waktu, setengah pada satu waktu, dan sekaligus. Strategi deployment yang sama ini untuk deployment di tempat tersedia dalam deployment biru/hijau.

Deployment Biru/Hijau

Deployment biru/hijau, kadang-kadang disebut sebagai merah-hitam, adalah teknik untuk merilis aplikasi dengan lalu lintas pergeseran di antara dua lingkungan identik yang menjalankan versi yang berbeda dari suatu aplikasi. Deployment biru/hijau membantu Anda meminimalkan waktu henti selama pembaruan aplikasi sehingga mengurangi risiko seputar waktu henti dan fungsi rollback. Deployment biru/hijau memungkinkan Anda meluncurkan versi baru (hijau) aplikasi Anda bersama

versi lama (biru), dan memantau serta menguji versi baru sebelum Anda merutekan ulang lalu lintas ke versi baru ini, dan melakukan rollback jika terdeteksi masalah.

Deployment Canary

Lalu lintas digeser dalam dua peningkatan. Deployment canary adalah strategi biru/hijau yang sangat menghindari risiko, yang memungkinkan penggunaan pendekatan bertahap. Deployment canary bisa berupa dua langkah atau linier tempat kode aplikasi baru di-deploy dan dipaparkan untuk uji coba, dan setelah penerimaan, akan diluncurkan baik ke sisa lingkungan tersebut atau secara linier.

Deployment Linier

Deployment linier berarti bahwa lalu lintas digeser dalam peningkatan yang sama dengan jumlah menit yang sama di antara setiap peningkatan. Anda dapat memilih dari pilihan linier yang telah ditentukan sebelumnya yang menetapkan persentase lalu lintas yang digeser dalam setiap peningkatan dan jumlah menit di antara setiap peningkatan.

Deployment Sekaligus

Deployment sekaligus berarti bahwa semua lalu lintas bergeser dari lingkungan asli ke lingkungan pengganti secara sekaligus.

Matriks Strategi Deployment

Matriks berikut mencantumkan strategi deployment yang didukung untuk Amazon Elastic Container Service (Amazon ECS), AWS Lambda, dan Amazon EC2/On-Premise.

- Amazon ECS adalah layanan orkestrasi yang terkelola penuh.
- AWS Lambda memungkinkan Anda menjalankan kode tanpa menyediakan atau mengelola server.
- Amazon EC2 memungkinkan Anda menjalankan kapasitas komputasi yang aman dan dapat diubah ukurannya di cloud.

	A	B	C	D
1	Matriks Strategi Deployment	Amazon ECS	AWS Lambda	Amazon EC2/ On-Premise
2	Di Tempat	✓	✓	✓
3	Biru/Hijau	✓	✓	✓*
4	Canary	✓	✓	X
5	Linier	✓	✓	X
6	Sekaligus	✓	✓	X

> **Note**
>
> Deployment biru/hijau dengan EC2/On-Premise hanya berfungsi dengan instans EC2.

Strategi Deployment AWS Elastic Beanstalk

AWS Elastic Beanstalk mendukung jenis strategi deployment berikut:

- Sekaligus: Melakukan deployment di tempat pada semua instans.
- Rolling: Memisahkan instans menjadi sejumlah batch dan men-deploy ke batch satu per satu.

- Rolling dengan Batch Tambahan: Memisahkan deployment menjadi sejumlah batch tetapi untuk batch pertama membuat instans EC2 baru, bukan men-deploy pada instans EC2 yang ada.
- Tetap: Jika Anda perlu men-deploy dengan instans baru, bukan menggunakan instans yang ada.
- Pembagian Lalu Lintas: Melakukan deployment tetap dan kemudian meneruskan persentase lalu lintas ke instans baru untuk durasi waktu yang telah ditentukan sebelumnya. Jika instans tetap berkondisi baik, teruskan semua lalu lintas ke instans baru dan matikan instans lama.

Infrastruktur sebagai Kode (IaC)

Prinsip mendasar dari DevOps adalah memperlakukan infrastruktur dengan cara yang sama seperti developer memperlakukan suatu kode. Kode aplikasi memiliki format dan sintaks yang ditetapkan. Jika kode tidak ditulis sesuai dengan aturan bahasa pemrograman, aplikasi tidak dapat dibuat. Kode disimpan dalam manajemen versi atau sistem kontrol sumber yang mencatat log riwayat pengembangan kode, perubahan, dan perbaikan bug. Saat kode dikompilasi atau dibangun ke dalam aplikasi, kita mengharapkan aplikasi yang konsisten akan dibuat, serta build-nya dapat diulang dan dapat diandalkan.

Mempraktikkan infrastruktur sebagai kode (IaC) berarti menerapkan keakuratan pengembangan kode aplikasi yang sama ke penyediaan infrastruktur. Semua konfigurasi harus didefinisikan dengan cara deklaratif dan disimpan dalam sistem kontrol sumber seperti AWS CodeCommit, sama seperti kode aplikasi. Penyediaan infrastruktur, orkestrasi, dan deployment juga harus mendukung penggunaan infrastruktur sebagai kode (IaC).

Infrastruktur secara tradisional disediakan menggunakan kombinasi skrip dan proses manual. Terkadang skrip ini disimpan dalam sistem kontrol versi atau didokumentasikan langkah demi langkah dalam file teks atau run-book. Sering kali orang yang menulis run-book bukanlah orang yang sama yang mengeksekusi skrip ini atau menindaklanjuti run-book tersebut. Jika skrip atau runbook ini tidak sering diperbarui, skrip dan runbook tersebut berpotensi menjadi faktor penghambat dalam deployment. Hal ini menghasilkan pembuatan lingkungan baru tidak selalu dapat diulang, dapat diandalkan, atau konsisten.

Berbeda dengan hal di atas, AWS menyediakan cara yang berfokus pada DevOps untuk membuat dan memelihara infrastruktur. Mirip dengan cara developer perangkat lunak menulis kode aplikasi, AWS menyediakan layanan yang memungkinkan pembuatan, deployment, dan pemeliharaan infrastruktur dengan cara yang terprogram, deskriptif, dan deklaratif. Layanan ini memberikan keakuratan, kejelasan, dan keandalan. Layanan AWS yang dibahas dalam dokumen ini merupakan inti dari metodologi DevOps dan membentuk dasar dari berbagai prinsip dan praktik AWS DevOps tingkat lebih tinggi.

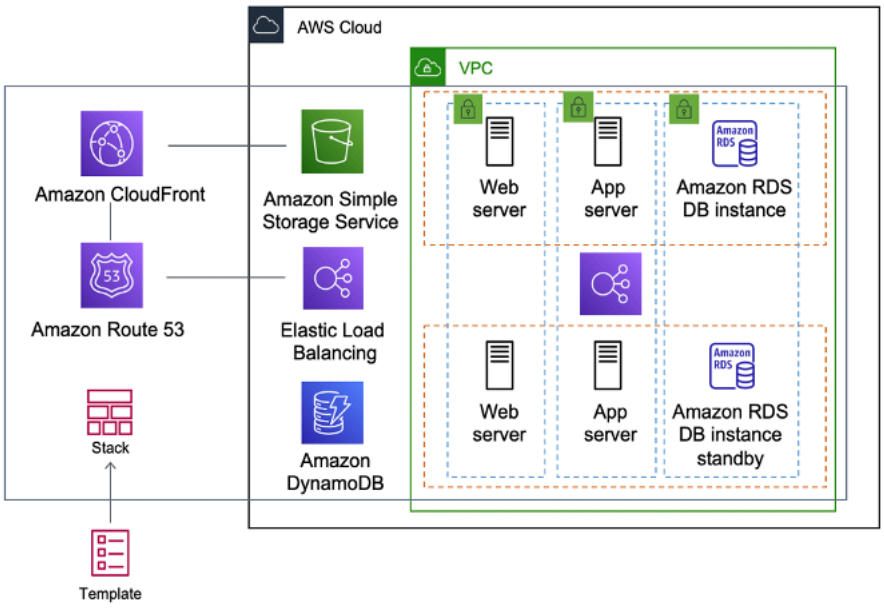
AWS menawarkan layanan berikut untuk menentukan Infrastruktur sebagai kode (IaC).

- AWS CloudFormation
- AWS Cloud Development Kit (AWS CDK)
- AWS Cloud Development Kit for Kubernetes

AWS CloudFormation

AWS CloudFormation adalah layanan yang memungkinkan developer membuat sumber daya AWS secara teratur dan dapat diprediksi. Sumber daya ditulis dalam file teks menggunakan JavaScript Object Notation (JSON) atau format Yet Another Markup Language (YAML). Templat tersebut memerlukan sintaks dan struktur khusus yang tergantung pada jenis sumber daya yang dibuat dan dikelola. Anda menulis sumber daya Anda di JSON atau YAML dengan editor kode apa pun seperti AWS Cloud9, memeriksanya dalam sistem kontrol versi, dan kemudian CloudFormation membangun layanan yang ditentukan dengan cara yang aman dan dapat diulang.

Templat CloudFormation di-deploy ke lingkungan AWS sebagai tumpukan. Anda dapat mengelola tumpukan melalui AWS Management Console, AWS Command Line Interface, atau AWS CloudFormation API. Jika Anda perlu membuat perubahan pada sumber daya yang berjalan dalam tumpukan, Anda dapat memperbarui tumpukan. Sebelum membuat perubahan pada sumber daya Anda, Anda dapat menghasilkan set perubahan, yang merupakan ringkasan dari perubahan yang Anda ajukan. Set perubahan memungkinkan Anda melihat bagaimana perubahan Anda dapat memengaruhi sumber daya Anda yang sedang berjalan, terutama untuk sumber daya penting, sebelum menerapkannya.



Gambar 1 - AWS CloudFormation membuat seluruh lingkungan (tumpukan) dari satu alur kerja templat

Anda dapat menggunakan satu templat untuk membuat dan memperbarui seluruh lingkungan atau templat terpisah untuk mengelola beberapa lapisan dalam lingkungan. Hal ini memungkinkan templat dimodulasi, dan juga menyediakan lapisan tata kelola yang penting bagi banyak organisasi.

Saat Anda membuat atau memperbarui tumpukan di konsol, peristiwa akan ditampilkan dengan status konfigurasi. Jika terjadi kesalahan, secara default tumpukan di-rollback ke kondisi sebelumnya. Amazon Simple Notification Service (Amazon SNS) memberikan pemberitahuan tentang peristiwa. Misalnya, Anda dapat menggunakan Amazon SNS untuk melacak kemajuan pembuatan dan penghapusan tumpukan melalui email dan berintegrasi dengan proses lain secara terprogram.

AWS CloudFormation memudahkan dalam mengatur dan men-deploy koleksi sumber daya AWS dan memungkinkan Anda mendeskripsikan dependensi apa pun atau meneruskan parameter khusus saat tumpukan dikonfigurasi.

Dengan templat CloudFormation, Anda dapat bekerja dengan serangkaian layanan AWS yang luas, seperti Amazon S3, Auto Scaling, Amazon CloudFront, Amazon DynamoDB, Amazon EC2, Amazon ElastiCache, AWS Elastic Beanstalk, Elastic Load Balancing, IAM AWS OpsWorks, dan Amazon VPC. Untuk daftar sumber daya terbaru yang didukung, lihat referensi jenis sumber daya dan properti AWS.

AWS Cloud Development Kit

AWS Cloud Development Kit (AWS CDK) adalah kerangka kerja pengembangan perangkat lunak sumber terbuka untuk membuat model dan menyediakan sumber daya aplikasi cloud Anda menggunakan bahasa pemrograman yang biasa digunakan. AWS CDK memungkinkan Anda membuat model infrastruktur aplikasi menggunakan TypeScript, Python, Java, dan .NET. Developer dapat memanfaatkan Integrated Development Environment (IDE) mereka yang ada, dengan menggunakan alat seperti autocomplete dan in-line documentation untuk mempercepat pengembangan infrastruktur.

AWS CDK menggunakan AWS CloudFormation di latar belakang untuk menyediakan sumber daya dengan cara yang aman dan dapat diulang. Konstruksi adalah komponen dasar untuk kode CDK. Konstruksi mewakili komponen cloud dan merangkum semua yang diperlukan AWS CloudFormation untuk membuat komponen tersebut. AWS CDK menyertakan AWS Construct Library yang berisi konstruksi yang mewakili banyak layanan AWS. Dengan menggabungkan konstruksi bersama-sama, Anda dapat secara cepat dan mudah membuat arsitektur kompleks untuk deployment di AWS.

AWS Cloud Development Kit for Kubernetes

AWS Cloud Development Kit for Kubernetes (cdk8s), adalah kerangka kerja pengembangan perangkat lunak sumber terbuka untuk mendefinisikan aplikasi Kubernetes menggunakan bahasa pemrograman tujuan umum.

Setelah Anda menetapkan aplikasi Anda dalam bahasa pemrograman (Pada tanggal publikasi, hanya Python dan TypeScript yang didukung), cdk8s akan mengonversi deskripsi aplikasi Anda menjadi YML pra-Kubernetes. File YML ini kemudian dapat digunakan oleh klaster Kubernetes yang berjalan di mana saja. Karena struktur didefinisikan dalam bahasa pemrograman, Anda dapat menggunakan fitur kaya yang disediakan oleh bahasa pemrograman. Anda dapat menggunakan fitur abstraksi bahasa pemrograman untuk membuat kode boiler-plate Anda sendiri dan menggunakannya kembali di semua deployment.

Otomatisasi

Filosofi inti dan praktik DevOps lainnya adalah otomatisasi. Otomatisasi berfokus pada penyiapan, konfigurasi, deployment, dan dukungan infrastruktur serta aplikasi yang berjalan di atasnya. Dengan menggunakan otomatisasi, Anda dapat menyiapkan lingkungan lebih cepat dengan cara standar dan berulang. Penghapusan proses manual adalah kunci untuk strategi DevOps yang sukses. Secara historis, konfigurasi server dan deployment aplikasi sebagian besar merupakan proses manual. Lingkungan menjadi non-standar, dan memunculkan kembali lingkungan saat masalah muncul menjadi sulit.

Penggunaan otomatisasi sangat penting untuk mewujudkan manfaat penuh dari cloud. AWS secara internal sangat bergantung pada otomatisasi untuk menyediakan fitur inti elastisitas dan skalabilitas. Proses manual rawan kesalahan, tidak dapat diandalkan, dan tidak memadai untuk mendukung bisnis yang tangkas. Sering kali sebuah organisasi mungkin mengikat sumber daya berketerampilan tinggi untuk menyediakan konfigurasi manual, yang seharusnya dikerahkan untuk mendukung aktivitas lain yang lebih penting dan bernilai lebih tinggi dalam bisnis.

Lingkungan operasi modern biasanya bergantung pada otomatisasi penuh untuk menghilangkan intervensi manual atau akses ke lingkungan produksi. Ini mencakup semua rilis perangkat lunak, konfigurasi mesin, patching sistem operasi, pemecahan masalah, atau perbaikan bug. Banyak tingkat praktik otomatisasi dapat digunakan bersama-sama untuk memberikan proses otomatis yang menyeluruh dengan tingkat yang lebih tinggi.

Otomatisasi memiliki manfaat utama berikut:

- Perubahan cepat
- Peningkatan produktivitas
- Konfigurasi berulang
- Lingkungan yang dapat dimunculkan kembali
- Elastisitas yang lebih baik
- Penskalaan otomatis yang lebih baik
- Pengujian otomatis

Otomatisasi adalah landasan dalam layanan AWS dan didukung secara internal di semua layanan, fitur, dan penawaran.

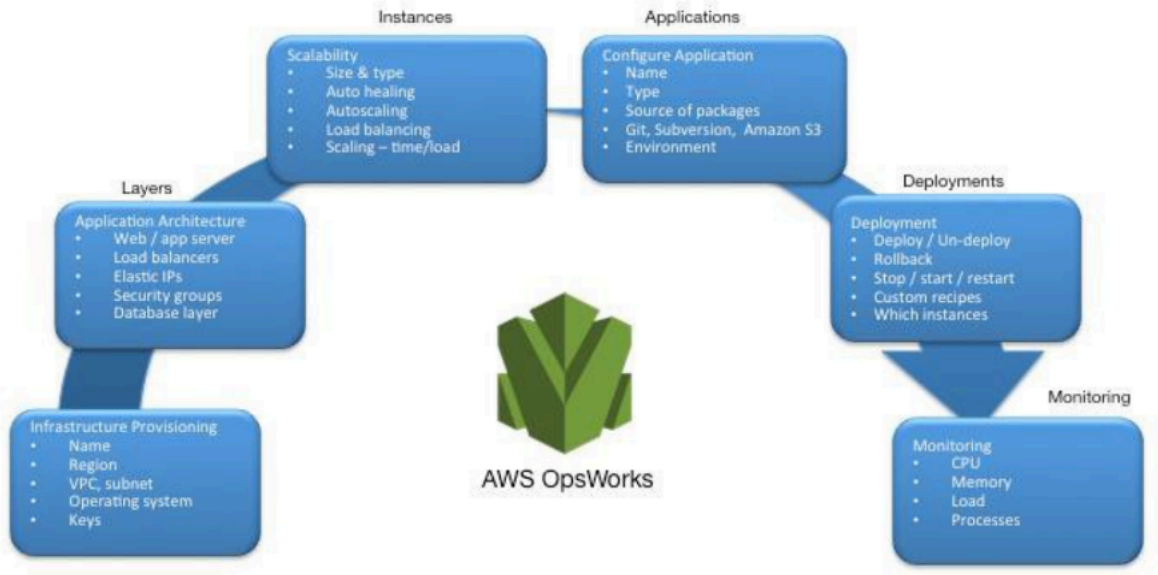
Topik

- AWS OpsWorks
- AWS Elastic Beanstalk

AWS OpsWorks

AWS OpsWorks mengambil prinsip-prinsip DevOps lebih jauh dari AWS Elastic Beanstalk. Hal ini dapat dianggap sebagai layanan manajemen aplikasi, bukan hanya kontainer aplikasi. AWS OpsWorks menyediakan lebih banyak tingkat otomatisasi dengan fitur tambahan seperti integrasi dengan perangkat lunak manajemen konfigurasi (Chef) dan manajemen siklus hidup aplikasi. Anda dapat menggunakan manajemen siklus hidup aplikasi untuk menentukan kapan sumber daya disiapkan, dikonfigurasi, di-deploy, tidak di-deploy, atau dimatikan.

Untuk menambahkan fleksibilitas AWS OpsWorks telah memungkinkan Anda mendefinisikan aplikasi Anda dalam tumpukan yang dapat dikonfigurasi. Anda juga dapat memilih tumpukan aplikasi yang telah ditetapkan sebelumnya. Tumpukan aplikasi berisi semua penyediaan untuk sumber daya AWS yang diperlukan aplikasi Anda, termasuk server aplikasi, server web, basis data, dan penyeimbang beban.



Gambar 2 - AWS OpsWorks menampilkan fitur dan arsitektur DevOps

Tumpukan aplikasi diatur ke dalam sejumlah lapisan arsitektur sehingga tumpukan dapat dipertahankan secara independen. Contoh lapisan dapat mencakup tingkat web, tingkat aplikasi, dan

tingkat basis data. Karena siap pakai, AWS OpsWorks juga menyederhanakan penyiapan grup Auto Scaling dan penyeimbang beban Elastic Load Balancing, yang lebih lanjut menggambarkan prinsip otomatisasi DevOps. Sama seperti AWS Elastic Beanstalk, AWS OpsWorks mendukung versioning aplikasi, deployment berkelanjutan, dan manajemen konfigurasi infrastruktur.

AWS OpsWorks juga mendukung praktik pemantauan dan pencatatan log DevOps (tercakup dalam bagian berikutnya). Dukungan pemantauan disediakan oleh Amazon CloudWatch. Semua peristiwa siklus hidup dicatat, dan log Chef terpisah mendokumentasikan resep Chef yang dijalankan, bersama dengan pengecualian apa pun.

AWS Elastic Beanstalk

AWS Elastic Beanstalk adalah layanan untuk cepat men-deploy dan menskalakan aplikasi web yang dikembangkan dengan Java, .NET, PHP, Node.js, Python, Ruby, Go, dan Docker pada server yang mudah digunakan seperti Apache, NGINX, Passenger, dan IIS.

Elastic Beanstalk adalah abstraksi di atas Amazon EC2, Auto Scaling, dan menyederhanakan deployment dengan memberikan fitur tambahan seperti kloning, deployment biru/hijau, Elastic Beanstalk Command Line Interface (eb cli) dan integrasi dengan AWS Toolkit for Visual Studio, Visual Studio Code, Eclipse, dan IntelliJ untuk meningkatkan produktivitas developer.

Pemantauan dan Pencatatan Log

Komunikasi dan kolaborasi adalah hal mendasar dalam filosofi DevOps. Untuk memudahkan hal ini, umpan balik sangat penting. Di AWS, umpan balik disediakan oleh dua layanan inti: Amazon CloudWatch dan AWS CloudTrail. Bersama-sama, kedua layanan tersebut menyediakan infrastruktur pemantauan, peringatan, dan audit yang tangguh sehingga developer dan tim operasi dapat bekerja sama secara erat dan transparan.

AWS menyediakan layanan berikut untuk pemantauan dan pencatatan log:

Topik

- Amazon CloudWatch
- Alarm Amazon CloudWatch
- Amazon CloudWatch Logs
- Amazon CloudWatch Logs Insights
- Amazon CloudWatch Events
- Amazon EventBridge
- AWS CloudTrail

Amazon CloudWatch

Metrik Amazon CloudWatch secara otomatis mengumpulkan data dari layanan AWS seperti instans Amazon EC2, volume Amazon EBS, dan instans Amazon RDS DB. Metrik ini kemudian dapat diatur sebagai dasbor dan alarm atau peristiwa dapat dibuat untuk memicu peristiwa atau melakukan tindakan Auto Scaling.

Alarm Amazon CloudWatch

Anda dapat menyiapkan alarm berdasarkan metrik yang dikumpulkan oleh Metrik Amazon CloudWatch. Alarm kemudian dapat mengirim notifikasi ke topik Amazon Simple Notification Service (Amazon SNS) atau memulai tindakan Auto Scaling. Alarm memerlukan periode (lama waktu untuk mengevaluasi metrik), Periode Evaluasi (jumlah titik data terbaru), dan Titik Data ke Alarm (jumlah titik data dalam Periode Evaluasi).

Amazon CloudWatch Logs

Amazon CloudWatch Logs adalah layanan agregasi dan pemantauan log. AWS CodeBuild, CodeCommit, CodeDeploy, dan CodePipeline menyediakan integrasi dengan log CloudWatch sehingga semua log dapat dipantau secara terpusat. Selain itu, berbagai layanan yang disebutkan sebelumnya dan layanan AWS lainnya memberikan integrasi langsung dengan CloudWatch.

Dengan CloudWatch Logs, Anda dapat:

- Mengkueri Data Log Anda
- Memantau Log dari Instans Amazon EC2
- Memantau Peristiwa yang Dicatat AWS CloudTrail
- Menentukan Kebijakan Retensi Log

Amazon CloudWatch Logs Insights

Amazon CloudWatch Logs Insights memindai log Anda dan memungkinkan Anda melakukan kueri dan visualisasi interaktif. Amazon CloudWatch Logs Insights memahami berbagai format log dan menemukan bidang dari Log JSON secara otomatis.

Amazon CloudWatch Events

Amazon CloudWatch Events menghadirkan aliran peristiwa sistem mendekati waktu nyata yang mendeskripsikan perubahan dalam sumber daya AWS. Dengan aturan sederhana yang dapat Anda siapkan dengan cepat, Anda dapat menyesuaikan peristiwa dan merutekannya ke satu atau beberapa fungsi atau aliran target. CloudWatch Events akan mengetahui perubahan operasional yang terjadi. CloudWatch Events merespons perubahan operasional ini dan mengambil tindakan korektif seperlunya, dengan mengirimkan pesan untuk merespons lingkungan, mengaktifkan fungsi, membuat perubahan, dan menangkap informasi status.

Anda dapat mengonfigurasi aturan di CloudWatch Events untuk mengingatkan Anda tentang perubahan dalam layanan AWS dan mengintegrasikan peristiwa ini dengan sistem pihak ketiga lainnya menggunakan Amazon EventBridge. Berikut layanan terkait AWS DevOps yang memiliki integrasi dengan CloudWatch Events.

- Peristiwa Application Auto Scaling
- Peristiwa CodeBuild

- Peristiwa CodeCommit
- Peristiwa CodeDeploy
- Peristiwa CodePipeline

Amazon EventBridge

Amazon CloudWatch Events dan EventBridge adalah layanan dasar dan API yang sama, namun, EventBridge menyediakan lebih banyak fitur.

Amazon EventBridge adalah bus peristiwa nirserver yang memungkinkan integrasi antara layanan AWS, Perangkat Lunak sebagai Layanan (SaaS), dan aplikasi Anda. Selain membangun aplikasi berbasis peristiwa, EventBridge dapat digunakan untuk memberi tahu tentang peristiwa dari layanan seperti CodeBuild, CodeDeploy, CodePipeline, dan CodeCommit.

AWS CloudTrail

Untuk menerapkan prinsip-prinsip kolaborasi, komunikasi, dan transparansi DevOps, penting untuk memahami siapa yang membuat modifikasi pada infrastruktur Anda. Di AWS, transparansi ini disediakan oleh layanan AWS CloudTrail. Semua interaksi AWS ditangani melalui panggilan API AWS yang dipantau dan dicatat oleh AWS CloudTrail. Semua file log yang dihasilkan disimpan dalam bucket Amazon S3 yang Anda tetapkan. File log dienkripsi menggunakan enkripsi sisi server (SSE) Amazon S3. Semua panggilan API dicatat apakah panggilan tersebut langsung dari pengguna atau atas nama pengguna oleh layanan AWS. Banyak grup dapat memperoleh manfaat dari log CloudTrail, termasuk tim operasi untuk dukungan, tim keamanan untuk tata kelola, dan tim keuangan untuk penagihan.

Komunikasi dan Kolaborasi

Apakah Anda mengadopsi Budaya DevOps di organisasi Anda atau sedang menjalani transformasi budaya DevOps, komunikasi dan kolaborasi adalah bagian penting dari pendekatan Anda. Di Amazon, kami telah menyadari bahwa ada kebutuhan untuk menghadirkan perubahan dalam pola pikir tim dan oleh karena itu kami mengadopsi konsep Two-Pizza Teams.

Topik

Two-Pizza Teams

"Kami mencoba membuat tim yang bisa mendapatkan porsi cukup dengan dua piza saja", ungkap Bezos. "Kami menyebutnya aturan Two-Pizza Teams."

Semakin kecil tim, semakin baik kolaborasinya. Kolaborasi juga sangat penting karena rilis perangkat lunak bergerak lebih cepat dari sebelumnya. Dan kemampuan tim untuk menghasilkan perangkat lunak dapat menjadi faktor pembeda bagi organisasi Anda untuk melawan pesaing Anda. Bayangkan situasi yang mengharuskan fitur produk baru perlu dirilis atau bug perlu diperbaiki dan Anda ingin hal ini terjadi secepat mungkin sehingga Anda dapat memiliki waktu go-to-market yang lebih singkat. Hal ini juga penting karena Anda tidak ingin transformasi menjadi proses yang bergerak lebih lambat daripada pendekatan yang tangkas yang memungkinkan gelombang perubahan mulai memberikan dampak.

Komunikasi di antara tim juga penting, kita bergerak menuju model tanggung jawab bersama dan mulai meninggalkan pendekatan pengembangan yang terisolasi. Hal ini menghadirkan konsep kepemilikan dalam tim dan menggeser perspektif mereka untuk melihat hal ini sebagai sesuatu yang menyeluruh. Tim Anda seharusnya tidak berpikir tentang lingkungan produksi Anda sebagai kotak hitam yang tidak menghadirkan visibilitas untuk mereka.

Transformasi budaya juga penting karena Anda mungkin akan membangun tim DevOps umum, atau pendekatan lainnya adalah Anda memiliki satu anggota atau lebih yang berfokus pada DevOps di tim Anda. Kedua pendekatan ini memperkenalkan tanggung jawab bersama ke tim.

Keamanan

Baik Anda sedang menjalani Transformasi DevOps atau menerapkan prinsip-prinsip DevOps untuk pertama kalinya, Anda harus mempertimbangkan Keamanan sebagai sesuatu yang terintegrasi dalam proses DevOps Anda. Hal ini harus memengaruhi banyak aspek di seluruh tahapan pembuatan, pengujian, dan deployment Anda.

Sebelum kita berbicara tentang Keamanan di DevOps di AWS, mari kita lihat Model Tanggung Jawab Bersama AWS.

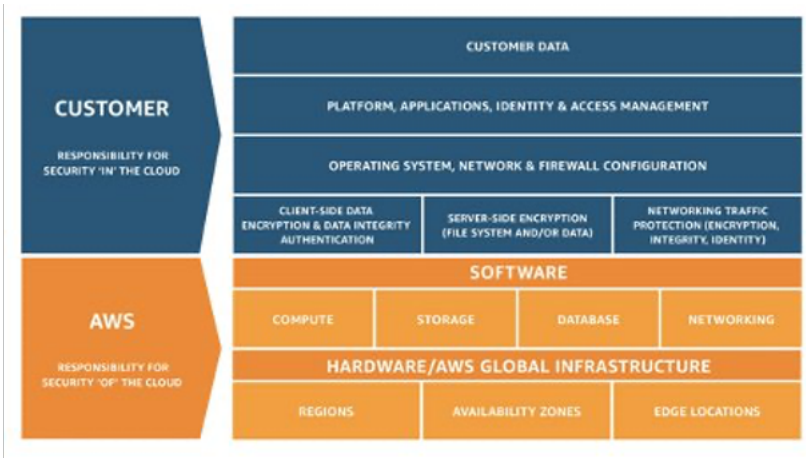
Topik

Model Tanggung Jawab Bersama AWS

Keamanan adalah tanggung jawab bersama antara AWS dan pelanggan. Berbagai bagian yang berbeda dari Model Tanggung Jawab Bersama dijelaskan di bawah ini:

- Tanggung jawab AWS “Keamanan dari Cloud” - AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan semua layanan yang ditawarkan di dalam AWS Cloud. Infrastruktur ini terdiri dari perangkat keras, perangkat lunak, jaringan, dan fasilitas yang menjalankan layanan AWS Cloud.
- Tanggung jawab pelanggan “Keamanan di Cloud” – Tanggung jawab pelanggan akan ditentukan oleh layanan AWS Cloud yang dipilih pelanggan. Hal ini menentukan jumlah pekerjaan konfigurasi yang harus dilakukan pelanggan sebagai bagian dari tanggung jawab keamanan mereka.

Model bersama seperti ini dapat membantu meringankan beban operasional pelanggan karena AWS mengoperasikan, mengelola, dan mengendalikan komponen dari lapisan sistem operasi dan virtualisasi host hingga keamanan fisik dari fasilitas tempat layanan tersebut beroperasi. Hal ini penting dalam kasus saat pelanggan ingin memahami keamanan lingkungan build mereka.



Gambar 3 - Model Tanggung Jawab Bersama AWS

Identity and Access Management

AWS Identity and Access Management (IAM) mendefinisikan kontrol dan kebijakan yang digunakan untuk mengelola akses ke Sumber Daya AWS. Menggunakan IAM, Anda dapat membuat pengguna dan grup serta menentukan izin untuk berbagai layanan DevOps.

Selain pengguna, berbagai layanan mungkin juga memerlukan akses ke sumber daya AWS. Misalnya, proyek CodeBuild Anda mungkin memerlukan akses untuk menyimpan image Docker di Amazon Elastic Container Registry (Amazon ECR) dan memerlukan izin untuk menulis ke Amazon ECR. Jenis izin ini didefinisikan oleh peran jenis khusus yang dikenal sebagai peran layanan.

IAM adalah salah satu komponen infrastruktur keamanan AWS. Dengan IAM, Anda dapat mengelola grup, pengguna, peran layanan, dan kredensial keamanan secara terpusat seperti kata sandi, kunci akses, dan kebijakan izin yang mengontrol layanan dan sumber daya AWS yang dapat diakses pengguna. Kebijakan IAM memungkinkan Anda menentukan rangkaian izin. Kebijakan ini kemudian dapat dilampirkan ke Peran, Pengguna, atau Layanan untuk menentukan izinnya. Anda juga dapat menggunakan IAM untuk membuat peran yang digunakan secara luas dalam strategi DevOps yang Anda inginkan. Dalam beberapa kasus, mungkin sebaiknya gunakan AssumeRole secara terprogram, bukan langsung mendapatkan izin. Saat layanan atau pengguna mengambil peran, layanan atau pengguna ini diberi kredensial sementara untuk mengakses layanan yang biasanya tidak Anda miliki aksesnya.

Kesimpulan

Untuk membuat perjalanan ke cloud menjadi lancar, efisien, dan efektif, perusahaan teknologi harus menerapkan prinsip dan praktik DevOps. Prinsip-prinsip ini tertanam dalam platform AWS. Prinsip-prinsip tersebut membentuk fondasi untuk berbagai layanan AWS, terutama layanan yang ada dalam penawaran deployment dan pemantauan.

Mulailah dengan mendefinisikan infrastruktur Anda sebagai kode menggunakan layanan AWS CloudFormation atau AWS Cloud Development Kit (AWS CDK). Selanjutnya, tentukan cara aplikasi Anda akan menggunakan deployment berkelanjutan dengan bantuan layanan seperti AWS CodeBuild, AWS CodeDeploy, AWS CodePipeline, dan AWS CodeCommit. Pada tingkat aplikasi, gunakan kontainer seperti AWS Elastic Beanstalk, Amazon Elastic Container Service (Amazon ECS), atau Amazon Elastic Kubernetes Service (Amazon EKS), dan AWS OpsWorks untuk menyederhanakan konfigurasi arsitektur umum. Menggunakan layanan ini juga memudahkan untuk menyertakan layanan penting lainnya seperti Auto Scaling dan Elastic Load Balancing. Terakhir, gunakan strategi pemantauan DevOps seperti Amazon CloudWatch, dan praktik keamanan yang solid seperti AWS IAM.

Dengan AWS sebagai partner Anda, prinsip DevOps Anda menghadirkan ketangkasan pada bisnis dan organisasi IT Anda serta mempercepat perjalanan Anda ke cloud.

Revisi Dokumen

Untuk mendapatkan notifikasi tentang pembaruan laporan resmi ini, sebaiknya berlangganan umpan RSS.

perubahan-riwayat-pembaruan	deskripsi-riwayat-pembaruan	tanggal-riwayat-pembaruan
Mengembalikan bagian Kontributor yang hilang	Mengembalikan bagian Kontributor yang hilang dan perubahan teks kecil	21 November 2020
Memperbarui bagian untuk menyertakan layanan baru	Memperbarui bagian untuk menyertakan layanan baru	16 Oktober 2020
Publikasi awal	Laporan resmi pertama kali dipublikasikan	1 Desember 2014

Kontributor

Kontributor dokumen ini meliputi:

- Muhammad Mansoor, Arsitek Solusi (Solutions Architect)
- Ajit Zadgaonkar, Pemimpin Teknis Global, Modernisasi (World Wide Tech Leader, Modernization)
- Juan Lamadrid - Arsitek Solusi (Solutions Architect)
- Darren Ball - Arsitek Solusi (Solutions Architect)
- Rajeswari Malladi - Arsitek Solusi (Solutions Architect)
- Pallavi Nargund - Arsitek Solusi (Solutions Architect)
- Bert Zahniser - Arsitek Solusi (Solutions Architect)
- Abdullahi Olaoye – Arsitek Solusi Cloud (Cloud Solutions Architect)
- Mohamed Kiswani – Manajer Pengembangan Perangkat Lunak (Software Development Manager)
- Tara McCann – Manajer, Arsitek Solusi (Manager, Solutions Architect)

Pemberitahuan

Pelanggan bertanggung jawab untuk membuat penilaian independen mereka sendiri atas informasi dalam dokumen ini. Dokumen ini: (a) hanya disediakan sebagai informasi, (b) berisi penawaran produk dan praktik AWS saat ini, yang dapat berubah tanpa pemberitahuan, dan (c) tidak menjadi komitmen atau jaminan apa pun dari AWS dan afiliasi, pemasok, atau pemberi lisensinya. Produk atau layanan AWS disediakan “sebagaimana adanya” tanpa jaminan, representasi, atau ketentuan apa pun, baik tersurat maupun tersirat. Tanggung jawab dan kewajiban AWS kepada pelanggannya dikendalikan oleh perjanjian AWS, dan dokumen ini bukan bagian dari, juga tidak mengubah, perjanjian apa pun antara AWS dan pelanggannya.