aws

Panduan Developer

AWS Panorama

AWS Panorama: Panduan Developer

Table of Contents

Apa itu AWS Panorama? 1
Mulai 3
Konsep 4
Alat Panorama AWS 4
Perangkat yang kompatibel 4
Aplikasi 5
Simpul 5
Model 5
Pengaturan 7
Prasyarat 7
Daftarkan dan konfigurasikan AWS Panorama Appliance 8
Tingkatkan perangkat lunak alat 11
Menambahkan stream kamera 12
Langkah selanjutnya 13
Men-deploy aplikasi 14
Prasyarat 14
Impor aplikasi 15
Deploy aplikasi 16
Lihat Output 18
Aktifkan SDK for Python 20
Bersihkan 20
Langkah selanjutnya 21
Mengembangkan aplikasi yang 22
Manifes aplikasi manifes 23
Bangunan dengan aplikasi sampel 26
Mengubah model visi komputer 28
Gambar prapemrosesan awal 30
Meng-upload metrik dengan SDK for Python 31
Langkah selanjutnya 34
Model dan kamera yang didukung 35
model yang didukung 35
Kamera yang didukung 36
Spesifikasi perangkat 37
Quotas 39

Izin 40
Kebijakan pengguna 41
Peran layanan 43
Mengamankan peran alat 43
Penggunaan layanan lain 45
Peran aplikasi 47
Alat 48
Mengelola 49
Memperbarui perangkat lunak alat 49
Membatalkan pendaftaran alat 50
Mem-boot ulang alat 50
Mengatur ulang alat 51
Pengaturan jaringan 52
Konfigurasi jaringan tunggal 52
Konfigurasi jaringan ganda 53
Mengkonfigurasi akses layanan 53
Mengkonfigurasi akses jaringan lokal 54
Konektivitas pribadi 54
Kamera 56
Menghapus aliran 57
Aplikasi 58
Tombol dan lampu 59
Lampu status 59
Cahaya jaringan 59
Tombol daya dan reset 60
Mengelola aplikasi 61
Deploy 62
Instal CLI Aplikasi AWS Panorama 62
Mengimpor aplikasi 63
Membangun gambar kontainer 64
Impor model 65
Unggah aset aplikasi 66
Menerapkan aplikasi dengan konsol AWS Panorama 67
Mengotomatiskan penerapan aplikasi 68
Kelola 69
Memperbarui atau menyalin aplikasi 69

Hapus versi dan aplikasi 69
Paket 70
Manifes aplikasi 72
Skema JSON 74
Simpul 75
Tepi 75
Simpul Abstrak 76
Parameter 79
Mengambil alih 81
Aplikasi Pembangunan 83
Model 84
Menggunakan model dalam kode 84
Membangun model khusus 85
Kemasan model 87
Model pelatihan 88
Membangun gambar 89
Menentukan dependensi 90
Penyimpanan lokal 90
Membangun aset gambar 90
AWS SDK 92
Menggunakan Amazon S3 92
Menggunakan topikAWS IoT MQTT 92
Aplikasi SDK 94
Menambahkan teks dan kotak ke output video 94
Menjalankan beberapa thread 96
Melayani lalu lintas masuk 99
Mengonfigurasi port masuk 99
Melayani lalu lintas 101
Menggunakan GPU 105
Tutorial — Lingkungan pengembangan Windows 107
Prasyarat 107
Instal WSL 2 dan Ubuntu 108
Instal Docker 108
Mengonfigurasi Ubuntu 108
Langkah selanjutnya 110
API AWS Panorama 111

Mengotomatiskan pendaftaran perangkat 112
Kelola alat 114
Lihat perangkat 114
Perbarui Perangkat Lunak Perlengkapan 115
Perlengkapan boot ulang 116
Mengotomatiskan penerapan aplikasi 118
Bangun wadah 118
Unggah kontainer dan daftarkan node 118
Deploy aplikasi 119
Pantau penyebaran 121
Mengelola aplikasi 123
Melihat aplikasi 123
Kelola aliran kamera kamera 124
Menggunakan titik akhir VPC 127
Membuat titik akhir VPC 127
Menghubungkan alat ke subnet pribadi 127
Contoh AWS CloudFormation template 128
Sampel 132
Aplikasi sampel 132
Skrip utilitas 133
AWS CloudFormation templat 133
Lebih banyak sampel dan alat 134
Memantau 135
Konsol AWS Panorama 136
Log 137
Melihat log perangkat 137
Melihat log aplikasi 138
Mengonfigurasi log aplikasi 138
Melihat log penyediaan 139
Egressing log dari perangkat 140
Metrik CloudWatch 142
Menggunakan metrik perangkat 142
Menggunakan metrik aplikasi 143
Mengonfigurasi alarm 143
Pemecahan Masalah 144
Penyediaan 144

Konfigurasi alat 144
Konfigurasi aplikasi 145
Aliran kamera 146
Keamanan 147
Fitur keamanan 148
Praktik terbaik 150
Perlindungan data 152
Enkripsi dalam bergerak 153
Alat AWS Panorama 153
Aplikasi 153
Layanan lainnya 154
Pengelolaan identitas dan akses 155
Audiens 155
Mengautentikasi dengan identitas 156
Mengelola akses menggunakan kebijakan 159
Cara AWS Panorama bekerja dengan IAM 162
Contoh kebijakan berbasis identitas 162
Kebijakan terkelola AWS 165
Menggunakan peran terkait layanan 167
Cross-service bingung wakil pencegahan 169
Memecahkan masalah 170
Validasi kepatuhan 173
Pertimbangan tambahan saat orang hadir 174
Keamanan infrastruktur 175
Menerapkan AWS Panorama Appliance di pusat data Anda 175
Lingkungan runtime 177
Rilis 178
..... clxxxiv

Apakah AWS Panorama itu?

AWS Panoramaadalah layanan yang membawa visi komputer ke jaringan kamera lokal Anda. Anda menginstalAWS PanoramaAppliance atau perangkat lain yang kompatibel di pusat data Anda, daftarkanAWS Panorama, dan menyebarkan aplikasi visi komputer dari cloud.AWS Panoramabekerja dengan kamera jaringan real time streaming protocol (RTSP) yang ada. Alat ini menjalankan aplikasi penglihatan komputer yang aman dariAWSMitra, atau aplikasi yang Anda bangun sendiri denganAWS PanoramaAplikasi SDK.

ParameterAWS PanoramaAppliance adalah alat tepi kompak yang menggunakan kuat system-on-module (SOM) yang dioptimalkan untuk beban kerja machine learning. Alat ini dapat menjalankan beberapa model penglihatan komputer terhadap beberapa aliran video secara parallel dan menampilkan hasilnya secara real time. Ini dirancang untuk digunakan dalam pengaturan komersial dan industri dan dinilai untuk perlindungan debu dan cairan (IP-62).

ParameterAWS PanoramaAppliance memungkinkan Anda menjalankan aplikasi computer vision mandiri di edge, tanpa mengirim gambar ke AWS Cloud. Dengan menggunakan AWS SDK, Anda dapat berintegrasi dengan layanan AWS lain dan menggunakannya untuk melacak data dari aplikasi dari waktu ke waktu. Dengan mengintegrasikan dengan layanan AWS lainnya, Anda dapat menggunakanAWS Panoramauntuk melakukan hal berikut:

- Analisis pola lalu lintas— Gunakan AWS SDK untuk merekam data untuk analitik ritel di Amazon DynamoDB. Gunakan aplikasi tanpa server untuk menganalisis data yang dikumpulkan dari waktu ke waktu, mendeteksi anomali dalam data, dan memprediksi perilaku di future.
- Menerima peringatan keamanan situs- Pantau area terlarang di lokasi industri. Saat aplikasi Anda mendeteksi situasi yang berpotensi tidak aman, unggah gambar ke Amazon Simple Storage Service (Amazon S3) dan kirim notifikasi ke topik Amazon Simple Notification Service (Amazon SNS) sehingga penerima dapat mengambil tindakan korektif.
- Tingkatkan kontrol kualitas- Pantau output jalur perakitan untuk mengidentifikasi bagian yang tidak sesuai dengan persyaratan. Sorot gambar bagian yang tidak sesuai dengan teks dan kotak pembatas dan tampilkan di monitor untuk ditinjau oleh tim kontrol kualitas Anda.
- Kumpulkan data pelatihan dan uji- Unggah gambar objek yang tidak dapat diidentifikasi oleh model penglihatan komputer Anda, atau di mana kepercayaan model dalam tebakannya adalah garis batas. Gunakan aplikasi tanpa server untuk membuat antrian gambar yang perlu diberi tag. Tandai gambar dan gunakan untuk melatih ulang model di Amazon SageMaker.

AWS Panoramamenggunakan layanan AWS lain untuk mengelolaAWS PanoramaAppliance, model akses dan kode, dan menyebarkan aplikasi.AWS Panoramamelakukan sebanyak mungkin tanpa mengharuskan Anda untuk berinteraksi dengan layanan lain, tetapi pengetahuan tentang layanan berikut dapat membantu Anda memahami caranyaAWS Panoramaworks.

- SageMaker— Anda dapat menggunakan SageMaker untuk mengumpulkan data pelatihan dari kamera atau sensor, membangun model pembelajaran mesin, dan melatihnya untuk penglihatan komputer.AWS Panoramamenggunakan SageMaker Neo untuk mengoptimalkan model untuk berjalan diAWS PanoramaAlat
- Amazon S3— Anda menggunakan titik akses Amazon S3 untuk menampilkan kode aplikasi, model, dan file konfigurasi untuk penerapan keAWS PanoramaAlat
- AWS IoT—AWS PanoramamenggunakanAWS IoTlayanan untuk memantau keadaanAWS PanoramaAppliance, mengelola pembaruan perangkat lunak, dan menyebarkan aplikasi. Anda tidak perlu menggunakanAWS IoTsecara langsung

Untuk memulai denganAWS PanoramaAlat dan pelajari selengkapnya tentang layanan ini, lanjutkanMemulai dengan AWS Panorama.

Memulai dengan AWS Panorama

Untuk memulaiAWS Panorama, pertama-tama pelajari konsep layanan dan terminologi yang digunakan dalam panduan ini. Kemudian Anda dapat menggunakanAWS Panorama konsol untuk mendaftarkanAWS Panorama Appliance Anda dan membuat aplikasi. Dalam waktu sekitar satu jam, Anda dapat mengonfigurasi perangkat, memperbarui perangkat lunaknya, dan menerapkan aplikasi sampel. Untuk menyelesaikan tutorial di bagian ini, Anda menggunakanAWS Panorama Appliance dan kamera yang mengalirkan video melalui jaringan lokal.

> **Note**
>
> Untuk membeliAWS Panorama Appliance, kunjungi AWS Panoramakonsol.

AplikasiAWS Panorama sampel menunjukkan penggunaanAWS Panorama fitur. Ini mencakup model yang telah dilatih SageMaker dan kode contoh yang menggunakan SDKAWS Panorama Aplikasi untuk menjalankan inferensi dan output video. Contoh aplikasi menyertakanAWS CloudFormation template dan skrip yang menunjukkan cara mengotomatiskan alur kerja pengembangan dan penyebaran dari baris perintah.

Dua topik terakhir dalam pasal ini merinci persyaratan untuk model dan kamera, dan spesifikasi perangkat kerasAWS Panorama Appliance. Jika Anda belum mendapatkan alat dan kamera, atau berencana mengembangkan model visi komputer Anda sendiri, lihat topik ini terlebih dahulu untuk informasi lebih lanjut.

Topik

- Konsep AWS Panorama
- Menyiapkan AWS Panorama
- Menerapkan aplikasi sampel AWS Panorama
- Mengembangkan aplikasi AWS Panorama
- Model dan kamera visi komputer yang didukung
- Spesifikasi Peralatan AWS Panorama
- Kuota layanan

Konsep AWS Panorama

Di AWS Panorama, Anda membuat aplikasi visi komputer dan menerapkannya ke AWS Panorama Appliance atau perangkat yang kompatibel untuk menganalisis aliran video dari kamera jaringan. Anda menulis kode aplikasi dengan Python dan membangun wadah aplikasi dengan Docker. Anda menggunakan AWS Panorama Application CLI untuk mengimpor model machine learning secara lokal atau dari Amazon Simple Storage Service (Amazon S3). Aplikasi menggunakan AWS Panorama Application SDK untuk menerima input video dari kamera dan berinteraksi dengan model.

Konsep

- Alat Panorama AWS
- Perangkat yang kompatibel
- Aplikasi
- Simpul
- Model

Alat Panorama AWS

AWS Panorama Appliance adalah perangkat keras yang menjalankan aplikasi Anda. Anda menggunakan konsol AWS Panorama untuk mendaftarkan alat, memperbarui perangkat lunaknya, dan menyebarkan aplikasi ke dalamnya. Perangkat lunak pada AWS Panorama Appliance terhubung ke aliran kamera, mengirimkan bingkai video ke aplikasi Anda, dan menampilkan output video pada tampilan terlampir.

AWS Panorama Appliance adalah sebuahperangkat tepi didukung oleh Nvidia Jetson AGX Xavier. Alih-alih mengirim gambar keAWSCloud untuk diproses, ia menjalankan aplikasi secara lokal pada perangkat keras yang dioptimalkan. Ini memungkinkan Anda menganalisis video secara real time dan memproses hasilnya secara lokal. Alat ini memerlukan koneksi internet untuk melaporkan statusnya, mengunggah log, dan melakukan pembaruan dan penerapan perangkat lunak.

Untuk informasi selengkapnya, lihat MengelolaAWS PanoramaAlat.

Perangkat yang kompatibel

Selain AWS Panorama Appliance, AWS Panorama mendukung perangkat yang kompatibel dariAWSMitra. Perangkat yang kompatibel mendukung fitur yang sama dengan AWS Panorama

Appliance. Anda mendaftarkan dan mengelola perangkat yang kompatibel dengan konsol dan API AWS Panorama, serta membuat serta menerapkan aplikasi dengan cara yang sama.

- Lenovo ThinkEdge® SE70— Didukung oleh Nvidia Jetson Xavier NX

Konten dan contoh aplikasi dalam panduan ini dikembangkan dengan AWS Panorama Appliance. Untuk informasi selengkapnya tentang fitur perangkat keras dan perangkat lunak tertentu untuk perangkat Anda, lihat dokumentasi pabrikan.

Aplikasi

Aplikasi berjalan di AWS Panorama Appliance untuk melakukan tugas visi komputer pada aliran video. Anda dapat membangun aplikasi visi komputer dengan menggabungkan kode Python dan model pembelajaran mesin, dan menerapkannya ke AWS Panorama Appliance melalui internet. Aplikasi dapat mengirim video ke layar, atau menggunakan AWS SDK untuk mengirim hasil ke layanan AWS.

Untuk membangun dan menerapkan aplikasi, Anda menggunakan AWS Panorama Application CLI. AWS Panorama Application CLI adalah alat baris perintah yang menghasilkan folder aplikasi default dan file konfigurasi, membangun kontainer dengan Docker, dan mengunggah aset. Anda dapat menjalankan beberapa aplikasi pada satu perangkat.

Untuk informasi selengkapnya, lihat MengelolaAWS Panoramaaplikasi.

Simpul

Sebuah aplikasi terdiri dari beberapa komponen yang disebutnode, yang mewakili input, output, model, dan kode. Sebuah node dapat berupa konfigurasi saja (input dan output), atau menyertakan artefak (model dan kode). Node kode aplikasi dibundel dalampaket nodeyang Anda unggah ke titik akses Amazon S3, tempat AWS Panorama Appliance dapat mengaksesnya. Sebuahmanifes aplikasiadalah file konfigurasi yang mendefinisikan koneksi antara node.

Untuk informasi selengkapnya, lihat Node aplikasi.

Model

Model visi komputer adalah jaringan pembelajaran mesin yang dilatih untuk memproses gambar. Model visi komputer dapat melakukan berbagai tugas seperti klasifikasi, deteksi, segmentasi, dan

pelacakan. Model visi komputer mengambil gambar sebagai input dan output informasi tentang gambar atau objek dalam gambar.

AWS Panorama mendukung model yang dibangun dengan PyTorch, Apache MXNet, dan TensorFlow. Anda dapat membuat model dengan Amazon SageMaker atau di lingkungan pengembangan Anda. Untuk informasi selengkapnya, lihat ???.

Menyiapkan AWS Panorama

Untuk mulai menggunakan AWS Panorama Appliance atau perangkat yang kompatibel, daftarkan di konsol AWS Panorama dan perbarui perangkat lunaknya. Selama proses penyiapan, Anda membuat sumber daya alat di AWS Panorama yang mewakili alat fisik, dan menyalin file ke alat dengan drive USB. Alat ini menggunakan sertifikat dan file konfigurasi ini untuk terhubung ke layanan AWS Panorama. Kemudian Anda menggunakan konsol AWS Panorama untuk memperbarui perangkat lunak alat dan mendaftarkan kamera.

Bagian

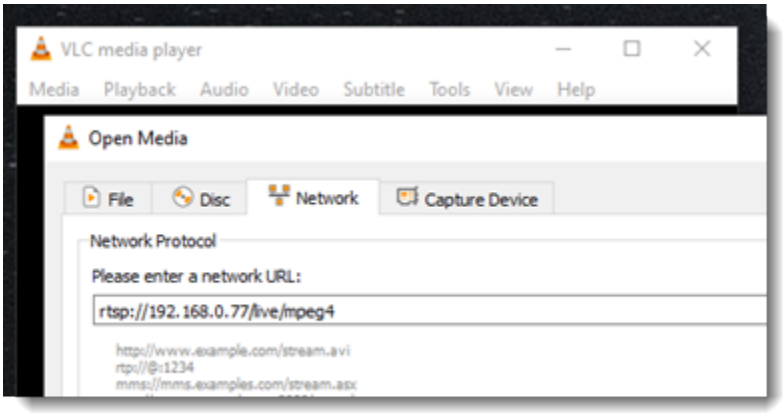
- Prasyarat
- Daftarkan dan konfigurasikan AWS Panorama Appliance
- Tingkatkan perangkat lunak alat
- Menambahkan stream kamera
- Langkah selanjutnya

Prasyarat

Untuk mengikuti tutorial ini, Anda memerlukan AWS Panorama Appliance atau perangkat yang kompatibel dan perangkat keras berikut:

- Tampilan - Tampilan dengan input HDMI untuk melihat output aplikasi sampel.
- Drive USB (disertakan dengan AWS Panorama Appliance) — Drive memori flash USB 3.0 berformat FAT32 dengan penyimpanan minimal 1 GB, untuk mentransfer arsip dengan file konfigurasi dan sertifikat ke AWS Panorama Appliance.
- Kamera - Kamera IP yang mengeluarkan aliran video RTSP.

Gunakan alat dan instruksi yang disediakan oleh produsen kamera Anda untuk mengidentifikasi alamat IP kamera dan jalur streaming. Anda dapat menggunakan pemutar video seperti VLC untuk memverifikasi URL aliran, dengan membukanya sebagai sumber media jaringan:



Konsol AWS Panorama menggunakan layanan AWS lain untuk merakit komponen aplikasi, mengelola izin, dan memverifikasi pengaturan. Untuk mendaftarkan alat dan menerapkan aplikasi sampel, Anda memerlukan izin berikut:

- AWSPanoramaFullAccess— Menyediakan akses penuh ke AWS Panorama, titik akses AWS Panorama di Amazon S3, kredensi alatAWS Secrets Manager, dan log alat di Amazon. CloudWatch Termasuk izin untuk membuat peran terkait layanan untuk AWS Panorama.
- AWS Identity and Access Management(IAM) — Pada proses pertama, untuk membuat peran yang digunakan oleh layanan AWS Panorama dan AWS Panorama Appliance.

Jika Anda tidak memiliki izin untuk membuat peran dalam IAM, minta administrator membuka konsol AWS Panorama dan menerima prompt untuk membuat peran layanan.

Daftarkan dan konfigurasikan AWS Panorama Appliance

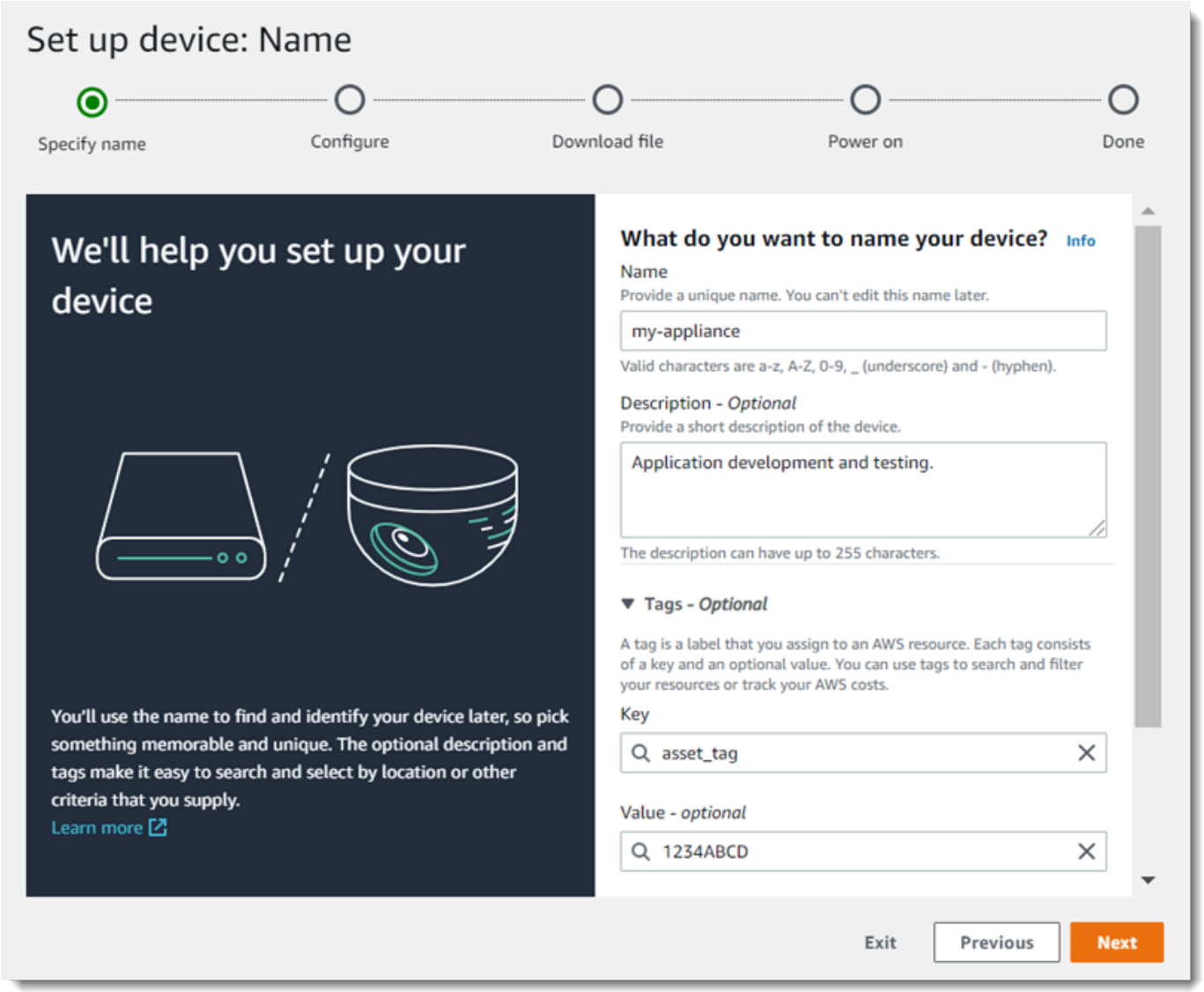
AWS Panorama Appliance adalah perangkat keras yang terhubung ke kamera berkemampuan jaringan melalui koneksi jaringan lokal. Ini menggunakan sistem operasi berbasis Linux yang mencakup AWS Panorama Application SDK dan perangkat lunak pendukung untuk menjalankan aplikasi visi komputer.

Untuk terhubung ke AWS manajemen alat dan penerapan aplikasi, alat menggunakan sertifikat perangkat. Anda menggunakan konsol AWS Panorama untuk menghasilkan sertifikat penyediaan. Alat menggunakan sertifikat sementara ini untuk menyelesaikan penyiapan awal dan mengunduh sertifikat perangkat permanen.

> **Important**
>
> Sertifikat penyediaan yang Anda hasilkan dalam prosedur ini hanya berlaku selama 5 menit. Jika Anda tidak menyelesaikan proses pendaftaran dalam jangka waktu ini, Anda harus memulai kembali.

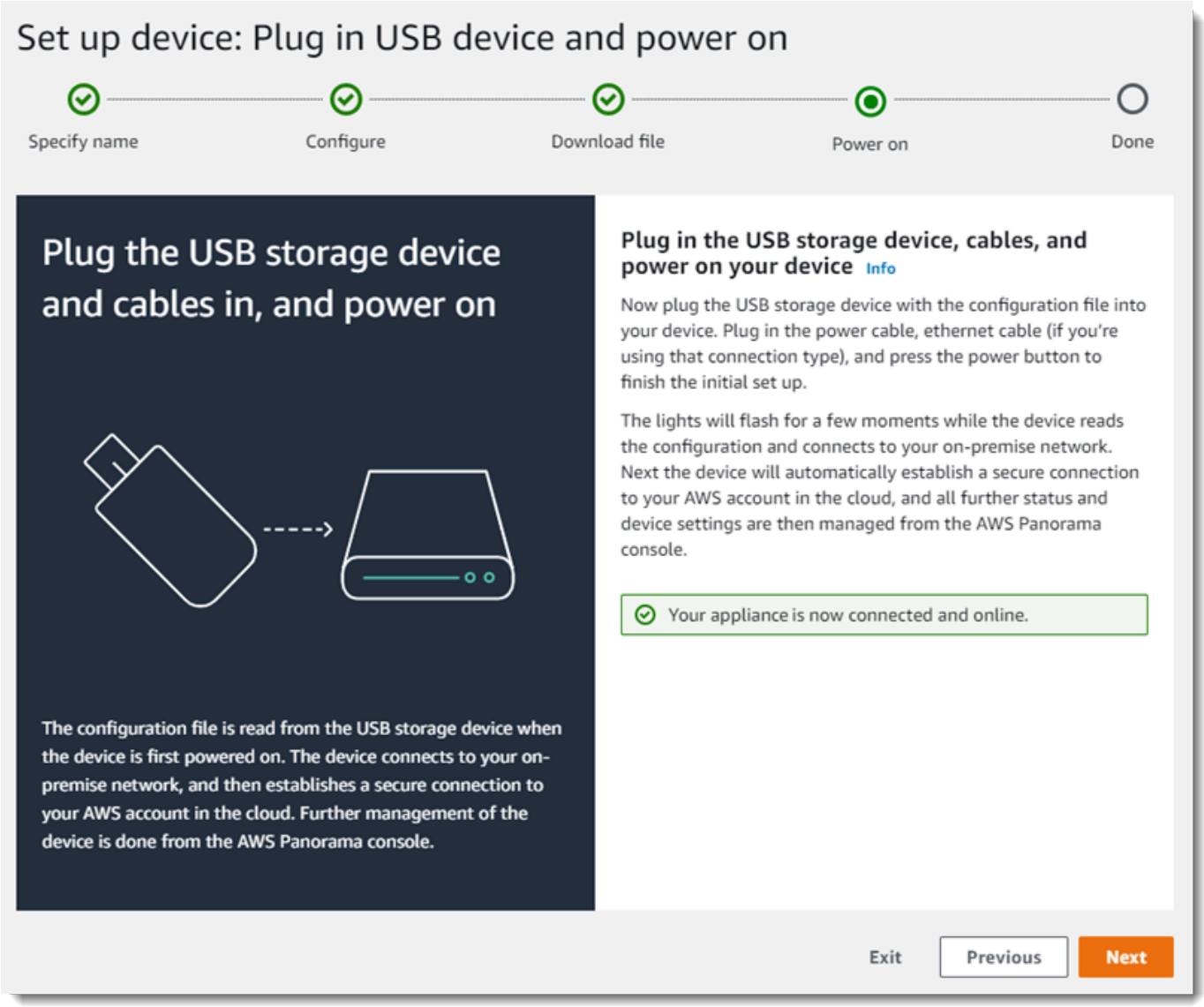
Untuk mendaftarkan alat

1. Connect drive USB ke komputer Anda. Siapkan alat dengan menghubungkan jaringan dan kabel listrik. Alat menyala dan menunggu drive USB dihubungkan.
2. Buka halaman AWS Panorama Console Memulai.
3. Pilih Tambah perangkat.
4. Pilih Mulai penyiapan.
5. Masukkan nama dan deskripsi untuk sumber daya perangkat yang mewakili AWS Panorama. Pilih Selanjutnya



6. Jika Anda perlu menetapkan alamat IP, server NTP, atau pengaturan DNS secara manual, pilih Pengaturan jaringan lanjutan. Jika tidak, pilih Selanjutnya.
7. Pilih Unduh arsip. Pilih Selanjutnya.
8. Salin arsip konfigurasi ke direktori root drive USB.
9. Connect drive USB ke port USB 3.0 di bagian depan alat, di sebelah port HDMI.

   Saat Anda menghubungkan drive USB, alat akan menyalin arsip konfigurasi dan file konfigurasi jaringan ke dirinya sendiri dan terhubung ke AWS Cloud. Lampu status alat berubah dari hijau menjadi biru saat menyelesaikan koneksi, dan kemudian kembali ke hijau.
10. Untuk melanjutkan, pilih Berikutnya.



11. PilihSelesai.

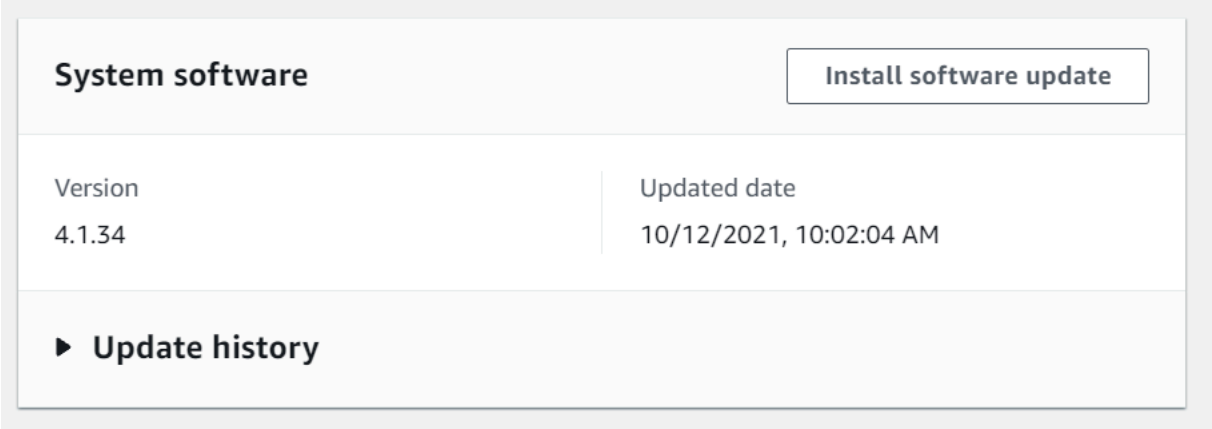
Tingkatkan perangkat lunak alat

AWS Panorama Appliance memiliki beberapa komponen perangkat lunak, termasuk sistem operasi Linux, SDK aplikasi AWS Panorama, dan mendukung library dan kerangka kerja visi komputer. Untuk memastikan bahwa Anda dapat menggunakan fitur dan aplikasi terbaru dengan alat Anda, tingkatkan perangkat lunaknya setelah pengaturan dan kapan pun pembaruan tersedia.

Untuk memperbarui perangkat lunak alat

1. Buka halaman Perangkat konsol AWS Panorama.

2. Pilih alat.
3. Pilih Pengaturan
4. Di bawah Perangkat lunak sistem, pilih Instal pembaruan perangkat lunak.



5. Pilih versi baru dan kemudian pilih Instal.

> **Important**
>
> Sebelum melanjutkan, lepaskan drive USB dari alat dan format untuk menghapus isinya. Arsip konfigurasi berisi data sensitif dan tidak dihapus secara otomatis.

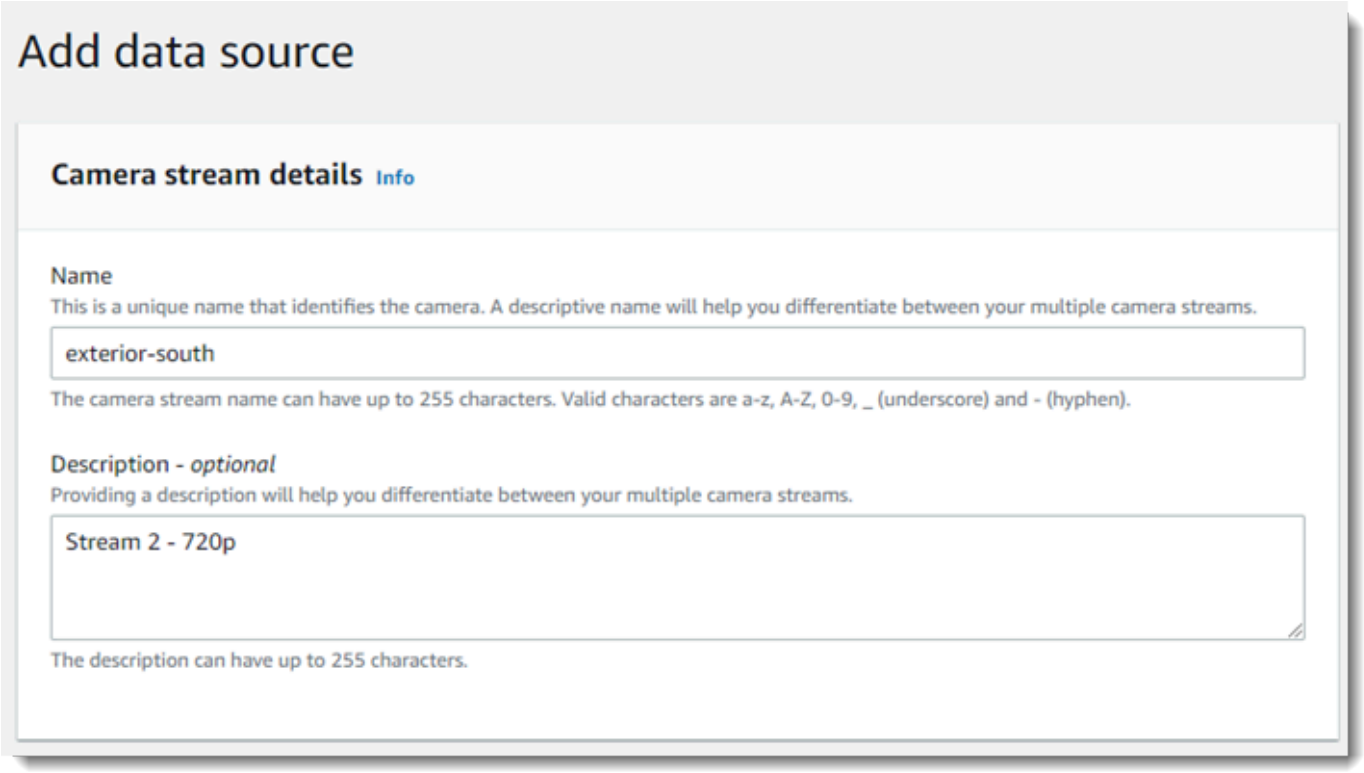
Proses upgrade dapat memakan waktu 30 menit atau lebih. Anda dapat memantau kemajuannya di konsol AWS Panorama atau pada monitor yang terhubung. Saat proses selesai, alat reboot.

Menambahkan stream kamera

Selanjutnya, daftarkan stream kamera dengan konsol AWS Panorama.

Untuk mendaftarkan stream kamera

1. Buka halaman Sumber data konsol AWS Panorama.
2. Memilih Tambahkan sumber data.



3. Konfigurasikan pengaturan berikut.

   - Nama - Nama untuk aliran kamera.
   - Deskripsi - Deskripsi singkat tentang kamera, lokasinya, atau detail lainnya.
   - RTSP URL - URL yang menentukan alamat IP kamera dan jalur ke aliran. Misalnya, `rtsp://192.168.0.77/live/mpeg4/`
   - Kredensi - Jika aliran kamera dilindungi kata sandinya.
4. Pilih Save (Simpan).

AWS Panorama menyimpan kredensi kamera Anda dengan aman. AWS Secrets Manager Beberapa aplikasi dapat memproses aliran kamera yang sama secara bersamaan.

Langkah selanjutnya

Jika Anda mengalami kesalahan saat penyiapan, lihatPemecahan Masalah.

Untuk menyebarkan aplikasi sampel, lanjutkan ke topik berikutnya.

Menerapkan aplikasi sampel AWS Panorama

Setelah Andamenyiapkan AWS Panorama Appliance atau perangkat yang kompatibeldan meningkatkan perangkat lunaknya, menyebarkan aplikasi sampel. Di bagian berikut, Anda mengimpor contoh aplikasi dengan AWS Panorama Application CLI dan menerapkannya dengan konsol AWS Panorama.

Aplikasi sampel menggunakan model pembelajaran mesin untuk mengklasifikasikan objek dalam bingkai video dari kamera jaringan. Ini menggunakan AWS Panorama Application SDK untuk memuat model, mendapatkan gambar, dan menjalankan model. Aplikasi kemudian melapisi hasil di atas video asli dan mengeluarkannya ke tampilan yang terhubung.

Dalam pengaturan ritel, menganalisis pola lalu lintas pejalan kaki memungkinkan Anda memprediksi tingkat lalu lintas. Dengan menggabungkan analisis dengan data lain, Anda dapat merencanakan peningkatan kebutuhan staf di sekitar hari libur dan acara lainnya, mengukur efektivitas iklan dan promosi penjualan, atau mengoptimalkan penempatan tampilan dan manajemen inventaris.

Bagian

- Prasyarat
- Impor aplikasi
- Deploy aplikasi
- Lihat Output
- Aktifkan SDK for Python
- Bersihkan
- Langkah selanjutnya

Prasyarat

Untuk mengikuti prosedur dalam tutorial ini, Anda memerlukan terminal atau shell baris perintah untuk menjalankan perintah. Dalam daftar kode, perintah didahului dengan simbol prompt ($) dan nama direktori saat ini, bila perlu.

```
~/panorama-project$ this is a command
this is output
```

Untuk perintah panjang, kita menggunakan karakter escape (\) untuk membagi perintah melalui beberapa baris.

Di Linux dan macOS, gunakan shell dan manajer paket pilihan Anda. Di Windows 10, Anda dapat menginstal Windows Subsystem for Linux untuk mendapatkan Ubuntu dan Bash versi terintegrasi Windows. Untuk bantuan menyiapkan lingkungan pengembangan di Windows, lihatMenyiapkan lingkungan pengembangan di Windows.

Anda menggunakan Python untuk mengembangkan aplikasi AWS Panorama dan menginstal alat dengan pip, manajer paket Python. Jika Anda belum memiliki Python,instal versi terbaru. Jika Anda memiliki Python 3 tetapi tidak pip, instal pip dengan manajer paket sistem operasi Anda, atau instal versi baru Python, yang dilengkapi dengan pip.

Dalam tutorial ini, Anda menggunakan Docker untuk membangun wadah yang menjalankan kode aplikasi Anda. Instal Docker dari situs web Docker: Dapatkan Docker

Tutorial ini menggunakan AWS Panorama Application CLI untuk mengimpor contoh aplikasi, membangun paket, dan mengunggah artefak. CLI Aplikasi AWS Panorama menggunakanAWS Command Line Interface(AWS CLI) untuk memanggil operasi API layanan. Jika Anda sudah memilikiAWS CLI, tingkatkan ke versi terbaru. Untuk menginstal CLI Aplikasi AWS Panorama danAWS CLI, gunakan`pip`.

```
$ pip3 install --upgrade awscli panoramacli
```

Unduh contoh aplikasi, dan ekstrak ke ruang kerja Anda.

- Aplikasi sampel—aws-panorama-sample.zip

Impor aplikasi

Untuk mengimpor contoh aplikasi untuk digunakan di akun Anda, gunakan AWS Panorama Application CLI. Folder dan manifes aplikasi berisi referensi ke nomor akun placeholder. Untuk memperbaruinya dengan nomor akun Anda, jalankan`panorama-cli import-application`perintah.

```
aws-panorama-sample$ panorama-cli import-application
```

Klaster`SAMPLE_CODE`paket, di`packages`direktori, berisi kode aplikasi dan konfigurasi, termasuk Dockerfile yang menggunakan gambar dasar aplikasi,`panorama-application`. Untuk membangun wadah aplikasi yang berjalan pada alat, gunakan`panorama-cli build-container`perintah.

```
aws-panorama-sample$ ACCOUNT_ID=$(aws sts get-caller-identity --output text --query
 'Account')
aws-panorama-sample$ panorama-cli build-container --container-asset-name code_asset --
package-path packages/${ACCOUNT_ID}-SAMPLE_CODE-1.0
```

Langkah terakhir dengan AWS Panorama Application CLI adalah mendaftarkan kode aplikasi dan node model, dan mengunggah aset ke titik akses Amazon S3 yang disediakan oleh layanan. Aset termasuk gambar kontainer kode, model, dan file deskriptor untuk masing-masing. Untuk mendaftarkan node dan mengunggah aset, jalankan`panorama-cli package-application`perintah.

```
aws-panorama-sample$ panorama-cli package-application
Uploading package model
Registered model with patch version
 bc9c58bd6f83743f26aa347dc86bfc3dd2451b18f964a6de2cc4570cb6f891f9
Uploading package code
Registered code with patch version
 11fd7001cb31ea63df6aaed297d600a5ecf641a987044a0c273c78ceb3d5d806
```

Deploy aplikasi

Gunakan konsol AWS Panorama untuk menerapkan aplikasi ke perangkat Anda.

Untuk men-deploy aplikasi

1. Buka konsol AWS PanoramaHalaman aplikasi yang di-deploy.
2. PilihTerapkan.
3. Tempelkan isi dari manifes aplikasi,`graphs/aws-panorama-sample/graph.json`, ke dalam editor teks. Pilih Selanjutnya.
4. Untuk Application name (Nama aplikasi), masukkan `aws-panorama-sample`.
5. PilihLanjut ke.
6. PilihMulai deployment.
7. PilihSelanjutnyatanpa memilih peran.
8. PilihPilih perangkat, dan kemudian pilih alat Anda. Pilih Selanjutnya.
9. PadaPilihlangkah, pilihLihat, dan tambahkan aliran kamera Anda sebagai sumber data. Pilih Selanjutnya.

10. PadaKonfigurasilangkah, pilihSelanjutnya.
11. PilihDeploy, dan kemudian pilihSelesai.
12. Dalam daftar aplikasi yang dikerahkan, pilihaws-panorama-sample.

Segarkan halaman ini untuk pembaruan, atau gunakan skrip berikut untuk memantau penyebaran dari baris perintah.

Example monitor-deployment.sh

```
while true; do
  aws panorama list-application-instances --query 'ApplicationInstances[?Name==`aws-
panorama-sample`]'
  sleep 10
done
```

```
[
    {
        "Name": "aws-panorama-sample",
        "ApplicationInstanceId": "applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "DefaultRuntimeContextDeviceName": "my-appliance",
        "Status": "DEPLOYMENT_PENDING",
        "HealthStatus": "NOT_AVAILABLE",
        "StatusDescription": "Deployment Workflow has been scheduled.",
        "CreatedTime": 1630010747.443,
        "Arn": "arn:aws:panorama:us-west-2:123456789012:applicationInstance/
applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "Tags": {}
    }
]
[
    {
        "Name": "aws-panorama-sample",
        "ApplicationInstanceId": "applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "DefaultRuntimeContextDeviceName": "my-appliance",
        "Status": "DEPLOYMENT_PENDING",
        "HealthStatus": "NOT_AVAILABLE",
        "StatusDescription": "Deployment Workflow has completed data validation.",
        "CreatedTime": 1630010747.443,
        "Arn": "arn:aws:panorama:us-west-2:123456789012:applicationInstance/
applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "Tags": {}
```

```
    }
]
...
```

Jika aplikasi tidak mulai berjalan, periksaaplikasi dan perangkat logdi Amazon CloudWatch Log.

Lihat Output

Saat penyebaran selesai, aplikasi mulai memproses aliran video dan mengirim log ke CloudWatch.

Untuk melihat log CloudWatch Beberapa catatan

1. BukaLog halaman grup dari CloudWatch Konsol log.
2. Temukan log aplikasi dan alat AWS Panorama dalam grup berikut:

   - Log perangkat—`/aws/panorama/devices/device-id`
   - Log aplikasi—`/aws/panorama/devices/device-id/applications/instance-id`

```
2022-08-26 17:43:39 INFO     INITIALIZING APPLICATION
2022-08-26 17:43:39 INFO     ## ENVIRONMENT VARIABLES
{'PATH': '/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin', 'TERM':
 'xterm', 'container': 'podman'...}
2022-08-26 17:43:39 INFO     Configuring parameters.
2022-08-26 17:43:39 INFO     Configuring AWS SDK for Python.
2022-08-26 17:43:39 INFO     Initialization complete.
2022-08-26 17:43:39 INFO     PROCESSING STREAMS
2022-08-26 17:46:19 INFO     epoch length: 160.183 s (0.936 FPS)
2022-08-26 17:46:19 INFO     avg inference time: 805.597 ms
2022-08-26 17:46:19 INFO     max inference time: 120023.984 ms
2022-08-26 17:46:19 INFO     avg frame processing time: 1065.129 ms
2022-08-26 17:46:19 INFO     max frame processing time: 149813.972 ms
2022-08-26 17:46:29 INFO     epoch length: 10.562 s (14.202 FPS)
2022-08-26 17:46:29 INFO     avg inference time: 7.185 ms
2022-08-26 17:46:29 INFO     max inference time: 15.693 ms
2022-08-26 17:46:29 INFO     avg frame processing time: 66.561 ms
2022-08-26 17:46:29 INFO     max frame processing time: 123.774 ms
```

Untuk melihat output video aplikasi, sambungkan alat ke monitor dengan kabel HDMI. Secara default, aplikasi menunjukkan hasil klasifikasi yang memiliki kepercayaan lebih dari 20%.

Example squeezenet_classes.json

```
["tench", "goldfish", "great white shark", "tiger shark",
"hammerhead", "electric ray", "stingray", "cock", "hen", "ostrich",
"brambling", "goldfinch", "house finch", "junco", "indigo bunting",
"robin", "bulbul", "jay", "magpie", "chickadee", "water ouzel",
"kite", "bald eagle", "vulture", "great grey owl",
"European fire salamander", "common newt", "eft",
"spotted salamander", "axolotl", "bullfrog", "tree frog",
...
```

Model sampel memiliki 1000 kelas termasuk banyak hewan, makanan, dan benda-benda umum. Coba arahkan kamera Anda ke keyboard atau cangkir kopi.

Untuk mempermudah, aplikasi sampel menggunakan model klasifikasi ringan. Model output array tunggal dengan probabilitas untuk masing-masing kelas. Aplikasi dunia nyata lebih sering

menggunakan model deteksi objek yang memiliki output multidimensi. Untuk contoh aplikasi dengan model yang lebih kompleks, lihatContoh aplikasi, skrip, dan templat.

Aktifkan SDK for Python

Aplikasi sampelAWS SDK for Python (Boto)mengirim metrik ke Amazon CloudWatch. Untuk mengaktifkan fungsionalitas ini, buat peran yang memberikan izin aplikasi untuk mengirim metrik, dan men-deploy ulang aplikasi dengan peran yang dilampirkan.

Aplikasi sampel termasukAWS CloudFormationtemplate yang menciptakan peran dengan izin yang dibutuhkan. Untuk membuat peran, gunakan`aws cloudformation deploy`perintah.

```
$ aws cloudformation deploy --template-file aws-panorama-sample.yml --stack-name aws-panorama-sample-runtime --capabilities CAPABILITY_NAMED_IAM
```

Untuk men-deploy aplikasi

1. Buka konsol AWS PanoramaHalaman aplikasi yang di-deploy.
2. Pilih aplikasi.
3. Pilih Ganti.
4. Selesaikan langkah-langkah untuk men-deploy aplikasi. DiTentukan peran IAMPilih peran yang Anda buat. Namanya dimulai dengan`aws-panorama-sample-runtime`.
5. Saat penyebaran selesai, bukaCloudWatchkonsoldan melihat metrik di`AWSPanoramaApplication`namespace Setiap 150 frame, aplikasi mencatat dan mengunggah metrik untuk pemrosesan bingkai dan waktu inferensi.

Bersihkan

Jika Anda selesai bekerja dengan aplikasi sampel, Anda dapat menggunakan konsol AWS Panorama untuk menghapusnya dari alat.

Untuk menghapus aplikasi dari alat

1. Buka konsol AWS PanoramaHalaman aplikasi yang di-deploy.
2. Pilih aplikasi.
3. PilihHapus dari perangkat.

Langkah selanjutnya

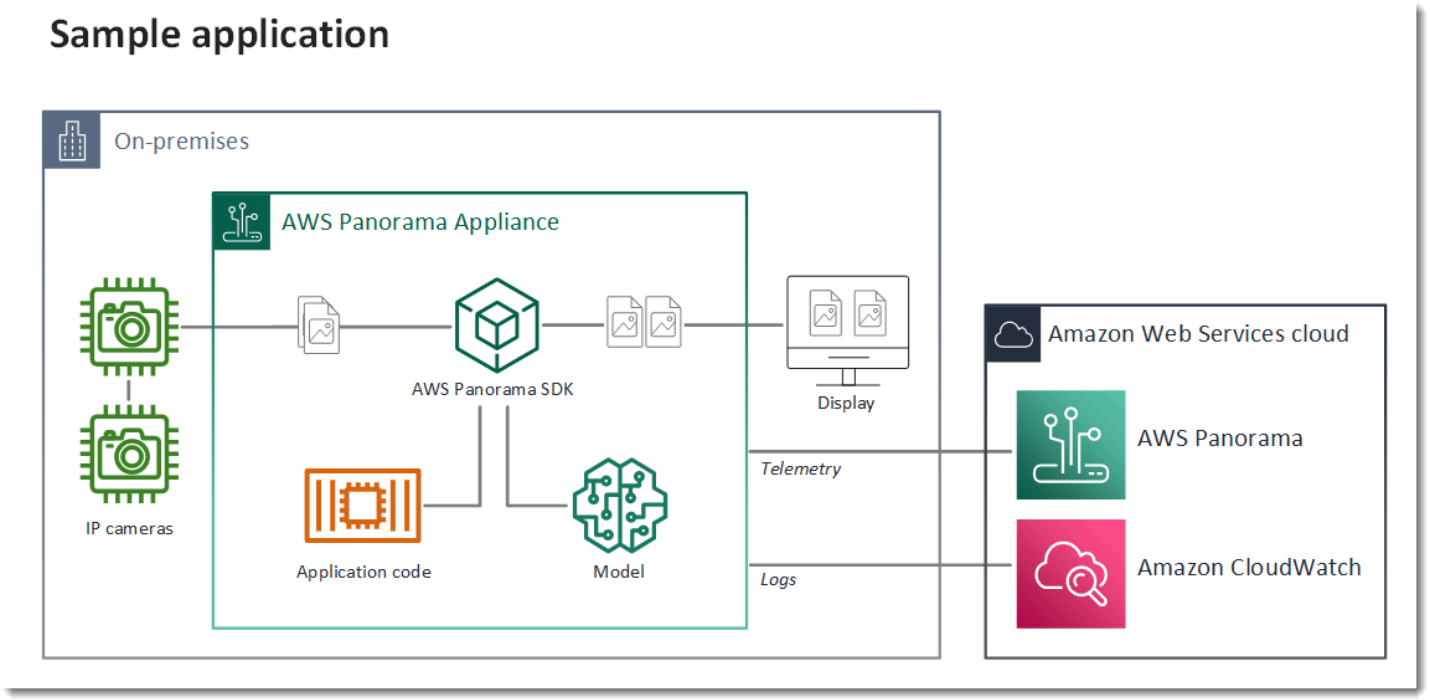
Jika Anda mengalami kesalahan saat men-deploy atau menjalankan aplikasi sampel, lihatPemecahan Masalah.

Untuk mempelajari lebih lanjut tentang fitur dan implementasi aplikasi contoh, lanjutkantopik selanjutnya.

Mengembangkan aplikasi AWS Panorama

Anda dapat menggunakan aplikasi sampel untuk mempelajari tentang struktur aplikasi AWS Panorama, dan sebagai titik awal untuk aplikasi Anda sendiri.

Diagram berikut menunjukkan komponen utama aplikasi yang berjalan pada AWS Panorama Appliance. Kode aplikasi menggunakan AWS Panorama Application SDK untuk mendapatkan gambar dan berinteraksi dengan model, yang tidak memiliki akses langsung ke. Aplikasi mengeluarkan video ke tampilan yang terhubung tetapi tidak mengirim data gambar di luar jaringan lokal Anda.



Dalam contoh ini, aplikasi menggunakan AWS Panorama Application SDK untuk mendapatkan bingkai video dari kamera, memproses data video terlebih dahulu, dan mengirim data ke model penglihatan komputer yang mendeteksi objek. Aplikasi menampilkan hasilnya pada layar HDMI yang terhubung ke alat.

Bagian

- Manifes aplikasi manifes
- Bangunan dengan aplikasi sampel
- Mengubah model visi komputer
- Gambar prapemrosesan awal

- Meng-upload metrik dengan SDK for Python
- Langkah selanjutnya

Manifes aplikasi manifes

Manifes aplikasi adalah file bernama`graph.json`di dalam`graphs`folder. Manifes mendefinisikan komponen aplikasi, yaitu paket, node, dan tepi.

Paket adalah kode, konfigurasi, dan file biner untuk kode aplikasi, model, kamera, dan tampilan. Aplikasi sampel menggunakan 4 paket:

Example **`graphs/aws-panorama-sample/graph.json`**— Paket - Paket:

```
    "packages": [
        {
            "name": "123456789012::SAMPLE_CODE",
            "version": "1.0"
        },
        {
            "name": "123456789012::SQUEEZENET_PYTORCH_V1",
            "version": "1.0"
        },
        {
            "name": "panorama::abstract_rtsp_media_source",
            "version": "1.0"
        },
        {
            "name": "panorama::hdmi_data_sink",
            "version": "1.0"
        }
    ],
```

Dua paket pertama didefinisikan dalam aplikasi, di`packages`direktori. Mereka berisi kode dan model khusus untuk aplikasi ini. Dua paket kedua adalah paket kamera dan tampilan generik yang disediakan oleh layanan AWS Panorama. Parameter`abstract_rtsp_media_source`paket adalah placeholder untuk kamera yang Anda menimpa selama penyebaran. Parameter`hdmi_data_sink`paket merupakan konektor output HDMI pada perangkat.

Node adalah antarmuka ke paket, serta parameter non-paket yang dapat memiliki nilai default yang Anda ganti pada waktu penyebaran. Kode dan model paket mendefinisikan antarmuka

di`package.json`file yang menentukan input dan output, yang dapat berupa aliran video atau tipe data dasar seperti float, boolean, atau string.

Misalnya,`code_node`node mengacu pada antarmuka dari`SAMPLE_CODE`paket.

```
        "nodes": [
            {
                "name": "code_node",
                "interface": "123456789012::SAMPLE_CODE.interface",
                "overridable": false,
                "launch": "onAppStart"
            },
```

Antarmuka ini didefinisikan dalam file konfigurasi paket,`package.json`. Antarmuka menentukan bahwa paket adalah logika bisnis dan bahwa dibutuhkan aliran video bernama`video_in`dan nomor floating point bernama`threshold`sebagai input. Antarmuka juga menentukan bahwa kode memerlukan buffer aliran video bernama`video_out`untuk menampilkan video ke tampilan

Example **`packages/123456789012-SAMPLE_CODE-1.0/package.json`**

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [],
        "interfaces": [
            {
                "name": "interface",
                "category": "business_logic",
                "asset": "code_asset",
                "inputs": [
                    {
                        "name": "video_in",
                        "type": "media"
                    },
                    {
                        "name": "threshold",
                        "type": "float32"
                    }
                ],
                "outputs": [
```

```
                    {
                        "description": "Video stream output",
                        "name": "video_out",
                        "type": "media"
                    }
                ]
            }
        ]
    }
}
```

Kembali ke manifes aplikasi,`camera_node`node merupakan aliran video dari kamera. Ini termasuk dekorator yang muncul di konsol saat Anda menerapkan aplikasi, meminta Anda untuk memilih aliran kamera.

Example **`graphs/aws-panorama-sample/graph.json`**- Simpul kamera

```
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "launch": "onAppStart",
                "decorator": {
                    "title": "Camera",
                    "description": "Choose a camera stream."
                }
            },
```

Sebuah node parameter,`threshold_param`, mendefinisikan parameter ambang kepercayaan yang digunakan oleh kode aplikasi. Ini memiliki nilai default 60, dan dapat diganti selama penyebaran.

Example **`graphs/aws-panorama-sample/graph.json`**- Simpul parameter

```
            {
                "name": "threshold_param",
                "interface": "float32",
                "value": 60.0,
                "overridable": true,
                "decorator": {
                    "title": "Confidence threshold",
                    "description": "The minimum confidence for a classification to be
 recorded."
                }
```

```
        }
```

Bagian terakhir dari manifes aplikasi,`edges`, membuat koneksi antar node. Aliran video kamera dan parameter ambang terhubung ke input node kode, dan output video dari node kode terhubung ke tampilan.

Example **`graphs/aws-panorama-sample/graph.json`**— Edge

```
    "edges": [
        {
            "producer": "camera_node.video_out",
            "consumer": "code_node.video_in"
        },
        {
            "producer": "code_node.video_out",
            "consumer": "output_node.video_in"
        },
        {
            "producer": "threshold_param",
            "consumer": "code_node.threshold"
        }
    ]
```

Bangunan dengan aplikasi sampel

Anda dapat menggunakan aplikasi sampel sebagai titik awal untuk aplikasi Anda sendiri.

Nama masing-masing paket harus unik di akun Anda. Jika Anda dan pengguna lain di akun Anda, keduanya menggunakan nama paket umum seperti`codeatau`model, Anda mungkin mendapatkan versi paket yang salah saat Anda men-deploy. Ubah nama paket kode menjadi salah satu yang mewakili aplikasi Anda.

Untuk mengganti nama paket kode

1. Ganti nama folder paket:`packages/123456789012-`*`SAMPLE_CODE`*`-1.0/`.
2. Perbarui nama paket di lokasi berikut.

   - Manifes aplikasi aplikasi—`graphs/aws-panorama-sample/graph.json`
   - Konfigurasi Package—`packages/123456789012-SAMPLE_CODE-1.0/package.json`
   - Bangunan skrip skrip—`3-build-container.sh`

Untuk memperbarui kode aplikasi

1. Memodifikasi kode aplikasi di`packages/123456789012-SAMPLE_CODE-1.0/src/application.py`.
2. Untuk membangun wadah, jalankan`3-build-container.sh`.

```
aws-panorama-sample$ ./3-build-container.sh
TMPDIR=$(pwd) docker build -t code_asset packages/123456789012-SAMPLE_CODE-1.0
Sending build context to Docker daemon  61.44kB
Step 1/2 : FROM public.ecr.aws/panorama/panorama-application
 ---> 9b197f256b48
Step 2/2 : COPY src /panorama
 ---> 55c35755e9d2
Successfully built 55c35755e9d2
Successfully tagged code_asset:latest
docker export --output=code_asset.tar $(docker create code_asset:latest)
gzip -9 code_asset.tar
Updating an existing asset with the same name
{
    "name": "code_asset",
    "implementations": [
        {
            "type": "container",
            "assetUri":
 "98aaxmpl1c1ef64cde5ac13bd3be5394e5d17064beccee963b4095d83083c343.tar.gz",
            "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
        }
    ]
}
Container asset for the package has been succesfully built at  ~/aws-panorama-
sample-dev/
assets/98aaxmpl1c1ef64cde5ac13bd3be5394e5d17064beccee963b4095d83083c343.tar.gz
```

   CLI secara otomatis menghapus aset kontainer lama dari`assets`folder dan memperbarui konfigurasi paket.
3. Untuk mengunggah paket, jalankan`4-package-application.py`.
4. Buka konsol AWS PanoramaHalaman aplikasi yang di-deploy aplikasi diterapkan.
5. Pilih aplikasi.
6. Pilih Ganti.

7. Selesaikan langkah-langkah untuk menerapkan aplikasi. Jika diperlukan, Anda dapat membuat perubahan pada manifes aplikasi, aliran kamera, atau parameter.

Mengubah model visi komputer

Aplikasi sampel termasuk model penglihatan komputer. Untuk menggunakan model Anda sendiri, ubah konfigurasi node model, dan gunakan AWS Panorama Application CLI untuk mengimpornya sebagai aset.

Contoh berikut menggunakan SSD MXNet ResNet50 model yang dapat Anda unduh dari panduan ini GitHub repo:ssd_512_resnet50_v1_voc.tar.gz

Untuk mengubah model aplikasi sampel

1. Ubah nama folder paket agar sesuai dengan model Anda. Misalnya, untuk`packages/123456789012-SSD_512_RESNET50_V1_VOC-1.0/`.
2. Perbarui nama paket di lokasi berikut.

   - Manifes aplikasi aplikasi—`graphs/aws-panorama-sample/graph.json`
   - Konfigurasi Package—`packages/123456789012-SSD_512_RESNET50_V1_VOC-1.0/package.json`
3. Dalam file konfigurasi paket (`package.json`). Mengubah`assets`nilai ke array kosong.

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SSD_512_RESNET50_V1_VOC",
        "version": "1.0",
        "description": "Compact classification model",
        "assets": [],
```

4. Buka file deskriptor paket (`descriptor.json`). Perbarui`framework`dan`shape`nilai yang sesuai dengan model Anda.

```
{
    "mlModelDescriptor": {
        "envelopeVersion": "2021-01-01",
        "framework": "MXNET",
        "inputs": [
```

```
            {
                "name": "data",
                "shape": [ 1, 3, 512, 512 ]
            }
        ]
    }
}
```

Nilai untukbentuk,`1,3,512,512`, menunjukkan jumlah gambar yang diambil model sebagai input (1), jumlah saluran di setiap gambar (3—merah, hijau, dan biru), dan dimensi gambar (512 x 512). Nilai-nilai dan urutan array bervariasi di antara model.

5. Impor model dengan AWS Panorama Application CLI. AWS Panorama Application CLI menyalin file model dan deskriptor ke`assets`folder dengan nama unik, dan memperbarui konfigurasi paket.

```
aws-panorama-sample$ panorama-cli add-raw-model --model-asset-name model-asset \
--model-local-path ssd_512_resnet50_v1_voc.tar.gz \
--descriptor-path packages/123456789012-SSD_512_RESNET50_V1_VOC-1.0/descriptor.json
 \
--packages-path packages/123456789012-SSD_512_RESNET50_V1_VOC-1.0
{
    "name": "model-asset",
    "implementations": [
        {
            "type": "model",
            "assetUri":
 "b1a1589afe449b346ff47375c284a1998c3e1522b418a7be8910414911784ce1.tar.gz",
            "descriptorUri":
 "a6a9508953f393f182f05f8beaa86b83325f4a535a5928580273e7fe26f79e78.json"
        }
    ]
}
```

6. Untuk mengunggah model, jalankan`panorama-cli package-application`.

```
$ panorama-cli package-application
Uploading package SAMPLE_CODE
Patch Version 1844d5a59150d33f6054b04bac527a1771fd2365e05f990ccd8444a5ab775809
 already registered, ignoring upload
Uploading package SSD_512_RESNET50_V1_VOC
Patch version for the package
 244a63c74d01e082ad012ebf21e67eef5d81ce0de4d6ad1ae2b69d0bc498c8fd
```

```
upload: assets/
b1a1589afe449b346ff47375c284a1998c3e1522b418a7be8910414911784ce1.tar.gz to
 s3://arn:aws:s3:us-west-2:454554846382:accesspoint/panorama-123456789012-
wc66m5eishf4si4sz5jefhx
63a/123456789012/nodePackages/SSD_512_RESNET50_V1_VOC/binaries/
b1a1589afe449b346ff47375c284a1998c3e1522b418a7be8910414911784ce1.tar.gz
upload: assets/
a6a9508953f393f182f05f8beaa86b83325f4a535a5928580273e7fe26f79e78.json to
 s3://arn:aws:s3:us-west-2:454554846382:accesspoint/panorama-123456789012-
wc66m5eishf4si4sz5jefhx63
a/123456789012/nodePackages/SSD_512_RESNET50_V1_VOC/binaries/
a6a9508953f393f182f05f8beaa86b83325f4a535a5928580273e7fe26f79e78.json
{
    "ETag": "\"2381dabba34f4bc0100c478e67e9ab5e\"",
    "ServerSideEncryption": "AES256",
    "VersionId": "KbY5fpESdpYamjWZ0YyGqHo3.LQQWUC2"
}
Registered SSD_512_RESNET50_V1_VOC with patch version
 244a63c74d01e082ad012ebf21e67eef5d81ce0de4d6ad1ae2b69d0bc498c8fd
Uploading package SQUEEZENET_PYTORCH_V1
Patch Version 568138c430e0345061bb36f05a04a1458ac834cd6f93bf18fdacdffb62685530
 already registered, ignoring upload
```

7. Perbarui kode aplikasi. Sebagian besar kode dapat digunakan kembali. Kode khusus untuk respons model ada di`process_results`metode.

```
    def process_results(self, inference_results, stream):
        """Processes output tensors from a computer vision model and annotates a
 video frame."""
        for class_tuple in inference_results:
            indexes = self.topk(class_tuple[0])
        for j in range(2):
            label = 'Class [%s], with probability %.3f.'%
 (self.classes[indexes[j]], class_tuple[0][indexes[j]])
            stream.add_label(label, 0.1, 0.25 + 0.1*j)
```

   Bergantung pada model Anda, Anda mungkin juga perlu memperbarui`preprocess`metode.

Gambar prapemrosesan awal

Sebelum aplikasi mengirimkan gambar ke model, ia mempersiapkannya untuk inferensi dengan mengubah ukurannya dan menormalkan data warna. Model yang digunakan aplikasi memerlukan

gambar 224 x 224 piksel dengan tiga saluran warna, agar sesuai dengan jumlah input di lapisan pertamanya. Aplikasi menyesuaikan setiap nilai warna dengan mengubahnya menjadi angka antara 0 dan 1, mengurangi nilai rata-rata untuk warna itu, dan membaginya dengan deviasi standar. Akhirnya, ini menggabungkan saluran warna dan mengubahnya menjadi a NumPy array yang model dapat memproses.

Example application.py— Prapemrosesan awal

```
    def preprocess(self, img, width):
        resized = cv2.resize(img, (width, width))
        mean = [0.485, 0.456, 0.406]
        std = [0.229, 0.224, 0.225]
        img = resized.astype(np.float32) / 255.
        img_a = img[:, :, 0]
        img_b = img[:, :, 1]
        img_c = img[:, :, 2]
        # Normalize data in each channel
        img_a = (img_a - mean[0]) / std[0]
        img_b = (img_b - mean[1]) / std[1]
        img_c = (img_c - mean[2]) / std[2]
        # Put the channels back together
        x1 = [[[], [], []]]
        x1[0][0] = img_a
        x1[0][1] = img_b
        x1[0][2] = img_c
        return np.asarray(x1)
```

Proses ini memberikan nilai model dalam rentang diprediksi berpusat di sekitar 0. Ini cocok dengan preprocessing yang diterapkan pada gambar dalam kumpulan data pelatihan, yang merupakan pendekatan standar tetapi dapat bervariasi per model.

Meng-upload metrik dengan SDK for Python

Contoh aplikasi menggunakan SDK untuk Python untuk mengunggah metrik ke Amazon CloudWatch.

Example application.py— SDK for Python

```
    def process_streams(self):
        """Processes one frame of video from one or more video streams."""
        ...
            logger.info('epoch length: {:.3f} s ({:.3f} FPS)'.format(epoch_time,
epoch_fps))
```

```
            logger.info('avg inference time: {:.3f} ms'.format(avg_inference_time))
            logger.info('max inference time: {:.3f} ms'.format(max_inference_time))
            logger.info('avg frame processing time: {:.3f}
ms'.format(avg_frame_processing_time))
            logger.info('max frame processing time: {:.3f}
ms'.format(max_frame_processing_time))
            self.inference_time_ms = 0
            self.inference_time_max = 0
            self.frame_time_ms = 0
            self.frame_time_max = 0
            self.epoch_start = time.time()
            self.put_metric_data('AverageInferenceTime', avg_inference_time)
            self.put_metric_data('AverageFrameProcessingTime',
avg_frame_processing_time)

    def put_metric_data(self, metric_name, metric_value):
        """Sends a performance metric to CloudWatch."""
        namespace = 'AWSPanoramaApplication'
        dimension_name = 'Application Name'
        dimension_value = 'aws-panorama-sample'
        try:
            metric = self.cloudwatch.Metric(namespace, metric_name)
            metric.put_data(
                Namespace=namespace,
                MetricData=[{
                    'MetricName': metric_name,
                    'Value': metric_value,
                    'Unit': 'Milliseconds',
                    'Dimensions': [
                        {
                            'Name': dimension_name,
                            'Value': dimension_value
                        },
                        {
                            'Name': 'Device ID',
                            'Value': self.device_id
                        }
                    ]
                }]
            )
            logger.info("Put data for metric %s.%s", namespace, metric_name)
        except ClientError:
            logger.warning("Couldn't put data for metric %s.%s", namespace,
metric_name)
```

```
        except AttributeError:
            logger.warning("CloudWatch client is not available.")
```

Ini mendapat izin dari peran runtime yang Anda tetapkan selama penyebaran. Peran didefinisikan dalam`aws-panorama-sample.yml` AWS CloudFormationtemplat yang tepat.

Example aws-panorama-sample.yml

```
Resources:
  runtimeRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - panorama.amazonaws.com
            Action:
              - sts:AssumeRole
      Policies:
        - PolicyName: cloudwatch-putmetrics
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action: 'cloudwatch:PutMetricData'
                Resource: '*'
      Path: /service-role/
```

Contoh aplikasi menginstal SDK untuk Python dan dependensi lainnya dengan pip. Ketika Anda membangun wadah aplikasi,`Dockerfile`menjalankan perintah untuk menginstal perpustakaan di atas apa yang datang dengan gambar dasar.

Example Dockerfile

```
FROM public.ecr.aws/panorama/panorama-application
WORKDIR /panorama
COPY . .
RUN pip install --no-cache-dir --upgrade pip && \
```

```
    pip install --no-cache-dir -r requirements.txt
```

Untuk menggunakanAWSSDK dalam kode aplikasi Anda, pertama-tama ubah template untuk menambahkan izin untuk semua tindakan API yang digunakan aplikasi. PerbaruiAWS CloudFormationstack dengan menjalankan`1-create-role.sh`setiap kali Anda membuat perubahan. Kemudian, terapkan perubahan pada kode aplikasi Anda.

Untuk tindakan yang memodifikasi atau menggunakan sumber daya yang ada, adalah praktik terbaik untuk meminimalkan cakupan kebijakan ini dengan menentukan nama atau pola untuk target`Resource`dalam pernyataan terpisah. Untuk detail tentang tindakan dan sumber daya yang didukung oleh setiap layanan, lihatTindakan, sumber daya, dan kunci kondisidalam Referensi Otorisasi Layanan

Langkah selanjutnya

Untuk petunjuk tentang penggunaan AWS Panorama Application CLI untuk membangun aplikasi dan membuat paket dari awal, lihat README CLI.

- github.com/id/aws/aws-panorama-cli

Untuk kode contoh lainnya dan utilitas pengujian yang dapat Anda gunakan untuk memvalidasi kode aplikasi Anda sebelum menerapkan, kunjungi repositori sampel AWS Panorama.

- github.com/aws-samples/aws-panorama-samples

Model dan kamera visi komputer yang didukung

AWS Panorama mendukung model yang dibuat dengan Apache MXNetPyTorch, dan. TensorFlow Saat Anda menerapkan aplikasi, AWS Panorama mengkompilasi model Anda di Neo. SageMaker Anda dapat membuat model di Amazon SageMaker atau di lingkungan pengembangan Anda, selama Anda menggunakan lapisan yang kompatibel dengan SageMaker Neo.

Untuk memproses video dan mendapatkan gambar yang akan dikirim ke model, AWS Panorama Appliance terhubung ke streaming video yang dikodekan H.264 dengan protokol RTSP. AWS Panorama menguji berbagai kamera umum untuk kompatibilitas.

Bagian

- model yang didukung
- Kamera yang didukung

model yang didukung

Saat membuat aplikasi untuk AWS Panorama, Anda menyediakan model pembelajaran mesin yang digunakan aplikasi untuk visi komputer. Anda dapat menggunakan model pra-bangun dan pra-terlatih yang disediakan oleh kerangka kerja model, model sampel, atau model yang Anda buat dan latih sendiri.

> **Note**
>
> Untuk daftar model pra-bangun yang telah diuji dengan AWS Panorama, lihat Kompatibilitas model.

Saat Anda menerapkan aplikasi, AWS Panorama menggunakan kompiler SageMaker Neo untuk mengompilasi model visi komputer Anda. SageMakerNeo adalah kompiler yang mengoptimalkan model agar berjalan secara efisien pada platform target, yang dapat berupa instans di Amazon Elastic Compute Cloud (Amazon EC2), atau perangkat edge seperti AWS Panorama Appliance.

AWS Panorama mendukung versiPyTorch, Apache MXNet, dan TensorFlow yang didukung untuk perangkat edge oleh Neo. SageMaker Ketika Anda membangun model Anda sendiri, Anda dapat menggunakan versi kerangka kerja yang tercantum dalam catatan rilis SageMaker Neo. DiSageMaker, Anda dapat menggunakan algoritma klasifikasi gambar bawaan.

Untuk informasi selengkapnya tentang menggunakan model di AWS Panorama, lihatModel visi komputer.

Kamera yang didukung

AWS Panorama Appliance mendukung aliran video H.264 dari kamera yang mengeluarkan RTSP melalui jaringan lokal. Untuk streaming kamera yang lebih besar dari 2 megapiksel, alat menskalakan gambar menjadi 1920x1080 piksel atau ukuran setara yang mempertahankan rasio aspek aliran.

Model kamera berikut telah diuji kompatibilitasnya dengan AWS Panorama Appliance:

- Sumbu — M3057-PLVE, M3058-PLVE, P1448-LE, P3225-LV Mk II
- LaView— LV-PB3040W
- Vivotek — IB9360-H
- Amcrest - IP2M-841B
- Aplikasi — IPC-B850W-S-3X, IPC-D250W-S
- WGCC - Kubah PoE 4MP ONVIF

Untuk spesifikasi perangkat keras alat, lihatSpesifikasi Peralatan AWS Panorama.

Spesifikasi Peralatan AWS Panorama

AWS Panorama Appliance memiliki spesifikasi perangkat keras berikut. Untuk lainnyaperangkat yang kompatibel, lihat dokumentasi pabrikan.

Komponen	Spesifikasi
Prosesor dan GPU	Nvidia Jetson AGX Xavierdengan RAM 32GB
eternet	2x 1000 Base-T (Gigabyte)
USB	1x USB 2.0 dan 1x USB 3.0 tipe-A perempuan
Output HDMI	2.0a
Dimensi	7,75 “x 9,6” x 1,6" (197mm x 243mm x 40mm)
Berat Badan	3.7lbs (1.7kg)
Catu daya	100V-240V 50-60Hz AC 65W
Masukan daya	Wadah IEC 60320 C6 (3-pin)
Debu dan perlindungan cairan	IP-62
Kepatuhan peraturan EMI/EMC	FCC Bagian-15 (AS)
Batas sentuh	IEC-62368
Suhu Operasi	-20° C sampai 60° C
Kelembaban pengoperasian	0% sampai 95% RH
Suhu penyimpanan	-20° C sampai 85° C
Kelembaban penyimpanan	Tidak terkontrol untuk suhu rendah. 90% RH pada suhu tinggi
Pendinginan	Ekstraksi panas udara paksa (fan)
Opsi pemasangan	Rackmount atau berdiri bebas

Komponen	Spesifikasi
Kabel daya	6 kaki (1,8 meter)
Kontrol daya	Tombol tekan
Atur ulang	Beralih sesaat
Status dan LED jaringan	LED RGB 3 warna yang dapat diprogram

Wi-Fi, Bluetooth dan penyimpanan kartu SD hadir pada alat tetapi tidak dapat digunakan.

AWS Panorama Appliance mencakup dua sekrup untuk pemasangan pada rak server. Anda dapat me-mount dua peralatan side-by-side di rak 19 inci.

Kuota layanan

AWS Panorama menerapkan kuota ke sumber daya yang Anda buat di akun dan aplikasi yang Anda gunakan. Jika Anda menggunakan AWS Panorama dalam beberapaAWSWilayah, kuota berlaku secara terpisah untuk setiap Wilayah. Kuota AWS Panorama tidak dapat disesuaikan.

Sumber daya di AWS Panorama mencakup perangkat, paket node aplikasi, dan instance aplikasi.

- Perangkat— Hingga 50 peralatan terdaftar per Wilayah.
- Paket node— 50 paket per Wilayah, dengan hingga 20 versi per paket.
- Contoh aplikasi— Hingga 10 aplikasi per perangkat. Setiap aplikasi dapat memantau hingga 8 aliran kamera. Penerapan dibatasi hingga 200 per hari untuk setiap perangkat.

Saat Anda menggunakan AWS Panorama Application CLI,AWS Command Line Interface, atauAWSSDK dengan layanan AWS Panorama, kuota berlaku untuk jumlah panggilan API yang Anda lakukan. Anda dapat membuat hingga 5 permintaan total per detik. Subset operasi API yang membuat atau memodifikasi sumber daya menerapkan batas tambahan 1 permintaan per detik.

Untuk daftar lengkap kuota, kunjungiKonsol Kuota Layanan, atau lihatTitik akhir dan kuota AWS PanoramadiReferensi Umum Amazon Web.

Izin AWS Panorama

Anda dapat menggunakanAWS Identity and Access Management (IAM) untuk mengelola akses keAWS Panorama layanan dan sumber daya seperti peralatan dan aplikasi. Untuk pengguna yang menggunakanAWS Panorama, Anda mengelola izin dalam kebijakan izin yang dapat Anda terapkan ke peran. Untuk mengelola izin untuk aplikasi, Anda membuat peran dan menetapkannya ke aplikasi.

Untuk mengelola izin bagi pengguna di akun Anda, gunakan kebijakan terkelola yangAWS Panorama memberikan, atau Anda tulis sendiri. Anda memerlukan izin keAWS layanan lain untuk mendapatkan log aplikasi dan alat, melihat metrik, dan menetapkan peran ke aplikasi.

AWS PanoramaAppliance juga memiliki peran yang memberikan izinnya untuk mengaksesAWS layanan dan sumber daya. Peran alat adalah salah satu peran layanan yang digunakanAWS Panorama layanan untuk mengakses layanan lain atas nama Anda.

Peran aplikasi adalah peran layanan terpisah yang Anda buat untuk aplikasi, untuk memberikannya izin untuk menggunakanAWS layanan denganAWS SDK for Python (Boto). Untuk membuat peran aplikasi, Anda memerlukan hak administratif atau bantuan administrator.

Anda dapat membatasi izin pengguna dengan sumber daya yang dipengaruhi tindakan, dan dalam beberapa kasus, dengan kondisi tambahan. Misalnya, Anda dapat menentukan pola untuk Amazon Resource Name (ARN) dari aplikasi yang mengharuskan pengguna menyertakan nama pengguna mereka dalam nama aplikasi yang mereka buat. Untuk sumber daya dan syarat yang didukung oleh setiap tindakan, lihat Tindakan, sumber daya, dan kunci syarat untukAWS Panorama di Referensi Otorisasi Layanan.

Untuk informasi lebih lanjut, lihat Apa itu IAM? Panduan Pengguna IAM

Topik

- Kebijakan IAM berbasis identitas untuk AWS AWS AWS AWS Panorama
- Peran layanan AWS Panorama dan sumber daya lintas layanan
- Memberikan izin untuk aplikasi

Kebijakan IAM berbasis identitas untuk AWS AWS AWS AWS Panorama

Untuk memberikan akses akun ke AWS Panorama, Anda menggunakan kebijakan berbasis identitas diAWS Identity and Access Management (IAM). Terapkan kebijakan berbasis identitas ke peran IAM yang terkait dengan pengguna. Anda juga dapat memberi pengguna di akun lain untuk berperan di akun Anda dan mengakses sumber daya AWS Panorama Anda.

AWS Panorama menyediakan kebijakan terkelola yang memberikan akses ke tindakan API AWS Panorama dan, dalam beberapa kasus, akses ke layanan lain yang digunakan untuk mengembangkan dan mengelola sumber daya AWS Panorama. AWS Panorama memperbarui kebijakan terkelola sesuai kebutuhan, untuk memastikan bahwa pengguna Anda memiliki akses ke fitur baru ketika dirilis.

- AWSPanoramaFullAccess— Menyediakan akses penuh ke AWS Panorama, titik akses AWS Panorama di Amazon S3, kredensi alatAWS Secrets Manager, dan log alat di Amazon CloudWatch. Termasuk izin untuk membuat peran terkait layanan untuk AWS Panorama. Lihat kebijakan

`AWSPanoramaFullAccess`Kebijakan ini memungkinkan Anda menandai sumber daya AWS Panorama, tetapi tidak memiliki semua izin terkait tag yang digunakan oleh konsol AWS Panorama. Untuk memberikan izin ini, tambahkan kebijakan berikut.

- ResourceGroupsandTagEditorFullAccess- Lihat kebijakan

`AWSPanoramaFullAccess`Kebijakan ini tidak menyertakan izin untuk membeli perangkat dari konsol AWS Panorama. Untuk memberikan izin ini, tambahkan kebijakan berikut.

- ElementalAppliancesSoftwareFullAccess- Lihat kebijakan

Kebijakan yang dikelola memberikan izin untuk tindakan API tanpa membatasi sumber daya yang dapat diubah pengguna. Untuk kontrol yang lebih cermat, Anda dapat membuat kebijakan Anda sendiri yang membatasi ruang lingkup izin pengguna. Gunakan kebijakan akses penuh sebagai titik awal kebijakan Anda.

Membuat peran layanan

Pertama kali Anda menggunakan konsol AWS Panorama, Anda memerlukan izin untuk membuat peran layanan yang digunakan oleh AWS Panorama Appliance. Peran layanan memberikan izin layanan untuk mengelola sumber daya atau berinteraksi dengan layanan lain. Buat peran ini sebelum memberikan akses ke pengguna Anda.

Untuk detail tentang sumber daya dan kondisi yang dapat Anda gunakan untuk membatasi cakupan izin pengguna di AWS Panorama, lihat Kunci Tindakan, sumber daya, dan kondisi untuk AWS Panorama dalam Referensi Otorisasi Layanan.

Peran layanan AWS Panorama dan sumber daya lintas layanan

AWS Panorama menggunakan layanan AWS lain untuk mengelola AWS Panorama Appliance, menyimpan data, dan mengimpor sumber daya aplikasi. Peran layanan memberikan izin layanan untuk mengelola sumber daya atau berinteraksi dengan layanan lain. Saat masuk ke konsol AWS Panorama untuk pertama kalinya, Anda membuat peran layanan berikut:

- AWSServiceRoleForAWSPanorama— Memungkinkan AWS Panorama mengelola sumber daya di AWS IoT, AWS Secrets Manager, dan AWS Panorama.

  Kebijakan terkelola:AWSPanoramaServiceLinkedRolePolicy
- AWSPanoramaApplianceServiceRole— Memungkinkan AWS Panorama Appliance untuk mengunggah log CloudWatch, dan untuk mendapatkan objek dari titik akses Amazon S3 yang dibuat oleh AWS Panorama.

  Kebijakan terkelola:AWSPanoramaApplianceServiceRolePolicy

Untuk melihat izin yang dilampirkan ke setiap peran, gunakanKonsol IAM. Jika memungkinkan, izin peran dibatasi pada sumber daya yang sesuai dengan pola penamaan yang digunakan AWS Panorama. Misalnya,`AWSServiceRoleForAWSPanorama`hanya memberikan izin bagi layanan untuk mengaksesAWS IoTsumber daya yang memiliki`panorama`atas nama mereka.

Bagian

- Mengamankan peran alat
- Penggunaan layanan lain

Mengamankan peran alat

AWS Panorama Appliance menggunakan`AWSPanoramaApplianceServiceRole`peran untuk mengakses sumber daya di akun Anda. Alat memiliki izin untuk mengunggah log CloudWatch Log, baca kredensi aliran kamera dariAWS Secrets Manager, dan untuk mengakses artefak aplikasi di titik akses Amazon Simple Storage Service (Amazon S3) yang dibuat AWS Panorama.

> **Note**
>
> Aplikasi tidak menggunakan izin alat. Untuk memberikan izin aplikasi Anda untuk menggunakanAWSjasa, membuatperan aplikasi.

AWS Panorama menggunakan peran layanan yang sama dengan semua peralatan di akun Anda, dan tidak menggunakan peran di seluruh akun. Untuk lapisan keamanan tambahan, Anda dapat mengubah kebijakan kepercayaan peran alat untuk menegakkannya secara eksplisit, yang merupakan praktik terbaik saat Anda menggunakan peran untuk memberikan izin layanan untuk mengakses sumber daya di akun Anda.

Untuk memperbarui kebijakan kepercayaan peran alat

1. Membuka peran alat pada konsol IAM:AWSPanoramaApplianceServiceRole
2. Pilih Edit trust relationship (Edit Hubungan Kepercayaan).
3. Perbarui isi kebijakan, lalu pilihKebijakan kepercayaan pembaruan.

Kebijakan kepercayaan berikut mencakup kondisi yang memastikan bahwa ketika AWS Panorama mengambil peran alat, kebijakan tersebut dilakukan untuk alat di akun Anda. Klasteraws:SourceAccountkondisi membandingkan ID akun yang ditentukan oleh AWS Panorama dengan yang Anda sertakan dalam kebijakan.

Example kebijakan kepercayaan — Akun khusus

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "panorama.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
```

```
  ]
}
```

Jika Anda ingin membatasi AWS Panorama lebih jauh, dan memungkinkannya untuk hanya mengambil peran dengan perangkat tertentu, Anda dapat menentukan perangkat dengan ARN. Klasteraws:SourceArnkondisi membandingkan ARN alat yang ditentukan oleh AWS Panorama dengan yang Anda sertakan dalam kebijakan.

Example kebijakan kepercayaan - Alat tunggal

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "panorama.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:panorama:us-east-1:123456789012:device/
device-lk7exmplpvcr3heqwjmesw76ky"
        },
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

Jika Anda mengatur ulang dan menyediakan ulang alat, Anda harus menghapus kondisi ARN sumber untuk sementara dan kemudian menambahkannya lagi dengan ID perangkat baru.

Untuk informasi lebih lanjut tentang kondisi ini, dan praktik terbaik keamanan saat layanan menggunakan peran untuk mengakses sumber daya di akun Anda, lihatMasalah confused deputydi Panduan Pengguna IAM.

Penggunaan layanan lain

AWS Panorama membuat atau mengakses sumber daya dalam layanan berikut:

- AWS IoT— Hal-hal, kebijakan, sertifikat, dan pekerjaan untuk AWS Panorama Appliance
- Amazon S3- Titik akses untuk pementasan model aplikasi, kode, dan konfigurasi.
- Secrets Manager— Kredensi jangka pendek untuk AWS Panorama Appliance.

Untuk informasi tentang format Amazon Resource Name (ARN) atau cakupan izin untuk setiap layanan, lihat topik diPanduan Pengguna IAMyang ditautkan ke dalam daftar ini.

Memberikan izin untuk aplikasi

Anda dapat membuat peran untuk aplikasi Anda untuk memberikan izin untuk meneleponAWSlayanan. Secara default, aplikasi tidak memiliki izin. Anda membuat peran aplikasi di IAM dan menetapkannya ke aplikasi selama penyebaran. Untuk memberikan aplikasi Anda hanya izin yang dibutuhkan, buat peran untuk itu dengan izin untuk tindakan API tertentu.

Parameteraplikasi sampeltermasukAWS CloudFormationtemplate dan script yang membuat peran aplikasi. Ini adalahperan layananyang dapat diasumsikan AWS Panorama. Peran ini memberikan izin untuk aplikasi untuk memanggil CloudWatch untuk mengunggah metrik.

Example aws-panorama-sample.yml~— Peran aplikasi

```
Resources:
  runtimeRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - panorama.amazonaws.com
            Action:
              - sts:AssumeRole
      Policies:
        - PolicyName: cloudwatch-putmetrics
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action: 'cloudwatch:PutMetricData'
                Resource: '*'
      Path: /service-role/
```

Anda dapat memperluas skrip ini untuk memberikan izin ke layanan lain, dengan menentukan daftar tindakan API atau pola untuk nilai`Action`.

Untuk informasi selengkapnya mengenai izin di AWS Panorama, lihatIzin AWS Panorama.

MengelolaAWS PanoramaAlat

ParameterAWS PanoramaAppliance adalah perangkat keras yang menjalankan aplikasi Anda. Anda menggunakanAWS Panoramakonsol untuk mendaftarkan alat, memperbarui perangkat lunak, dan menyebarkan aplikasi untuk itu. Perangkat lunak padaAWS PanoramaAppliance terhubung ke aliran kamera, mengirim frame video ke aplikasi Anda, dan menampilkan output video pada layar terlampir.

Setelah menyiapkan alat Anda atau yang lainperangkat kompatibel, Anda mendaftarkan kamera untuk digunakan dengan aplikasi. Andamengelola aliran kameradiAWS Panoramakonsol. Saat Anda menerapkan aplikasi, Anda memilih aliran kamera mana yang dikirim oleh alat untuk diproses.

Untuk tutorial yang memperkenalkanAWS PanoramaAppliance dengan aplikasi sampel, lihatMemulai dengan AWS Panorama.

Topik

- Mengelola AWS Panorama Appliance
- Menghubungkan AWS Panorama Appliance ke jaringan Anda
- Mengelola aliran kamera di AWS Panorama
- Mengelola aplikasi pada Appliance AWS Panorama
- Tombol dan lampu AWS Panorama Appliance

Mengelola AWS Panorama Appliance

Anda menggunakan konsol AWS Panorama untuk mengonfigurasi, meningkatkan, atau membatalkan pendaftaran AWS Panorama Appliance dan perangkat lain yang kompatibel.

Untuk menyiapkan alat, ikuti petunjuk dalam tutorial memulai. Proses penyiapan menciptakan sumber daya di AWS Panorama yang melacak alat Anda dan mengoordinasikan pembaruan dan penerapan.

Untuk mendaftarkan alat dengan AWS Panorama API, lihatMengotomatiskan pendaftaran perangkat.

Bagian

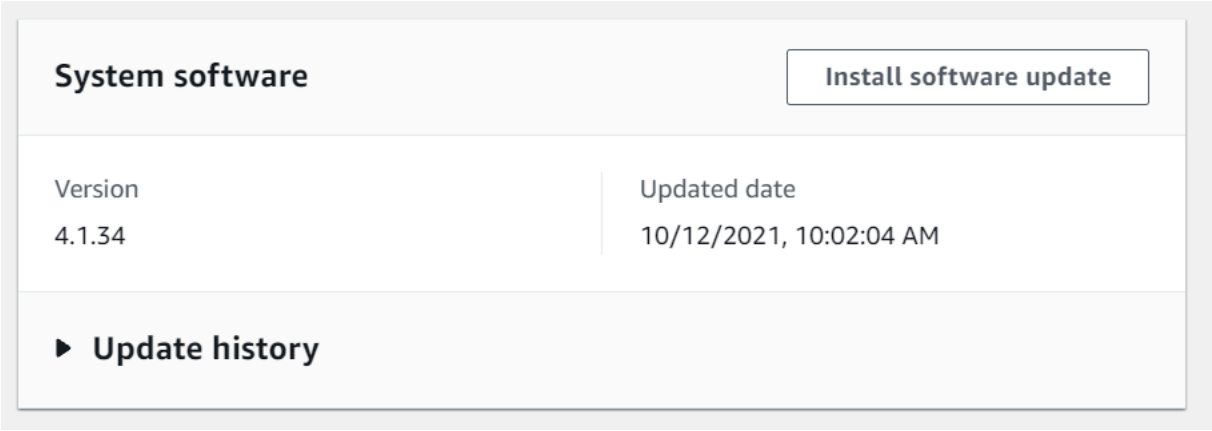
- Memperbarui perangkat lunak alat
- Membatalkan pendaftaran alat
- Mem-boot ulang alat
- Mengatur ulang alat

Memperbarui perangkat lunak alat

Anda melihat dan menerapkan pembaruan perangkat lunak untuk alat di konsol AWS Panorama. Pembaruan dapat diperlukan atau opsional. Ketika pembaruan yang diperlukan tersedia, konsol meminta Anda untuk menerapkannya. Anda dapat menerapkan pembaruan opsional pada halaman Pengaturan alat.

Untuk memperbarui perangkat lunak alat

1. Buka halaman Perangkat konsol AWS Panorama.
2. Pilih alat.
3. Pilih Pengaturan
4. Di bawah Perangkat lunak sistem, pilih Instal pembaruan perangkat lunak.



5. Pilih versi baru dan kemudian pilih Instal.

Membatalkan pendaftaran alat

Jika Anda selesai bekerja dengan alat, Anda dapat menggunakan konsol AWS Panorama untuk membatalkan pendaftaran dan menghapusAWS IoT sumber daya terkait.

Untuk menghapus alat

1. Buka halaman Perangkat konsol AWS Panorama.
2. Pilih nama alat.
3. Pilih Delete (Hapus).
4. Masukkan nama alat dan pilih Hapus.

Saat Anda menghapus alat dari layanan AWS Panorama, data pada alat tidak akan dihapus secara otomatis. Alat yang telah didaftarkan tidak dapat terhubung keAWS layanan dan tidak dapat didaftarkan lagi hingga perangkat diatur ulang.

Mem-boot ulang alat

Anda dapat me-reboot alat dari jarak jauh.

Untuk mem-boot ulang alat

1. Buka halaman Perangkat konsol AWS Panorama.
2. Pilih nama alat.
3. Pilih Boot ulang.

Konsol mengirim pesan ke alat untuk me-reboot. Untuk menerima sinyal, alat harus dapat terhubung keAWS IoT. Untuk me-reboot alat dengan AWS Panorama API, lihatPerlengkapan boot ulang.

Mengatur ulang alat

Untuk menggunakan alat di Wilayah yang berbeda atau dengan akun lain, Anda harus mengatur ulang dan menyediakannya kembali dengan sertifikat baru. Menyetel ulang perangkat menerapkan versi perangkat lunak terbaru yang diperlukan dan menghapus semua data akun.

Untuk memulai operasi reset, alat harus dicolokkan dan dimatikan. Tekan dan tahan tombol power dan reset selama lima detik. Saat Anda melepaskan tombol, lampu status berkedip oranye. Tunggu hingga lampu status berkedip hijau sebelum menyediakan atau melepaskan alat.

Anda juga dapat mengatur ulang perangkat lunak alat tanpa menghapus sertifikat dari perangkat. Untuk informasi selengkapnya, lihat Tombol daya dan reset.

Menghubungkan AWS Panorama Appliance ke jaringan Anda

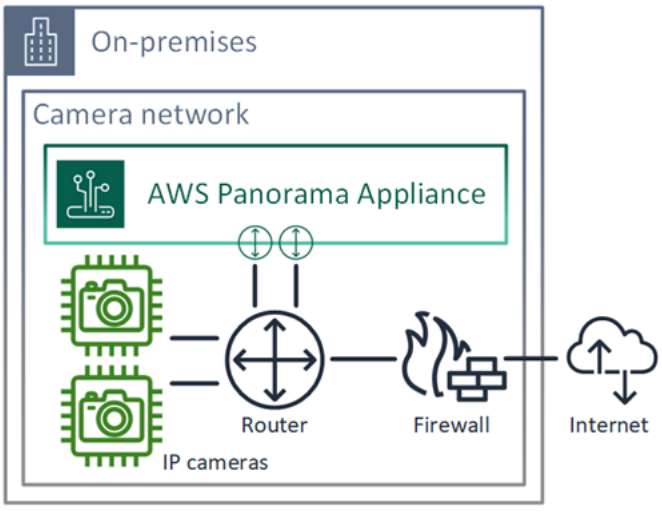
AWS Panorama Appliance memerlukan konektivitas ke AWS cloud dan jaringan kamera IP lokal Anda. Anda dapat menghubungkan alat ke firewall tunggal yang memberikan akses ke keduanya, atau menghubungkan masing-masing dari dua antarmuka jaringan perangkat ke subnet yang berbeda. Dalam kedua kasus tersebut, Anda harus mengamankan koneksi jaringan alat untuk mencegah akses tidak sah ke aliran kamera Anda.

Bagian-bagian

- Konfigurasi jaringan tunggal
- Konfigurasi jaringan ganda
- Mengkonfigurasi akses layanan
- Mengkonfigurasi akses jaringan lokal
- Konektivitas pribadi

Konfigurasi jaringan tunggal

Alat ini memiliki dua port Ethernet. Jika Anda merutekan semua lalu lintas ke dan dari perangkat melalui satu router, Anda dapat menggunakan port kedua untuk redundansi jika koneksi fisik ke port pertama rusak. Konfigurasikan router Anda untuk memungkinkan alat terhubung hanya ke aliran kamera dan internet, dan untuk memblokir aliran kamera agar tidak meninggalkan jaringan internal Anda.

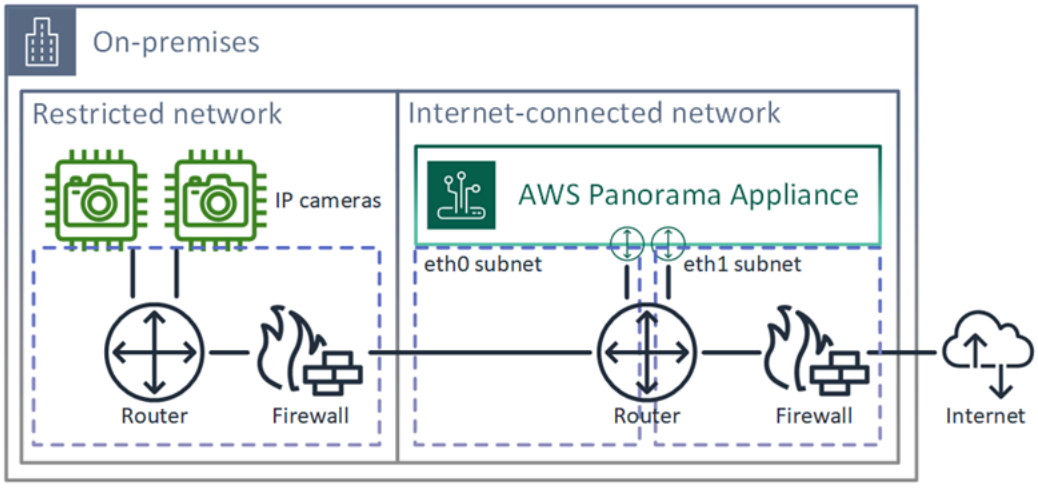


Untuk detail tentang port dan titik akhir yang perlu diakses oleh alat, lihat Mengkonfigurasi akses layanan danMengkonfigurasi akses jaringan lokal.

Konfigurasi jaringan ganda

Untuk lapisan keamanan tambahan, Anda dapat menempatkan alat di jaringan yang terhubung ke internet terpisah dari jaringan kamera Anda. Firewall antara jaringan kamera terbatas Anda dan jaringan alat hanya memungkinkan alat untuk mengakses aliran video. Jika jaringan kamera Anda sebelumnya memiliki celah udara untuk tujuan keamanan, Anda mungkin lebih memilih metode ini daripada menghubungkan jaringan kamera ke router yang juga memberikan akses ke internet.

Contoh berikut menunjukkan alat yang menghubungkan ke subnet yang berbeda pada setiap port. Router menempatkan `eth0` antarmuka pada subnet yang merutekan ke jaringan kamera, dan `eth1` pada subnet yang merutekan ke internet.



Anda dapat mengonfirmasi alamat IP dan alamat MAC dari setiap port di konsol AWS Panorama.

Mengkonfigurasi akses layanan

Selama penyediaan, Anda dapat mengonfigurasi alat untuk meminta alamat IP tertentu. Pilih alamat IP sebelumnya untuk menyederhanakan konfigurasi firewall dan memastikan bahwa alamat alat tidak berubah jika offline untuk jangka waktu yang lama.

Alat ini menggunakan AWS layanan untuk mengoordinasikan pembaruan dan penerapan perangkat lunak. Konfigurasikan firewall Anda untuk memungkinkan alat terhubung ke titik akhir ini.

Akses internet

- AWS IoT(HTTPS dan MQTT, port 443, 8443 dan 8883) — dan titik akhir manajemen perangkat. AWS IoT Core Untuk detailnya, lihat titik akhir dan kuota AWS IoT Device Management di bagian. Referensi Umum Amazon Web Services
- AWS IoTkredensi (HTTPS, port 443) — `credentials.iot.<region>.amazonaws.com` dan subdomain.
- Amazon Elastic Container Registry (HTTPS, port 443) —`api.ecr.<region>.amazonaws.com`, `dkr.ecr.<region>.amazonaws.com` dan subdomain.
- Amazon CloudWatch (HTTPS, port 443) —`monitoring.<region>.amazonaws.com`.
- CloudWatch Log Amazon (HTTPS, port 443) —`logs.<region>.amazonaws.com`.
- Amazon Simple Storage Service (HTTPS, port 443) —`s3.<region>.amazonaws.com`, `s3-accesspoint.<region>.amazonaws.com` dan subdomain.

Jika aplikasi Anda memanggil AWS layanan lain, alat memerlukan akses ke titik akhir untuk layanan tersebut juga. Untuk informasi selengkapnya, lihat Titik akhir dan kuota layanan.

Mengkonfigurasi akses jaringan lokal

Alat ini membutuhkan akses ke aliran video RTSP secara lokal, tetapi tidak melalui internet. Konfigurasikan firewall Anda untuk memungkinkan alat mengakses aliran RTSP di port 554 secara internal, dan untuk tidak mengizinkan aliran keluar atau masuk dari internet.

Akses lokal

- Protokol streaming waktu nyata (RTSP, port 554) - Untuk membaca aliran kamera.
- Protokol waktu jaringan (NTP, port 123) - Untuk menjaga jam alat tetap sinkron. Jika Anda tidak menjalankan server NTP di jaringan Anda, alat ini juga dapat terhubung ke server NTP publik melalui internet.

Konektivitas pribadi

AWS Panorama Appliance tidak memerlukan akses internet jika Anda menyebarkannya di subnet VPC pribadi dengan koneksi VPN ke. AWS Anda dapat menggunakan Site-to-Site VPN AWS Direct Connect atau untuk membuat koneksi VPN antara router lokal dan. AWS Dalam subnet VPC pribadi Anda, Anda membuat titik akhir yang memungkinkan alat terhubung ke Amazon Simple Storage

ServiceAWS IoT, dan layanan lainnya. Untuk informasi selengkapnya, lihat Menghubungkan alat ke subnet pribadi.

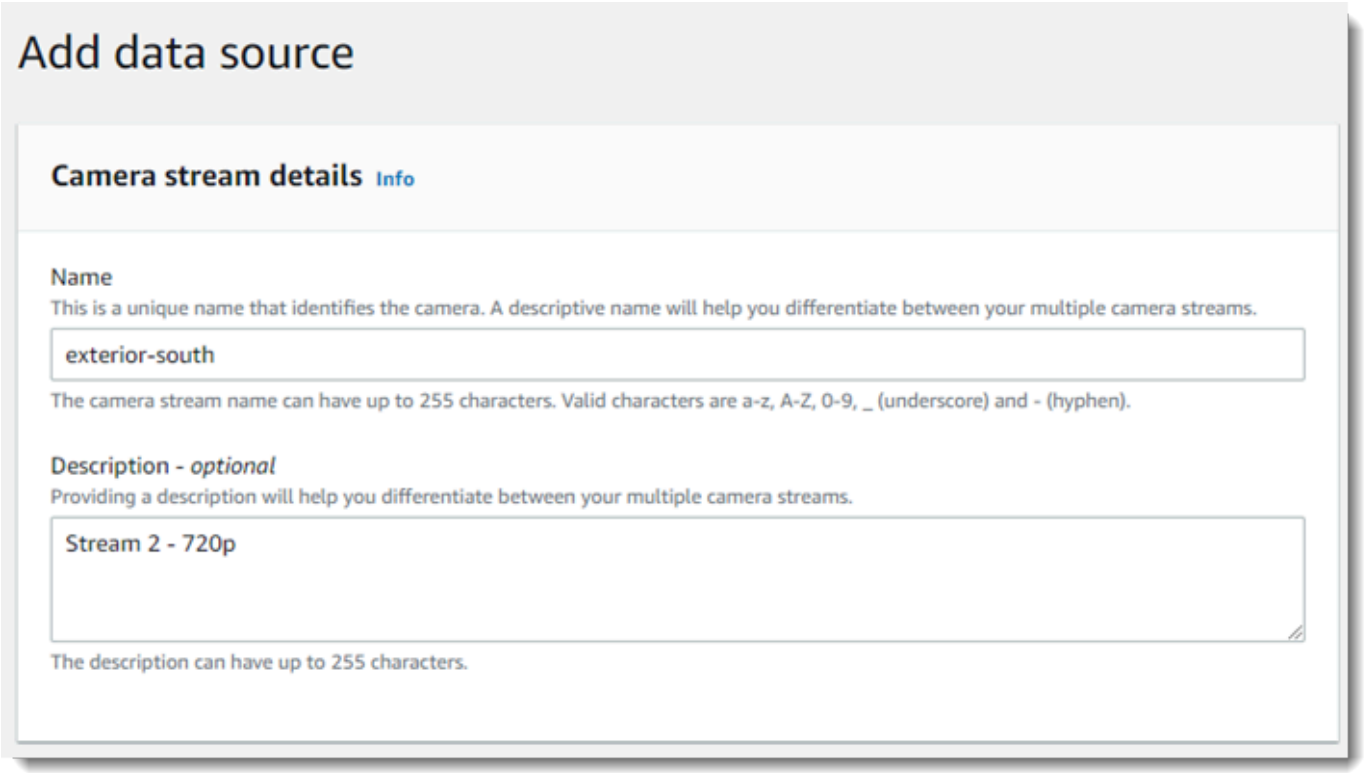
Mengelola aliran kamera di AWS Panorama

Untuk mendaftarkan aliran video sebagai sumber data untuk aplikasi Anda, gunakan konsol AWS Panorama. Aplikasi dapat memproses beberapa aliran secara bersamaan dan beberapa peralatan dapat terhubung ke aliran yang sama.

> **Important**
>
> Sebuah aplikasi dapat terhubung ke setiap aliran kamera yang routable dari jaringan lokal yang terhubung ke. Untuk mengamankan aliran video Anda, konfigurasikan jaringan Anda untuk mengizinkan hanya lalu lintas RTSP secara lokal. Untuk informasi selengkapnya, lihat Keamanan di AWS Panorama.

Untuk mendaftarkan aliran kamera

1. Buka konsol AWS PanoramaHalaman sumber data.
2. MemiilihMenambahkan sumber data.



3. Konfigurasikan pengaturan berikut.

- Nama— Sebuah nama untuk aliran kamera.
- Deskripsi— Deskripsi singkat tentang kamera, lokasinya, atau detail lainnya.
- URL— URL yang menentukan alamat IP kamera dan jalur ke aliran. Misalnya, `rtsp://192.168.0.77/live/mpeg4/`
- Kredensial— Jika aliran kamera dilindungi kata sandinya, tentukan nama pengguna dan kata sandinya.

4. Pilih Save (Simpan).

Untuk mendaftarkan stream kamera dengan AWS Panorama API, lihatMengotomatiskan pendaftaran perangkat.

Untuk daftar kamera yang kompatibel dengan AWS Panorama Appliance, lihatModel dan kamera visi komputer yang didukung.

Menghapus aliran

Anda dapat menghapus stream kamera di konsol AWS Panorama.

Untuk menghapus stream kamera

1. Buka konsol AWS PanoramaHalaman sumber data.
2. Memiilih aliran kamera.
3. MemiilihMenghapus sumber data.

Menghapus aliran kamera dari layanan tidak berhenti menjalankan aplikasi atau menghapus kredensi kamera dari Secrets Manager. Untuk menghapus rahasia, gunakanSecrets Manager.

Mengelola aplikasi pada Appliance AWS Panorama

Aplikasi adalah kombinasi kode, model, dan konfigurasi. DariPerangkathalaman di konsol AWS Panorama, Anda dapat mengelola aplikasi pada alat.

Untuk mengelola aplikasi di AWS Panorama Appliance

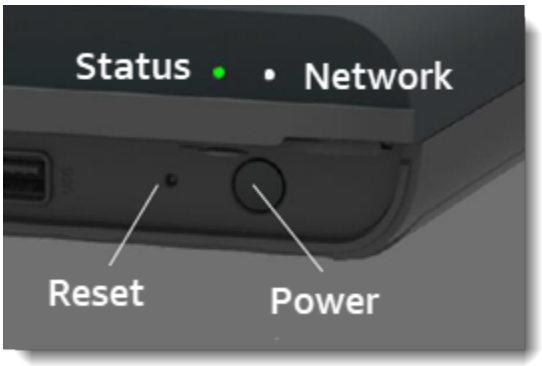
1. Buka konsol AWS PanoramaHalaman Perangkat.
2. Pilih alat.

ParameterAplikasi yang di-deployHalaman menunjukkan aplikasi yang telah dikerahkan ke alat.

Gunakan opsi di halaman ini untuk menghapus aplikasi yang dikerahkan dari alat, atau mengganti aplikasi yang sedang berjalan dengan versi baru. Anda juga dapat mengkloning aplikasi (berjalan atau dihapus) untuk menyebarkan salinan baru itu.

Tombol dan lampu AWS Panorama Appliance

AWS Panorama Appliance memiliki dua lampu LED di atas tombol daya yang menunjukkan status perangkat dan konektivitas jaringan.



Lampu status

LED berubah warna dan berkedip untuk menunjukkan status. Kedipan lambat setiap tiga detik sekali. Kedipan cepat sekali per detik.

Status status LED

- Hijau Berkedip cepat- Alat booting up.
- Hijau pekat- Alat beroperasi secara normal.
- Biru Berkedip lambat- Alat ini menyalin file konfigurasi dan mencoba mendaftarAWS IoT.
- Biru Berkedip cepat- Alat inimenyalin gambar logke drive USB.
- Merah pekat cepat- Alat mengalami kesalahan saat startup atau terlalu panas.
- Oranye Berkedip lambat- Alat ini memulihkan versi perangkat lunak terbaru.
- Oranye Berkedip cepat- Alat ini memulihkan versi perangkat lunak minimum.

Cahaya jaringan

LED jaringan memiliki status sebagai berikut:

Status LED jaringan

- Hijau pekat- Kabel Ethernet terhubung.
- Berkedip hijau- Alat berkomunikasi melalui jaringan.

- Merah pekat- Kabel Ethernet tidak terhubung.

Tombol daya dan reset

Tombol daya dan reset ada di bagian depan perangkat di bawah penutup pelindung. Tombol reset lebih kecil dan tersembunyi. Gunakan obeng kecil atau penjepit kertas untuk menekannya.

Untuk mengatur ulang alat

1. Alat harus dicolokkan dan dimatikan. Untuk mematikan alat, tahan tombol daya selama 1 detik dan tunggu hingga urutan shutdown selesai. Urutan shutdown memakan waktu sekitar 10 detik.
2. Untuk mengatur ulang alat, gunakan kombinasi tombol berikut. Pers singkat adalah 1 detik. Tekan lama adalah 5 detik. Untuk operasi yang memerlukan banyak tombol, tekan dan tahan kedua tombol secara bersamaan.

   - Pengaturan ulang penuh- Tekan daya lama dan reset.

     Mengembalikan versi perangkat lunak minimum dan menghapus semua file konfigurasi dan aplikasi.
   - Kembalikan versi perangkat lunak terbaru- Tekan ulang singkat.

     Menerapkan kembali pembaruan perangkat lunak terbaru ke alat.
   - Kembalikan versi perangkat lunak minimum- Tekan lama ulang.

     Menerapkan kembali pembaruan perangkat lunak terbaru yang diperlukan ke alat.
3. Lepaskan kedua tombol. Alat menyala dan lampu status berkedip oranye selama beberapa menit.
4. Saat alat sudah siap, lampu status berkedip hijau.

Mengatur ulang alat tidak menghapusnya dari layanan AWS Panorama. Untuk informasi selengkapnya, lihat Membatalkan pendaftaran alat.

MengelolaAWS Panoramaaplikasi

Aplikasi berjalan diAWS PanoramaAlat untuk melakukan tugas visi komputer pada aliran video. Anda dapat membangun aplikasi visi komputer dengan menggabungkan kode Python dan model machine learning, dan menerapkannya keAWS PanoramaAppliance melalui internet. Aplikasi dapat mengirim video ke layar, atau menggunakan AWS SDK untuk mengirim hasil ke layanan AWS.

Topik

- Menerapkan aplikasi
- Mengelola aplikasi di konsol AWS Panorama
- Konfigurasi
- Manifes aplikasi AWS Panorama
- Node aplikasi
- Parameter aplikasi
- Konfigurasi Deploy-time dengan menimpa

Menerapkan aplikasi

Untuk menerapkan aplikasi, Anda menggunakan AWS Panorama Application CLI yang mengimpornya ke akun Anda, membuat kontainer, mengunggah dan mendaftarkan aset, dan membuat instance aplikasi. Topik ini masuk ke masing-masing langkah ini secara rinci dan menjelaskan apa yang terjadi di latar belakang.

Jika Anda belum menerapkan aplikasi, lihat Memulai dengan AWS Panorama panduan.

Untuk informasi lebih lanjut tentang menyesuaikan dan memperluas aplikasi sampel, lihat. Membangun aplikasi AWS Panorama

Bagian

- Instal CLI Aplikasi AWS Panorama
- Mengimpor aplikasi
- Membangun gambar kontainer
- Impor model
- Unggah aset aplikasi
- Menerapkan aplikasi dengan konsol AWS Panorama
- Mengotomatiskan penerapan aplikasi

Instal CLI Aplikasi AWS Panorama

Untuk menginstal AWS Panorama Application CLI danAWS CLI, gunakan pip.

```
$ pip3 install --upgrade awscli panoramacli
```

Untuk membuat gambar aplikasi dengan AWS Panorama Application CLI, Anda memerlukan Docker. Di Linux, `qemu` dan perpustakaan sistem terkait juga diperlukan. Untuk informasi selengkapnya tentang menginstal dan mengonfigurasi AWS Panorama Application CLI, lihat file README di repositori proyek. GitHub

- github.com/aws/ aws-panorama-cli

Untuk petunjuk tentang pengaturan lingkungan build di Windows dengan WSL2, lihat. Menyiapkan lingkungan pengembangan di Windows

Mengimpor aplikasi

Jika Anda bekerja dengan aplikasi sampel atau aplikasi yang disediakan oleh pihak ketiga, gunakan AWS Panorama Application CLI untuk mengimpor aplikasi.

```
my-app$ panorama-cli import-application
```

Perintah ini mengganti nama paket aplikasi dengan ID akun Anda. Nama paket dimulai dengan ID akun akun tempat mereka digunakan. Saat Anda menerapkan aplikasi ke beberapa akun, Anda harus mengimpor dan mengemas aplikasi secara terpisah untuk setiap akun.

Misalnya, contoh aplikasi panduan ini paket kode dan paket model, masing-masing diberi nama dengan ID akun placeholder. `import-application`Perintah mengganti nama ini untuk menggunakan ID akun yang disimpulkan CLI dari kredensyal ruang kerja Anda. AWS

```
/aws-panorama-sample
### assets
### graphs
#   ### my-app
#       ### graph.json
### packages
    ### 123456789012-SAMPLE_CODE-1.0
    #   ### Dockerfile
    #   ### application.py
    #   ### descriptor.json
    #   ### package.json
    #   ### requirements.txt
    #   ### squeezenet_classes.json
    ### 123456789012-SQUEEZENET_PYTORCH-1.0
        ### descriptor.json
        ### package.json
```

`123456789012`diganti dengan ID akun Anda dalam nama direktori paket, dan dalam manifes aplikasi (`graph.json`), yang merujuk kepada mereka. Anda dapat mengonfirmasi ID akun Anda dengan menelepon `aws sts get-caller-identity` denganAWS CLI.

```
$ aws sts get-caller-identity
{
    "UserId": "AIDAXMPL7W66UC3GFXMPL",
    "Account": "210987654321",
    "Arn": "arn:aws:iam::210987654321:user/devenv"
```

```
}
```

Membangun gambar kontainer

Kode aplikasi Anda dikemas dalam gambar kontainer Docker, yang mencakup kode aplikasi dan pustaka yang Anda instal di Dockerfile Anda. Gunakan `build-container` perintah AWS Panorama Application CLI untuk membuat image Docker dan mengekspor image filesystem.

```
my-app$ panorama-cli build-container --container-asset-name code_asset --package-path
 packages/210987654321-SAMPLE_CODE-1.0
{
    "name": "code_asset",
    "implementations": [
        {
            "type": "container",
            "assetUri":
 "5fa5xmplbc8c16bf8182a5cb97d626767868d3f4d9958a4e49830e1551d227c5.tar.gz",
            "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
        }
    ]
}
Container asset for the package has been succesfully built at
 assets/5fa5xmplbc8c16bf8182a5cb97d626767868d3f4d9958a4e49830e1551d227c5.tar.gz
```

Perintah ini membuat image Docker bernama `code_asset` dan mengekspor filesystem ke arsip dalam `.tar.gz` folder. `assets` CLI menarik gambar dasar aplikasi dari Amazon Elastic Container Registry (Amazon ECR), sebagaimana ditentukan dalam Dockerfile aplikasi.

Selain arsip kontainer, CLI menciptakan aset untuk deskriptor paket ()`descriptor.json`. Kedua file diganti namanya dengan pengenal unik yang mencerminkan hash dari file asli. AWS Panorama Application CLI juga menambahkan blok ke konfigurasi paket yang mencatat nama kedua aset tersebut. Nama-nama ini digunakan oleh alat selama proses penyebaran.

Example Paket/123456789012-Sample_Code-1.0/Package.json - dengan blok aset

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
```

```
        "description": "Computer vision application code.",
        "assets": [
            {
                "name": "code_asset",
                "implementations": [
                    {
                        "type": "container",
                        "assetUri":
"5fa5xmplbc8c16bf8182a5cb97d626767868d3f4d9958a4e49830e1551d227c5.tar.gz",
                        "descriptorUri":
"1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
                    }
                ]
            }
        ],
        "interfaces": [
            {
                "name": "interface",
                "category": "business_logic",
                "asset": "code_asset",
                "inputs": [
                    {
                        "name": "video_in",
                        "type": "media"
                    },
```

Nama aset kode, yang ditentukan dalam `build-container` perintah, harus sesuai dengan nilai `asset` bidang dalam konfigurasi paket. Dalam contoh sebelumnya, kedua nilai tersebut`code_asset`.

Impor model

Aplikasi Anda mungkin memiliki arsip model di folder asetnya atau yang Anda unduh secara terpisah. Jika Anda memiliki model baru, model yang diperbarui, atau file deskriptor model yang diperbarui, gunakan `add-raw-model` perintah untuk mengimpornya.

```
my-app$ panorama-cli add-raw-model --model-asset-name model_asset \
      --model-local-path my-model.tar.gz \
      --descriptor-path packages/210987654321-SQUEEZENET_PYTORCH-1.0/descriptor.json \
      --packages-path packages/210987654321-SQUEEZENET_PYTORCH-1.0
```

Jika Anda hanya perlu memperbarui file deskriptor, Anda dapat menggunakan kembali model yang ada di direktori aset. Anda mungkin perlu memperbarui file deskriptor untuk mengonfigurasi fitur

seperti mode presisi floating point. Misalnya, skrip berikut menunjukkan cara melakukannya dengan aplikasi contoh.

Example util-skrip/.sh update-model-config

```
#!/bin/bash
set -eo pipefail
MODEL_ASSET=fd1axmplacc3350a5c2673adacffab06af54c3f14da6fe4a8be24cac687a386e
MODEL_PACKAGE=SQUEEZENET_PYTORCH
ACCOUNT_ID=$(ls packages | grep -Eo '[0-9]{12}' | head -1)
panorama-cli add-raw-model --model-asset-name model_asset --model-local-path assets/
${MODEL_ASSET}.tar.gz --descriptor-path packages/${ACCOUNT_ID}-${MODEL_PACKAGE}-1.0/
descriptor.json --packages-path packages/${ACCOUNT_ID}-${MODEL_PACKAGE}-1.0
cp packages/${ACCOUNT_ID}-${MODEL_PACKAGE}-1.0/package.json packages/${ACCOUNT_ID}-
${MODEL_PACKAGE}-1.0/package.json.bup
```

Perubahan pada file deskriptor dalam direktori paket model tidak diterapkan sampai Anda mengimpor ulang dengan CLI. CLI memperbarui konfigurasi paket model dengan nama aset baru di tempat, mirip dengan cara memperbarui konfigurasi untuk paket kode aplikasi saat Anda membangun kembali wadah.

Unggah aset aplikasi

Untuk mengunggah dan mendaftarkan aset aplikasi, yang mencakup arsip model, arsip filesystem kontainer, dan file deskriptor mereka, gunakan perintah. `package-application`

```
my-app$ panorama-cli package-application
Uploading package SQUEEZENET_PYTORCH
Patch version for the package
 5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96
Deregistering previous patch version
 e845xmpl8ea0361eb345c313a8dded30294b3a46b486dc8e7c174ee7aab29362
Asset fd1axmplacc3350a5c2673adacffab06af54c3f14da6fe4a8be24cac687a386e.tar.gz already
 exists, ignoring upload
upload: assets/87fbxmpl6f18aeae4d1e3ff8bbc6147390feaf47d85b5da34f8374974ecc4aaf.json
 to s3://arn:aws:s3:us-east-2:212345678901:accesspoint/
panorama-210987654321-6k75xmpl2jypelgzst7uux62ye/210987654321/nodePackages/
SQUEEZENET_PYTORCH/
binaries/87fbxmpl6f18aeae4d1e3ff8bbc6147390feaf47d85b5da34f8374974ecc4aaf.json
Called register package version for SQUEEZENET_PYTORCH with patch version
 5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96
```

```
...
```

Jika tidak ada perubahan pada file aset atau konfigurasi paket, CLI akan melewatkannya.

```
Uploading package SAMPLE_CODE
Patch Version ca91xmplca526fe3f07821fb0c514f70ed0c444f34cb9bd3a20e153730b35d70 already
 registered, ignoring upload
Register patch version complete for SQUEEZENET_PYTORCH with patch version
 5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96
Register patch version complete for SAMPLE_CODE with patch version
 ca91xmplca526fe3f07821fb0c514f70ed0c444f34cb9bd3a20e153730b35d70
All packages uploaded and registered successfully
```

CLI mengunggah aset untuk setiap paket ke titik akses Amazon S3 yang khusus untuk akun Anda. AWS Panorama mengelola titik akses untuk Anda, dan memberikan informasi tentangnya melalui API. DescribePackage CLI mengunggah aset untuk setiap paket ke lokasi yang disediakan untuk paket tersebut, dan mendaftarkannya dengan layanan AWS Panorama dengan pengaturan yang dijelaskan oleh konfigurasi paket.

Menerapkan aplikasi dengan konsol AWS Panorama

Anda dapat menerapkan aplikasi dengan konsol AWS Panorama. Selama proses penyebaran, Anda memilih aliran kamera mana yang akan diteruskan ke kode aplikasi, dan mengonfigurasi opsi yang disediakan oleh pengembang aplikasi.

Untuk menyebarkan aplikasi

1. Buka halaman AWS Panorama console Aplikasi yang disebarkan.
2. Pilih Menyebarkan aplikasi.
3. Tempel isi manifes aplikasi`graph.json`,, ke editor teks. Pilih Selanjutnya.
4. Masukkan nama dan descroption.
5. Pilih Lanjutkan untuk menyebarkan.
6. Pilih Mulai penyebaran.
7. Jika aplikasi Anda menggunakan peran, pilih dari menu drop-down. Pilih Selanjutnya.
8. Pilih Pilih perangkat, lalu pilih alat Anda. Pilih Selanjutnya.
9. Pada langkah Pilih sumber data, pilih Lihat masukan, dan tambahkan aliran kamera Anda sebagai sumber data. Pilih Selanjutnya.

10. Pada langkah Konfigurasi, konfigurasikan pengaturan khusus aplikasi apa pun yang ditentukan oleh pengembang. Pilih Selanjutnya.
11. Pilih Deploy, lalu pilih Selesai.
12. Dalam daftar aplikasi yang dikerahkan, pilih aplikasi untuk memantau statusnya.

Proses penyebaran memakan waktu 15-20 menit. Output alat bisa kosong untuk waktu yang lama saat aplikasi dimulai. Jika Anda menemukan kesalahan, lihatPemecahan Masalah.

Mengotomatiskan penerapan aplikasi

Anda dapat mengotomatiskan proses penyebaran aplikasi dengan API. CreateApplicationInstance API mengambil dua file konfigurasi sebagai input. Manifes aplikasi menentukan paket yang digunakan dan hubungannya. File kedua adalah file menimpa yang menentukan penggantian waktu penerapan nilai dalam manifes aplikasi. Menggunakan file menimpa memungkinkan Anda menggunakan manifes aplikasi yang sama untuk menerapkan aplikasi dengan aliran kamera yang berbeda, dan mengonfigurasi setelan khusus aplikasi lainnya.

Untuk informasi lebih lanjut, dan contoh skrip untuk setiap langkah dalam topik ini, lihatMengotomatiskan penerapan aplikasi.

Mengelola aplikasi di konsol AWS Panorama

Gunakan konsol AWS Panorama untuk mengelola aplikasi yang diterapkan.

Bagian

- Memperbarui atau menyalin aplikasi
- Hapus versi dan aplikasi

Memperbarui atau menyalin aplikasi

Untuk memperbarui aplikasi, gunakanGantipilihan. Saat Anda mengganti aplikasi, Anda dapat memperbarui kode atau modelnya.

Untuk memperbarui aplikasi

1. Buka konsol AWS PanoramaHalaman aplikasi yang di-deploy.
2. Pilih aplikasi.
3. Pilih Ganti.
4. Ikuti petunjuk untuk membuat versi atau aplikasi baru.

Ada jugaclonepilihan yang bertindak mirip denganGanti, tetapi tidak menghapus versi lama dari aplikasi. Anda dapat menggunakan opsi ini untuk menguji perubahan pada aplikasi tanpa menghentikan versi yang sedang berjalan, atau untuk men-deploy ulang versi yang telah Anda hapus.

Hapus versi dan aplikasi

Untuk membersihkan versi aplikasi yang tidak terpakai, hapus dari peralatan Anda.

Untuk menghapus aplikasi

1. Buka konsol AWS PanoramaHalaman aplikasi yang di-deploy.
2. Pilih aplikasi.
3. PilihHapus dari perangkat.

Konfigurasi

Saat Anda menggunakan perintah AWS Panorama Application CLI`panorama-cli package-application`, CLI mengunggah aset aplikasi Anda ke Amazon S3 dan mendaftarkannya dengan AWS Panorama. Aset mencakup file biner (gambar dan model kontainer) dan file deskriptor, yang diunduh AWS Panorama Appliance selama penerapan. Untuk mendaftarkan aset paket, Anda menyediakan file konfigurasi paket terpisah yang mendefinisikan paket, asetnya, dan antarmuka.

Contoh berikut menunjukkan konfigurasi paket untuk node kode dengan satu input dan satu output. Input video menyediakan akses ke data gambar dari aliran kamera. Node output mengirimkan gambar diproses ke layar.

Example Paket/1234567890-sample_code-1.0/package.json

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [
            {
                "name": "code_asset",
                "implementations": [
                    {
                        "type": "container",
                        "assetUri":
 "3d9bxmplbdb67a3c9730abb19e48d78780b507f3340ec3871201903d8805328a.tar.gz",
                        "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
                    }
                ]
            }
        ],
        "interfaces": [
            {
                "name": "interface",
                "category": "business_logic",
                "asset": "code_asset",
                "inputs": [
                    {
                        "name": "video_in",
```

```
                        "type": "media"
                    }
                ],
                "outputs": [
                    {
                        "description": "Video stream output",
                        "name": "video_out",
                        "type": "media"
                    }
                ]
            }
        ]
    }
}
```

Parameter`assets`bagian menentukan nama artefak yang diunggah CLI Aplikasi AWS Panorama ke Amazon S3. Jika Anda mengimpor aplikasi sampel atau aplikasi dari pengguna lain, bagian ini bisa kosong atau merujuk ke aset yang tidak ada di akun Anda. Ketika Anda menjalankan`panorama-cli package-application`, AWS Panorama Application CLI mengisi bagian ini dengan nilai yang benar.

Manifes aplikasi AWS Panorama

Saat Anda menerapkan aplikasi, Anda menyediakan file konfigurasi yang disebut manifes aplikasi. File ini mendefinisikan aplikasi sebagai grafik dengan node dan tepi. Manifes aplikasi adalah bagian dari kode sumber aplikasi dan disimpan dalam`graphs`direktori

Example Grafik/aws-panorama-sample/graph.json

```
{
    "nodeGraph": {
        "envelopeVersion": "2021-01-01",
        "packages": [
            {
                "name": "123456789012::SAMPLE_CODE",
                "version": "1.0"
            },
            {
                "name": "123456789012::SQUEEZENET_PYTORCH_V1",
                "version": "1.0"
            },
            {
                "name": "panorama::abstract_rtsp_media_source",
                "version": "1.0"
            },
            {
                "name": "panorama::hdmi_data_sink",
                "version": "1.0"
            }
        ],
        "nodes": [
            {
                "name": "code_node",
                "interface": "123456789012::SAMPLE_CODE.interface"
            }
            {
                "name": "model_node",
                "interface": "123456789012::SQUEEZENET_PYTORCH_V1.interface"
            },
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "overrideMandatory": true,
```

```
                "decorator": {
                    "title": "IP camera",
                    "description": "Choose a camera stream."
                }
            },
            {
                "name": "output_node",
                "interface": "panorama::hdmi_data_sink.hdmi0"
            },
            {
                "name": "log_level",
                "interface": "string",
                "value": "INFO",
                "overridable": true,
                "decorator": {
                    "title": "Logging level",
                    "description": "DEBUG, INFO, WARNING, ERROR, or CRITICAL."
                }
            }
            ...
        ],
        "edges": [
            {
                "producer": "camera_node.video_out",
                "consumer": "code_node.video_in"
            },
            {
                "producer": "code_node.video_out",
                "consumer": "output_node.video_in"
            },
            {
                "producer": "log_level",
                "consumer": "code_node.log_level"
            }
        ]
    }
}
```

Node dihubungkan oleh tepi, yang menentukan pemetaan antara input dan output node. Output dari satu node terhubung ke input yang lain, membentuk grafik.

Skema JSON

Format aplikasi manifest dan menimpa dokumen didefinisikan dalam skema JSON. Anda dapat menggunakan skema JSON untuk memvalidasi dokumen konfigurasi Anda sebelum menerapkan. Skema JSON tersedia dalam panduan iniGitHubrepositori.

- Skema JSON—aws-panorama-developer-Panduan/sumber daya

Node aplikasi

Node adalah model, kode, aliran kamera, output, dan parameter. Sebuah node memiliki antarmuka, yang mendefinisikan input dan output. Antarmuka dapat didefinisikan dalam paket di akun Anda, paket yang disediakan oleh AWS Panorama, atau tipe bawaan.

Pada contoh berikut,`code_node`dan`model_node`lihat contoh kode dan model paket disertakan dengan aplikasi sampel.`camera_node`menggunakan paket yang disediakan oleh AWS Panorama untuk membuat placeholder untuk streaming kamera yang Anda tentukan selama penerapan.

Example graph.json — Node

```
        "nodes": [
            {
                "name": "code_node",
                "interface": "123456789012::SAMPLE_CODE.interface"
            },
            {
                "name": "model_node",
                "interface": "123456789012::SQUEEZENET_PYTORCH_V1.interface"
            },
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "overrideMandatory": true,
                "decorator": {
                    "title": "IP camera",
                    "description": "Choose a camera stream."
                }
            }
    ]
```

Tepi

Tepi memetakan output dari satu node ke input yang lain. Pada contoh berikut, tepi pertama memetakan output dari node aliran kamera ke input dari node kode aplikasi. Nama`video_in`dan`video_out`didefinisikan dalam antarmuka paket node.

Example graph.json — tepi

```
        "edges": [
```

```
            {
                "producer": "camera_node.video_out",
                "consumer": "code_node.video_in"
            },
            {
                "producer": "code_node.video_out",
                "consumer": "output_node.video_in"
            },
```

Pada kode aplikasi Anda, gunakan`inputs`dan`outputs`atribut untuk mendapatkan gambar dari aliran input, dan mengirim gambar ke aliran output.

Example application.py - masukan Video dan output

```
    def process_streams(self):
        """Processes one frame of video from one or more video streams."""
        frame_start = time.time()
        self.frame_num += 1
        logger.debug(self.frame_num)
        # Loop through attached video streams
        streams = self.inputs.video_in.get()
        for stream in streams:
            self.process_media(stream)
        ...
        self.outputs.video_out.put(streams)
```

Simpul Abstrak

Dalam manifes aplikasi, node abstrak mengacu pada paket yang didefinisikan oleh AWS Panorama, yang dapat Anda gunakan sebagai placeholder dalam manifes aplikasi Anda. AWS Panorama menyediakan dua tipe node abstrak.

- Aliran kamera— Pilih aliran kamera yang digunakan aplikasi selama penyebaran.

  Nama paket—`panorama::abstract_rtsp_media_source`

  Nama antarmuka—`rtsp_v1_interface`
- Output HDMI— Menunjukkan bahwa aplikasi output video.

  Nama paket—`panorama::hdmi_data_sink`

Nama antarmuka—`hdmi0`

Contoh berikut menunjukkan satu set dasar paket, node, dan tepi untuk aplikasi yang memproses aliran kamera dan output video ke layar. Node kamera, yang menggunakan antarmuka dari`abstract_rtsp_media_source`paket di AWS Panorama, dapat menerima beberapa aliran kamera sebagai masukan. Output node, yang referensi`hdmi_data_sink`, memberikan akses kode aplikasi ke buffer video yang output dari port HDMI alat.

Example graph.json — Abstrak node

```
{
    "nodeGraph": {
        "envelopeVersion": "2021-01-01",
        "packages": [
            {
                "name": "123456789012::SAMPLE_CODE",
                "version": "1.0"
            },
            {
                "name": "123456789012::SQUEEZENET_PYTORCH_V1",
                "version": "1.0"
            },
            {
                "name": "panorama::abstract_rtsp_media_source",
                "version": "1.0"
            },
            {
                "name": "panorama::hdmi_data_sink",
                "version": "1.0"
            }
        ],
        "nodes": [
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "decorator": {
                    "title": "IP camera",
                    "description": "Choose a camera stream."
                }
            },
```

```
            {
                "name": "output_node",
                "interface": "panorama::hdmi_data_sink.hdmi0"
            }
        ],
        "edges": [
            {
                "producer": "camera_node.video_out",
                "consumer": "code_node.video_in"
            },
            {
                "producer": "code_node.video_out",
                "consumer": "output_node.video_in"
            }
        ]
    }
}
```

Parameter aplikasi

Parameter adalah node yang memiliki tipe dasar dan dapat diganti selama penyebaran. Sebuah parameter dapat memiliki nilai default dandekorator, yang menginstruksikan pengguna aplikasi cara mengkonfigurasinya.

Tipe parameter

- `string`String. Misalnya, `DEBUG`.
- `int32`— Bilangan bulat. Misalnya, `20`
- `float32`Nomor floating point. Misalnya, `47.5`
- `boolean`—`true`atau`false`.

Contoh berikut menunjukkan dua parameter, string dan angka, yang dikirim ke node kode sebagai input.

Example graph.json — Parameter

```
       "nodes": [
           {
               "name": "detection_threshold",
               "interface": "float32",
               "value": 20.0,
               "overridable": true,
               "decorator": {
                   "title": "Threshold",
                   "description": "The minimum confidence percentage for a positive
classification."
               }
           },
           {
               "name": "log_level",
               "interface": "string",
               "value": "INFO",
               "overridable": true,
               "decorator": {
                   "title": "Logging level",
                   "description": "DEBUG, INFO, WARNING, ERROR, or CRITICAL."
               }
```

```
            }
            ...
        ],
        "edges": [
            {
                "producer": "detection_threshold",
                "consumer": "code_node.threshold"
            },
            {
                "producer": "log_level",
                "consumer": "code_node.log_level"
            }
            ...
        ]
    }
```

Anda dapat memodifikasi parameter secara langsung dalam manifes aplikasi, atau memberikan nilai baru pada waktu penyebaran dengan penggantian. Untuk informasi selengkapnya, lihat Konfigurasi Deploy-time dengan menimpa.

Konfigurasi Deploy-time dengan menimpa

Anda mengkonfigurasi parameter dan node abstrak selama penyebaran. Jika Anda menggunakan konsol AWS Panorama untuk menerapkan, Anda dapat menentukan nilai untuk setiap parameter dan memilih aliran kamera sebagai input. Jika Anda menggunakan AWS Panorama API untuk menerapkan aplikasi, Anda menentukan pengaturan ini dengan dokumen penggantian.

Dokumen menimpa serupa dalam struktur untuk manifes aplikasi. Untuk parameter dengan tipe dasar, Anda mendefinisikan sebuah node. Untuk aliran kamera, Anda menentukan node dan paket yang memetakan ke aliran kamera terdaftar. Kemudian Anda mendefinisikan override untuk setiap node yang menentukan node dari manifes aplikasi yang menggantikannya.

Example overrides.json

```
{
    "nodeGraphOverrides": {
        "nodes": [
            {
                "name": "my_camera",
                "interface": "123456789012::exterior-south.exterior-south"
            },
            {
                "name": "my_region",
                "interface": "string",
                "value": "us-east-1"
            }
        ],
        "packages": [
            {
                "name": "123456789012::exterior-south",
                "version": "1.0"
            }
        ],
        "nodeOverrides": [
            {
                "replace": "camera_node",
                "with": [
                    {
                        "name": "my_camera"
                    }
                ]
            },
```

```
            {
                "replace": "region",
                "with": [
                    {
                        "name": "my_region"
                    }
                ]
            }
        ],
        "envelopeVersion": "2021-01-01"
    }
}
```

Dalam contoh sebelumnya, dokumen mendefinisikan menimpa untuk satu parameter string dan node kamera abstrak. Parameter`node0verrides`memberitahu AWS Panorama node mana dalam dokumen ini menimpa yang dalam manifes aplikasi.

Membangun aplikasi AWS Panorama

Aplikasi berjalan diAWS PanoramaAlat untuk melakukan tugas visi komputer pada aliran video. Anda dapat membangun aplikasi visi komputer dengan menggabungkan kode Python dan model machine learning, dan menerapkannya keAWS PanoramaAppliance melalui internet. Aplikasi dapat mengirim video ke layar, atau menggunakan AWS SDK untuk mengirim hasil ke layanan AWS.

SEBUAHmodelmenganalisis gambar untuk mendeteksi orang, kendaraan, dan objek lainnya. Berdasarkan gambar yang telah dilihatnya selama pelatihan, model memberi tahu Anda apa pendapatnya, dan seberapa percaya diri itu dalam tebakannya. Anda dapat melatih model dengan data gambar Anda sendiri atau memulai dengan sampel.

Aplikasikodememproses gambar diam dari aliran kamera, mengirimkannya ke model, dan memproses hasilnya. Model mungkin mendeteksi beberapa objek dan mengembalikan bentuk dan lokasinya. Kode dapat menggunakan informasi ini untuk menambahkan teks atau grafik ke video, atau untuk mengirim hasil keAWSlayanan untuk penyimpanan atau pemrosesan lebih lanjut.

Untuk mendapatkan gambar dari stream, berinteraksi dengan model, dan output video, kode aplikasi menggunakansangAWS PanoramaAplikasi SDK. Aplikasi SDK adalah pustaka Python yang mendukung model yang dihasilkan dengan PyTorch, Apache MXNet, dan TensorFlow.

Topik

- Model visi komputer
- Membangun gambar aplikasi
- Memanggil layanan AWS dari kode aplikasi Anda
- SDK Aplikasi AWS Panorama
- Menjalankan beberapa thread
- Melayani lalu lintas masuk
- Menggunakan GPU
- Menyiapkan lingkungan pengembangan di Windows

Model visi komputer

Model visi komputer adalah program perangkat lunak yang dilatih untuk mendeteksi objek dalam gambar. Model belajar mengenali sekumpulan objek dengan terlebih dahulu menganalisis gambar objek tersebut melalui pelatihan. Model penglihatan komputer mengambil gambar sebagai input dan mengeluarkan informasi tentang objek yang dideteksi, seperti jenis objek dan lokasinya. AWS Panorama mendukung model visi komputer yang dibuat dengan Apache MXNetPyTorch, dan. TensorFlow

> **Note**
>
> Untuk daftar model pra-bangun yang telah diuji dengan AWS Panorama, lihat Kompatibilitas model.

Bagian

- Menggunakan model dalam kode
- Membangun model khusus
- Kemasan model
- Model pelatihan

Menggunakan model dalam kode

Model mengembalikan satu atau lebih hasil, yang dapat mencakup probabilitas untuk kelas terdeteksi, informasi lokasi, dan data lainnya.Contoh berikut menunjukkan cara menjalankan inferensi pada gambar dari aliran video dan mengirim output model ke fungsi pemrosesan.

Example application.py - Inferensi

```
    def process_media(self, stream):
        """Runs inference on a frame of video."""
        image_data = preprocess(stream.image,self.MODEL_DIM)
        logger.debug('Image data: {}'.format(image_data))
        # Run inference
        inference_start = time.time()
        inference_results = self.call({"data":image_data}, self.MODEL_NODE)
         # Log metrics
```

```
        inference_time = (time.time() - inference_start) * 1000
        if inference_time > self.inference_time_max:
            self.inference_time_max = inference_time
        self.inference_time_ms += inference_time
        # Process results (classification)
        self.process_results(inference_results, stream)
```

Contoh berikut menunjukkan fungsi yang memproses hasil dari model klasifikasi dasar. Model sampel mengembalikan array probabilitas, yang merupakan nilai pertama dan satu-satunya dalam array hasil.

Example application.py - Hasil pengolahan

```
    def process_results(self, inference_results, stream):
        """Processes output tensors from a computer vision model and annotates a video
frame."""
        if inference_results is None:
            logger.warning("Inference results are None.")
            return
        max_results = 5
        logger.debug('Inference results: {}'.format(inference_results))
        class_tuple = inference_results[0]
        enum_vals = [(i, val) for i, val in enumerate(class_tuple[0])]
        sorted_vals = sorted(enum_vals, key=lambda tup: tup[1])
        top_k = sorted_vals[::-1][:max_results]
        indexes =  [tup[0] for tup in top_k]

        for j in range(max_results):
            label = 'Class [%s], with probability %.3f.'% (self.classes[indexes[j]],
class_tuple[0][indexes[j]])
            stream.add_label(label, 0.1, 0.1 + 0.1*j)
```

Kode aplikasi menemukan nilai dengan probabilitas tertinggi dan memetakannya ke label dalam file sumber daya yang dimuat selama inisialisasi.

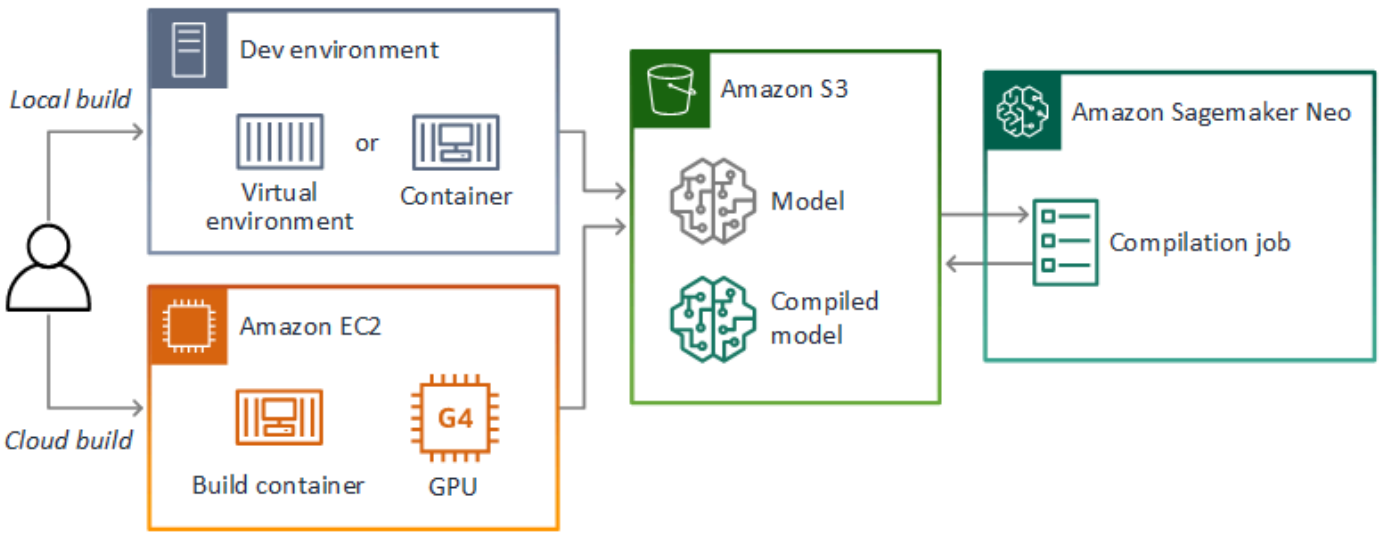
Membangun model khusus

Anda dapat menggunakan model yang Anda buatPyTorch, Apache MXNet, dan TensorFlow dalam aplikasi AWS Panorama. Sebagai alternatif untuk membangun dan melatih modelSageMaker, Anda dapat menggunakan model terlatih atau membangun dan melatih model Anda sendiri dengan kerangka kerja yang didukung dan mengekspornya di lingkungan lokal atau di Amazon EC2.

> **Note**
>
> Untuk detail tentang versi framework dan format file yang didukung oleh SageMaker Neo, lihat Frameworks yang Didukung dalam Panduan SageMaker Pengembang Amazon.

Repositori untuk panduan ini menyediakan contoh aplikasi yang menunjukkan alur kerja ini untuk model Keras dalam format. TensorFlow `SavedModel` Menggunakan TensorFlow 2 dan dapat berjalan secara lokal di lingkungan virtual atau dalam wadah Docker. Aplikasi contoh juga menyertakan templat dan skrip untuk membuat model pada instans Amazon EC2.

- Aplikasi sampel model khusus



AWS Panorama menggunakan SageMaker Neo untuk mengompilasi model untuk digunakan pada AWS Panorama Appliance. Untuk setiap kerangka kerja, gunakan format yang didukung oleh SageMaker Neo, dan paket model dalam `.tar.gz` arsip.

Untuk informasi selengkapnya, lihat Mengompilasi dan menerapkan model dengan Neo di Panduan SageMaker Pengembang Amazon.

Kemasan model

Paket model terdiri dari deskriptor, konfigurasi paket, dan arsip model. Seperti dalam paket gambar aplikasi, konfigurasi paket memberi tahu layanan AWS Panorama tempat model dan deskriptor disimpan di Amazon S3.

Example Paket/123456789012-squeezenet_pytorch-1.0/descriptor.json

```
{
    "mlModelDescriptor": {
        "envelopeVersion": "2021-01-01",
        "framework": "PYTORCH",
        "frameworkVersion": "1.8",
        "precisionMode": "FP16",
        "inputs": [
            {
                "name": "data",
                "shape": [
                    1,
                    3,
                    224,
                    224
                ]
            }
        ]
    }
}
```

> **Note**
>
> Tentukan versi utama dan minor versi kerangka kerja saja. Untuk daftar versi Apache MXNetPyTorch, dan TensorFlow versi yang didukung, lihat Framework yang didukung.

Untuk mengimpor model, gunakan perintah AWS Panorama Application CLI`import-raw-model`. Jika Anda membuat perubahan pada model atau deskriptornya, Anda harus menjalankan kembali perintah ini untuk memperbarui aset aplikasi. Untuk informasi selengkapnya, lihat Mengubah model visi komputer.

Untuk skema JSON file deskriptor, lihat AssetDescriptor.schema.json.

Model pelatihan

Saat Anda melatih model, gunakan gambar dari lingkungan target, atau dari lingkungan pengujian yang sangat mirip dengan lingkungan target. Pertimbangkan faktor-faktor berikut yang dapat mempengaruhi kinerja model:

- Pencahayaan - Jumlah cahaya yang dipantulkan oleh subjek menentukan seberapa banyak detail yang harus dianalisis model. Model yang dilatih dengan gambar subjek yang cukup terang mungkin tidak berfungsi dengan baik di lingkungan cahaya redup atau cahaya latar.
- Resolusi - Ukuran input model biasanya ditetapkan pada resolusi antara 224 dan 512 piksel lebar dalam rasio aspek persegi. Sebelum Anda melewati bingkai video ke model, Anda dapat menurunkan skala atau memotongnya agar sesuai dengan ukuran yang diperlukan.
- Distorsi gambar — Panjang fokus kamera dan bentuk lensa dapat menyebabkan gambar menunjukkan distorsi jauh dari tengah bingkai. Posisi kamera juga menentukan fitur subjek mana yang terlihat. Misalnya, kamera overhead dengan lensa sudut lebar akan menunjukkan bagian atas subjek ketika berada di tengah bingkai, dan tampilan miring sisi subjek saat bergerak lebih jauh dari tengah.

Untuk mengatasi masalah ini, Anda dapat memproses gambar sebelum mengirimnya ke model, dan melatih model pada variasi gambar yang lebih luas yang mencerminkan variasi di lingkungan dunia nyata. Jika model perlu beroperasi dalam situasi pencahayaan dan dengan berbagai kamera, Anda memerlukan lebih banyak data untuk pelatihan. Selain mengumpulkan lebih banyak gambar, Anda bisa mendapatkan lebih banyak data pelatihan dengan membuat variasi gambar yang ada yang miring atau memiliki pencahayaan yang berbeda.

Membangun gambar aplikasi

AWS Panorama Appliance menjalankan aplikasi sebagai sistem file kontainer yang diekspor dari gambar yang Anda buat. Anda menentukan dependensi dan sumber daya aplikasi Anda dalam Dockerfile yang menggunakan image dasar aplikasi AWS Panorama sebagai titik awal.

Untuk membuat image aplikasi, Anda menggunakan Docker dan AWS Panorama Application CLI. Contoh berikut dari contoh aplikasi panduan ini menunjukkan kasus penggunaan ini.

Example Paket/123456789012-sample_code-1.0/dockerfile

```
FROM public.ecr.aws/panorama/panorama-application
WORKDIR /panorama
COPY . .
RUN pip install --no-cache-dir --upgrade pip && \
    pip install --no-cache-dir -r requirements.txt
```

Instruksi Dockerfile berikut digunakan.

- `FROM`- Memuat gambar dasar aplikasi (`public.ecr.aws/panorama/panorama-application`).
- `WORKDIR`- Mengatur direktori kerja pada gambar. `/panorama`digunakan untuk kode aplikasi dan file terkait. Pengaturan ini hanya bertahan selama build dan tidak memengaruhi direktori kerja untuk aplikasi Anda saat runtime (`/`).
- `COPY`- Menyalin file dari jalur lokal ke jalur pada gambar. `COPY . .`menyalin file dalam direktori saat ini (direktori paket) ke direktori kerja pada gambar. Misalnya, kode aplikasi disalin dari `packages/123456789012-SAMPLE_CODE-1.0/application.py` ke`/panorama/application.py`.
- `RUN`- Menjalankan perintah shell pada gambar selama build. Sebuah `RUN` operasi tunggal dapat menjalankan beberapa perintah secara berurutan dengan menggunakan && antara perintah. Contoh ini memperbarui manajer `pip` paket dan kemudian menginstal pustaka yang tercantum dalam. `requirements.txt`

Anda dapat menggunakan instruksi lain, seperti `ADD` dan`ARG`, yang berguna pada waktu pembuatan. Petunjuk yang menambahkan informasi waktu proses ke wadah, sepertiENV, tidak berfungsi dengan AWS Panorama. AWS Panorama tidak menjalankan kontainer dari gambar. Ini hanya menggunakan gambar untuk mengekspor filesystem, yang ditransfer ke alat.

Menentukan dependensi

`requirements.txt`adalah file persyaratan Python yang menentukan perpustakaan yang digunakan oleh aplikasi. Contoh aplikasi menggunakan Open CV danAWS SDK for Python (Boto3).

Example Paket/123456789012-Sample_Code-1.0/Persyaratan.txt

```
boto3==1.24.*
opencv-python==4.6.*
```

`pip install`Perintah di Dockerfile menginstal pustaka ini ke `dist-packages` direktori Python di bawah`/usr/local/lib`, sehingga mereka dapat diimpor oleh kode aplikasi Anda.

Penyimpanan lokal

AWS Panorama menyimpan `/opt/aws/panorama/storage` direktori untuk penyimpanan aplikasi. Aplikasi Anda dapat membuat dan memodifikasi file di jalur ini. File yang dibuat di direktori penyimpanan tetap ada di reboot. Lokasi file sementara lainnya dihapus saat boot.

Membangun aset gambar

Saat Anda membuat gambar untuk paket aplikasi Anda dengan AWS Panorama Application CLI, CLI berjalan `docker build` di direktori paket. Ini membangun gambar aplikasi yang berisi kode aplikasi Anda. CLI kemudian membuat wadah, mengekspor filesystem, mengompresnya, dan menyimpannya di folder. `assets`

```
$ panorama-cli build-container --container-asset-name code_asset --package-path
 packages/123456789012-SAMPLE_CODE-1.0
docker build -t code_asset packages/123456789012-SAMPLE_CODE-1.0 --pull
docker export --output=code_asset.tar $(docker create code_asset:latest)
gzip -1 code_asset.tar
{
    "name": "code_asset",
    "implementations": [
        {
            "type": "container",
            "assetUri":
 "6f67xmpl32743ed0e60c151a02f2f0da1bf70a4ab9d83fe236fa32a6f9b9f808.tar.gz",
            "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
        }
```

```
    ]
}
Container asset for the package has been succesfully built at  /home/
user/aws-panorama-developer-guide/sample-apps/aws-panorama-sample/
assets/6f67xmpl32743ed0e60c151a02f2f0da1bf70a4ab9d83fe236fa32a6f9b9f808.tar.gz
```

Blok JSON dalam output adalah definisi aset yang ditambahkan CLI ke konfigurasi paket (`package.json`) dan mendaftar dengan layanan AWS Panorama. CLI juga menyalin file deskriptor, yang menentukan jalur ke skrip aplikasi (titik masuk aplikasi).

Example Paket/123456789012-sample_code-1.0/descriptor.json

```
{
    "runtimeDescriptor":
    {
        "envelopeVersion": "2021-01-01",
        "entry":
        {
            "path": "python3",
            "name": "/panorama/application.py"
        }
    }
}
```

Dalam folder aset, deskriptor dan gambar aplikasi diberi nama untuk checksum SHA-256 mereka. Nama ini digunakan sebagai pengenal unik untuk aset saat disimpan adalah Amazon S3.

Memanggil layanan AWS dari kode aplikasi Anda

Anda dapat menggunakan layananAWS SDK for Python (Boto) untuk memanggil AWS dari kode aplikasi Anda. Misalnya, jika model Anda mendeteksi sesuatu yang tidak biasa, Anda dapat memposting metrik ke Amazon CloudWatch, mengirim pemberitahuan dengan Amazon SNS, menyimpan gambar ke Amazon S3, atau memanggil fungsi Lambda untuk diproses lebih lanjut. Sebagian besar layanan AWS memiliki API publik yang dapat Anda gunakan dengan AWS SDK.

Alat ini tidak memiliki izin untuk mengakses layanan AWS apa pun secara default. Untuk memberikan izin, buat peran untuk aplikasi, dan tetapkan ke instance aplikasi selama penyebaran.

Bagian

- Menggunakan Amazon S3
- Menggunakan topikAWS IoT MQTT

Menggunakan Amazon S3

Sebagai tambahan, Anda dapat menggunakan Amazon S3 untuk menyimpan hasil pemrosesan dan data aplikasi lainnya.

```
import boto3
s3_client=boto3.client("s3")
s3_clients3.upload_file(data_file,
                    s3_bucket_name,
                    os.path.basename(data_file))
```

Menggunakan topikAWS IoT MQTT

Sebagai tambahan, Anda dapat menggunakan SDK for Python (Boto3) untuk mengirim pesan ke topik diAWS IoT. Pada contoh berikut, posting aplikasi ke topik yang dinamai sesuai nama alat, yang dapat Anda temukan di AWS IoTkonsol.

```
import boto3
iot_client=boto3.client('iot-data')
topic = "panorama/panorama_my-appliance_Thing_a01e373b"
iot_client.publish(topic=topic, payload="my message")
```

Pilih nama yang menunjukkan ID perangkat atau pengidentifikasi lain pilihan Anda. Untuk mempublikasikan pesan, aplikasi membutuhkan izin untuk meneleponiot:Publish.

Untuk memantau antrian MQTT

1. Buka halaman UjiAWS IoT konsol.
2. Untuk topik, masukkan nama topik. Sebagai contoh, panorama/panorama_my-appliance_Thing_a01e373b.
3. Pilih Berlangganan topik.

SDK Aplikasi AWS Panorama

AWS Panorama Application SDK adalah pustaka Python untuk mengembangkan aplikasi AWS Panorama. Dalamkode aplikasi, Anda menggunakan AWS Panorama Application SDK untuk memuat model visi komputer, menjalankan inferensi, dan output video ke monitor.

> **Note**
>
> Untuk memastikan bahwa Anda memiliki akses ke fungsionalitas terbaru AWS Panorama Application SDK,meng-upgrade perangkat lunak alat.

Untuk rincian tentang kelas yang SDK aplikasi mendefinisikan dan metode mereka, lihatReferensi SDK Aplikasi.

Bagian

- Menambahkan teks dan kotak ke output video

Menambahkan teks dan kotak ke output video

Dengan AWS Panorama SDK, Anda dapat menampilkan streaming video ke layar. Video dapat mencakup teks dan kotak yang menunjukkan output dari model, keadaan aplikasi saat ini, atau data lainnya.

Setiap objek di`video_in`array adalah gambar dari aliran kamera yang terhubung ke alat. Jenis objek ini`panoramasdk.media`. Ini memiliki metode untuk menambahkan teks dan kotak persegi panjang ke gambar, yang kemudian dapat Anda tetapkan ke`video_out`array.

Dalam contoh berikut, aplikasi sampel menambahkan label untuk masing-masing hasil. Setiap hasil diposisikan pada posisi kiri yang sama, tetapi pada ketinggian yang berbeda.

```
        for j in range(max_results):
            label = 'Class [%s], with probability %.3f.'% (self.classes[indexes[j]],
class_tuple[0][indexes[j]])
            stream.add_label(label, 0.1, 0.1 + 0.1*j)
```

Untuk menambahkan kotak ke gambar output, gunakan`add_rect`. Metode ini mengambil 4 nilai antara 0 dan 1, menunjukkan posisi sudut kiri atas dan kanan bawah kotak.

```
        w,h,c = stream.image.shape
        stream.add_rect(x1/w, y1/h, x2/w, y2/h)
```

Menjalankan beberapa thread

Anda dapat menjalankan logika aplikasi Anda pada thread pengolahan dan menggunakan thread lain untuk proses latar belakang lainnya. Misalnya, Anda dapat membuat threadmelayani lalu lintas HTTPuntuk debugging, atau thread yang memantau hasil inferensi dan mengirimkan data keAWS.

Untuk menjalankan beberapa thread, Anda menggunakanmodul threadingdari pustaka standar Python untuk membuat thread untuk setiap proses. Contoh berikut menunjukkan loop utama dari aplikasi sampel server debug, yang menciptakan objek aplikasi dan menggunakannya untuk menjalankantigautas.

Example Paket/123456789012-debug_server-1.0/application.py- Loop Utama

```
def main():
    panorama = panoramasdk.node()
    while True:
        try:
            # Instantiate application
            logger.info('INITIALIZING APPLICATION')
            app = Application(panorama)
            # Create threads for stream processing, debugger, and client
            app.run_thread = threading.Thread(target=app.run_cv)
            app.server_thread = threading.Thread(target=app.run_debugger)
            app.client_thread = threading.Thread(target=app.run_client)
            # Start threads
            logger.info('RUNNING APPLICATION')
            app.run_thread.start()
            logger.info('RUNNING SERVER')
            app.server_thread.start()
            logger.info('RUNNING CLIENT')
            app.client_thread.start()
            # Wait for threads to exit
            app.run_thread.join()
            app.server_thread.join()
            app.client_thread.join()
            logger.info('RESTARTING APPLICATION')
        except:
            logger.exception('Exception during processing loop.')
```

Ketika semua thread keluar, aplikasi restart sendiri. Parameter`run_cv`loop memproses gambar dari aliran kamera. Jika menerima sinyal untuk berhenti, itu mematikan proses debugger, yang

menjalankan server HTTP dan tidak dapat mematikan dirinya sendiri. Setiap utas harus menangani kesalahannya sendiri. Jika kesalahan tidak tertangkap dan dicatat, thread keluar diam-diam.

Example Paket/123456789012-debug_server-1.0/application.py- Loop pemrosesan

```
    # Processing loop
    def run_cv(self):
        """Run computer vision workflow in a loop."""
        logger.info("PROCESSING STREAMS")
        while not self.terminate:
            try:
                self.process_streams()
                # turn off debug logging after 15 loops
                if logger.getEffectiveLevel() == logging.DEBUG and self.frame_num ==
15:
                    logger.setLevel(logging.INFO)
            except:
                logger.exception('Exception on processing thread.')
        # Stop signal received
        logger.info("SHUTTING DOWN SERVER")
        self.server.shutdown()
        self.server.server_close()
        logger.info("EXITING RUN THREAD")
```

Thread berkomunikasi melalui aplikasi`self`objek. Untuk me-restart loop pemrosesan aplikasi, thread debugger memanggil`stop`metode. Metode ini menetapkan`terminate`atribut, yang sinyal benang lain untuk menutup.

Example Paket/123456789012-debug_server-1.0/application.py- Metode stop

```
    # Interrupt processing loop
    def stop(self):
        """Signal application to stop processing."""
        logger.info("STOPPING APPLICATION")
        # Signal processes to stop
        self.terminate = True
    # HTTP debug server
    def run_debugger(self):
        """Process debug commands from local network."""
        class ServerHandler(SimpleHTTPRequestHandler):
            # Store reference to application
            application = self
            # Get status
```

```
            def do_GET(self):
                """Process GET requests."""
                logger.info('Get request to {}'.format(self.path))
                if self.path == "/status":
                    self.send_200('OK')
                else:
                    self.send_error(400)
            # Restart application
            def do_POST(self):
                """Process POST requests."""
                logger.info('Post request to {}'.format(self.path))
                if self.path == '/restart':
                    self.send_200('OK')
                    ServerHandler.application.stop()
                else:
                    self.send_error(400)
```

Melayani lalu lintas masuk

Anda dapat memantau atau men-debug aplikasi secara lokal dengan menjalankan server HTTP bersama kode aplikasi Anda. Untuk melayani lalu lintas eksternal, Anda memetakan port pada AWS Panorama Appliance ke port pada wadah aplikasi Anda.

> **Important**
>
> Secara default, AWS Panorama Appliance tidak menerima lalu lintas masuk pada port apa pun. Membuka port pada alat memiliki risiko keamanan implisit. Saat Anda menggunakan fitur ini, Anda harus mengambil langkah tambahanamankan alat Anda dari lalu lintas eksternaldan mengamankan komunikasi antara klien resmi dan alat.
> Kode contoh yang disertakan dalam panduan ini adalah untuk tujuan demonstrasi dan tidak menerapkan otentikasi, otorisasi, atau enkripsi.

Anda dapat membuka port di kisaran 8000-9000 pada alat. Port ini, ketika dibuka, dapat menerima lalu lintas dari klien yang dapat dirutekan. Saat Anda menerapkan aplikasi, Anda menentukan port mana yang akan dibuka, dan memetakan port pada alat ke port pada wadah aplikasi Anda. Perangkat lunak alat meneruskan lalu lintas ke wadah, dan mengirimkan tanggapan kembali ke pemohon. Permintaan diterima pada port alat yang Anda tentukan dan tanggapan keluar pada port fana acak.

Mengonfigurasi port masuk

Anda menentukan pemetaan port di tiga tempat dalam konfigurasi aplikasi Anda. Paket kodenya`package.json`, Anda menentukan port yang didengarkan oleh node kode dalam`network`blok Contoh berikut menyatakan bahwa node mendengarkan pada port 80.

Example Paket/123456789012-debug_server-1.0/package.json

```
        "outputs": [
            {
                "description": "Video stream output",
                "name": "video_out",
                "type": "media"
            }
        ],
        "network": {
            "inboundPorts": [
```

```
                    {
                        "port": 80,
                        "description": "http"
                    }
                ]
            }
```

Dalam manifes aplikasi, Anda mendeklarasikan aturan perutean yang memetakan port pada alat ke port pada wadah kode aplikasi. Contoh berikut menambahkan aturan yang memetakan port 8080 pada perangkat ke port 80 padacode_nodekontainer

Example grafik/my-app/graph.json

```
        {
            "producer": "model_input_width",
            "consumer": "code_node.model_input_width"
        },
        {
            "producer": "model_input_order",
            "consumer": "code_node.model_input_order"
        }
    ],
    "networkRoutingRules": [
        {
            "node": "code_node",
            "containerPort": 80,
            "hostPort": 8080,
            "decorator": {
                "title": "Listener port 8080",
                "description": "Container monitoring and debug."
            }
        }
    ]
```

Saat menerapkan aplikasi, Anda menentukan aturan yang sama di konsol AWS Panorama, atau dengan dokumen penggantian yang diteruskan keCreateApplicationInstanceAPI Anda harus menyediakan konfigurasi ini pada waktu penyebaran untuk mengonfirmasi bahwa Anda ingin membuka port pada alat.

Example grafik/my-app/override.json

```
        {
```

```
                "replace": "camera_node",
                "with": [
                    {
                        "name": "exterior-north"
                    }
                ]
            }
        ],
        "networkRoutingRules":[
            {
                "node": "code_node",
                "containerPort": 80,
                "hostPort": 8080
            }
        ],
        "envelopeVersion": "2021-01-01"
    }
}
```

Jika port perangkat yang ditentukan dalam manifes aplikasi sedang digunakan oleh aplikasi lain, Anda dapat menggunakan dokumen ganti untuk memilih port yang berbeda.

Melayani lalu lintas

Dengan port terbuka pada wadah, Anda dapat membuka soket atau menjalankan server untuk menangani permintaan masuk. Parameter`debug-server`sampel menunjukkan implementasi dasar dari server HTTP yang berjalan bersama kode aplikasi visi komputer.

> **Important**
>
> Implementasi sampel tidak aman untuk penggunaan produksi. Untuk menghindari membuat perangkat Anda rentan terhadap serangan, Anda harus menerapkan kontrol keamanan yang sesuai dalam kode dan konfigurasi jaringan Anda.

Example Paket/123456789012-debug_server-1.0/application.py— Server HTTP

```
    # HTTP debug server
    def run_debugger(self):
        """Process debug commands from local network."""
        class ServerHandler(SimpleHTTPRequestHandler):
```

```
            # Store reference to application
            application = self
            # Get status
            def do_GET(self):
                """Process GET requests."""
                logger.info('Get request to {}'.format(self.path))
                if self.path == '/status':
                    self.send_200('OK')
                else:
                    self.send_error(400)
            # Restart application
            def do_POST(self):
                """Process POST requests."""
                logger.info('Post request to {}'.format(self.path))
                if self.path == '/restart':
                    self.send_200('OK')
                    ServerHandler.application.stop()
                else:
                    self.send_error(400)
            # Send response
            def send_200(self, msg):
                """Send 200 (success) response with message."""
                self.send_response(200)
                self.send_header('Content-Type', 'text/plain')
                self.end_headers()
                self.wfile.write(msg.encode('utf-8'))
        try:
            # Run HTTP server
            self.server = HTTPServer(("", self.CONTAINER_PORT), ServerHandler)
            self.server.serve_forever(1)
            # Server shut down by run_cv loop
            logger.info("EXITING SERVER THREAD")
        except:
            logger.exception('Exception on server thread.')
```

Server menerima permintaan GET di`/status`path untuk mengambil beberapa informasi tentang aplikasi. Hal ini juga menerima permintaan POST untuk`/restart`untuk me-restart aplikasi.

Untuk mendemonstrasikan fungsionalitas ini, aplikasi sampel menjalankan klien HTTP pada thread terpisah. Klien memanggil`/status`jalur melalui jaringan lokal tak lama setelah startup, dan restart aplikasi beberapa menit kemudian.

Example Paket/123456789012-debug_server-1.0/application.py— Klien HTTP

```
    # HTTP test client
    def run_client(self):
        """Send HTTP requests to device port to demnostrate debug server functions."""
        def client_get():
            """Get container status"""
            r = requests.get('http://{}:{}/status'.format(self.device_ip,
 self.DEVICE_PORT))
            logger.info('Response: {}'.format(r.text))
            return
        def client_post():
            """Restart application"""
            r = requests.post('http://{}:{}/restart'.format(self.device_ip,
 self.DEVICE_PORT))
            logger.info('Response: {}'.format(r.text))
            return
        # Call debug server
        while not self.terminate:
            try:
                time.sleep(30)
                client_get()
                time.sleep(300)
                client_post()
            except:
                logger.exception('Exception on client thread.')
        # stop signal received
        logger.info("EXITING CLIENT THREAD")
```

Loop utama mengelola thread dan restart aplikasi ketika mereka keluar.

Example Paket/123456789012-debug_server-1.0/application.py— Loop Utama

```
def main():
    panorama = panoramasdk.node()
    while True:
        try:
            # Instantiate application
            logger.info('INITIALIZING APPLICATION')
            app = Application(panorama)
            # Create threads for stream processing, debugger, and client
            app.run_thread = threading.Thread(target=app.run_cv)
            app.server_thread = threading.Thread(target=app.run_debugger)
```

```
            app.client_thread = threading.Thread(target=app.run_client)
            # Start threads
            logger.info('RUNNING APPLICATION')
            app.run_thread.start()
            logger.info('RUNNING SERVER')
            app.server_thread.start()
            logger.info('RUNNING CLIENT')
            app.client_thread.start()
            # Wait for threads to exit
            app.run_thread.join()
            app.server_thread.join()
            app.client_thread.join()
            logger.info('RESTARTING APPLICATION')
        except:
            logger.exception('Exception during processing loop.')
```

Untuk men-deploy aplikasi sampel, lihatpetunjuk dalam panduan ini GitHub repositori.

Menggunakan GPU

Anda dapat mengakses prosesor grafis (GPU) di AWS Panorama Appliance untuk menggunakan library yang dipercepat GPU, atau menjalankan model machine learning dalam kode aplikasi Anda. Untuk mengaktifkan akses GPU, Anda menambahkan akses GPU sebagai persyaratan ke konfigurasi paket setelah membuat kontainer kode aplikasi.

> **Important**
>
> Jika Anda mengaktifkan akses GPU, Anda tidak dapat menjalankan node model dalam aplikasi apa pun di alat. Untuk tujuan keamanan, akses GPU dibatasi saat alat menjalankan model yang dikompilasi dengan SageMaker Neo. Dengan akses GPU, Anda harus menjalankan model Anda di node kode aplikasi, dan semua aplikasi pada perangkat berbagi akses ke GPU.

Untuk mengaktifkan akses GPU untuk aplikasi Anda, perbarui konfigurasi paket setelah Anda membangun paket dengan AWS Panorama Application CLI. Contoh berikut menunjukkan `requirements` blok yang menambahkan akses GPU ke node kode aplikasi.

Example package.json dengan persyaratan blok

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [
            {
                "name": "code_asset",
                "implementations": [
                    {
                        "type": "container",
                        "assetUri":
 "eba3xmpl71aa387e8f89be9a8c396416cdb80a717bb32103c957a8bf41440b12.tar.gz",
                        "descriptorUri":
 "4abdxmpl5a6f047d2b3047adde44704759d13f0126c00ed9b4309726f6bb43400ba9.json",
                        "requirements": [
                            {
                                "type": "hardware_access",
```

```
                            "inferenceAccelerators": [
                                {
                                    "deviceType": "nvhost_gpu",
                                    "sharedResourcePolicy": {
                                        "policy" : "allow_all"
                                    }
                                }
                            ]
                        }
                    ]
                }
            ]
        }
    ],
    "interfaces": [
    ...
```

Perbarui konfigurasi paket antara langkah pembuatan dan pengemasan dalam alur kerja pengembangan Anda.

Menyebarkan aplikasi dengan akses GPU

1. Untuk membangun wadah aplikasi, gunakan `build-container` perintah.

   ```
   $ panorama-cli build-container --container-asset-name code_asset --package-path
    packages/123456789012-SAMPLE_CODE-1.0
   ```

2. Tambahkan `requirements` blok ke konfigurasi paket.
3. Untuk mengunggah aset kontainer dan konfigurasi paket, gunakan `package-application` perintah.

   ```
   $ panorama-cli package-application
   ```

4. Men-deploy aplikasi.

Untuk contoh aplikasi yang menggunakan akses GPU, kunjungi aws-panorama-samplesGitHubrepositori.

Menyiapkan lingkungan pengembangan di Windows

Untuk membangun aplikasi AWS Panorama, Anda menggunakan Docker, alat baris perintah, dan Python. Di Windows, Anda dapat mengatur lingkungan pengembangan dengan menggunakan Docker Desktop dengan Windows Subsystem untuk Linux dan Ubuntu. Tutorial ini memandu Anda melalui proses penyiapan untuk lingkungan pengembangan yang telah diuji dengan alat AWS Panorama dan aplikasi sampel.

Bagian

- Prasyarat
- Instal WSL 2 dan Ubuntu
- Instal Docker
- Mengonfigurasi Ubuntu
- Langkah selanjutnya

Prasyarat

Untuk mengikuti tutorial ini, Anda memerlukan versi Windows yang mendukung Windows Subsystem untuk Linux 2 (WSL 2).

- Windows 10 versi 1903 dan lebih tinggi (Build 18362 dan lebih tinggi) atau Windows 11
- Fitur Windows
  - Subsistem Windows untuk Linux
  - Hyper-V
  - Platform mesin virtual

Tutorial ini dikembangkan dengan versi perangkat lunak berikut.

- Ubuntu 20.04
- Python 3.8.5
- Docker 20.10.8

Instal WSL 2 dan Ubuntu

Jika Anda memiliki Windows 10 versi 2004 dan lebih tinggi (Build 19041 dan lebih tinggi), Anda dapat menginstal WSL 2 dan Ubuntu 20.04 dengan perintah PowerShell berikut.

```
> wsl --install -d Ubuntu-20.04
```

Untuk versi Windows yang lebih tua, ikuti petunjuk dalam dokumentasi WSL 2: Langkah instalasi manual untuk versi yang lebih lama

Instal Docker

Untuk menginstal Docker Desktop, download dan jalankan paket installer darihub.docker.com. Jika Anda mengalami masalah, ikuti petunjuk di situs Docker: Docker Desktop WSL 2 backend.

Jalankan Docker Desktop dan ikuti tutorial yang dijalankan pertama untuk membangun sebuah wadah contoh.

> **Note**
>
> Docker Desktop hanya memungkinkan Docker dalam distribusi default. Jika Anda memiliki distribusi Linux lain yang diinstal sebelum menjalankan tutorial ini, aktifkan Docker di distribusi Ubuntu yang baru diinstal di menu pengaturan Docker Desktop di bawahSumber Daya,Integrasi WSL.

Mengonfigurasi Ubuntu

Anda sekarang dapat menjalankan perintah Docker di mesin virtual Ubuntu Anda. Untuk membuka terminal baris perintah, jalankan distribusi dari menu awal. Pertama kali Anda menjalankannya, Anda mengkonfigurasi nama pengguna dan kata sandi yang dapat Anda gunakan untuk menjalankan perintah administrator.

Untuk menyelesaikan konfigurasi lingkungan pengembangan Anda, perbarui perangkat lunak mesin virtual dan instal alat.

Untuk mengkonfigurasi mesin virtual

1. Perbarui perangkat lunak yang disertakan dengan Ubuntu.

```
$ sudo apt update && sudo apt upgrade -y && sudo apt autoremove
```

2. Instal alat pengembangan dengan apt.

```
$ sudo apt install unzip python3-pip
```

3. Instal pustaka Python dengan pip.

```
$ pip3 install awscli panoramacli
```

4. Buka terminal baru, lalu jalankan`aws configure`untuk mengonfigurasiAWS CLI.

```
$ aws configure
```

   Jika Anda tidak memiliki access key, Anda dapat membuatnya diKonsol IAM.

Akhirnya, download dan impor aplikasi sampel.

Untuk mendapatkan aplikasi sampel

1. Download dan ekstrak sampel aplikasi.

```
$ wget https://github.com/awsdocs/aws-panorama-developer-guide/releases/download/
v1.0-ga/aws-panorama-sample.zip
$ unzip aws-panorama-sample.zip
$ cd aws-panorama-sample
```

2. Jalankan skrip yang disertakan untuk menguji kompilasi, membangun wadah aplikasi, dan unggah paket ke AWS Panorama.

```
aws-panorama-sample$ ./0-test-compile.sh
aws-panorama-sample$ ./1-create-role.sh
aws-panorama-sample$ ./2-import-app.sh
aws-panorama-sample$ ./3-build-container.sh
aws-panorama-sample$ ./4-package-app.sh
```

AWS Panorama Application CLI mengunggah paket dan mendaftarkannya dengan layanan AWS Panorama. Sekarang Anda dapatmen-deploy aplikasi sampeldengan AWS Panorama console.

Langkah selanjutnya

Untuk menjelajahi dan mengedit file proyek, Anda dapat menggunakan File Explorer atau lingkungan pengembangan terpadu (IDE) yang mendukung WSL.

Untuk mengakses sistem file mesin virtual, buka File explorer dan masukkan`\\wsl$`di bilah navigasi. Direktori ini berisi tautan ke sistem file mesin virtual (`Ubuntu-20.04`) dan sistem file untuk data Docker. Di bawah`Ubuntu-20.04`, direktori pengguna Anda berada di`home\`*`username`*.

> **Note**
>
> Untuk mengakses file dalam instalasi Windows Anda dari dalam Ubuntu, arahkan ke`/mnt/c`direktori. Misalnya, Anda dapat mencantumkan file di direktori unduhan Anda dengan menjalankan`ls /mnt/c/Users/`*`windows-username`*`/Downloads`.

Dengan Visual Studio Code, Anda dapat mengedit kode aplikasi di lingkungan pengembangan Anda dan menjalankan perintah dengan terminal terintegrasi. Untuk menginstal Visual Studio Code, kunjungicode.visualstudio.com. Setelah instalasi, tambahkanWSLekstensi.

Terminal Windows adalah alternatif untuk terminal Ubuntu standar yang Anda telah menjalankan perintah di. Ini mendukung beberapa tab dan dapat menjalankan PowerShell, Command Prompt, dan terminal untuk berbagai Linux lainnya yang Anda instal. Mendukung copy dan paste dengan`Ctrl+C`dan`Ctrl+V`, URL yang dapat diklik, dan perbaikan berguna lainnya. Untuk menginstal Windows Terminal, kunjungimicrosoft.com.

API AWS Panorama

Anda dapat menggunakan API publik layanan AWS Panorama untuk mengotomatiskan alur kerja manajemen perangkat dan aplikasi. DenganAWS SDKAWS Command Line Interface atau, Anda dapat mengembangkan skrip atau aplikasi yang mengelola sumber daya dan penerapan. GitHub Repositori panduan ini meliputi skrip yang dapat Anda gunakan sebagai titik awal untuk kode Anda sendiri.

- aws-panorama-developer-guide/util-skrip

Bagian

- Mengotomatiskan pendaftaran perangkat
- Mengelola peralatan dengan API AWS Panorama
- Mengotomatiskan penerapan aplikasi
- Mengelola aplikasi dengan AWS Panorama API
- Menggunakan titik akhir VPC

Mengotomatiskan pendaftaran perangkat

Untuk menyediakan alat, gunakanProvisionDeviceAPI Respons mencakup file ZIP dengan konfigurasi perangkat dan kredensi sementara. Decode file dan simpan dalam arsip dengan awalan`certificates-omni_`.

Example provision-device.sh

```
if [[ $# -eq 1 ]] ; then
    DEVICE_NAME=$1
else
    echo "Usage: ./provision-device.sh <device-name>"
    exit 1
fi
CERTIFICATE_BUNDLE=certificates-omni_${DEVICE_NAME}.zip
aws panorama provision-device --name ${DEVICE_NAME} --output text --query Certificates
 | base64 --decode > ${CERTIFICATE_BUNDLE}
echo "Created certificate bundle ${CERTIFICATE_BUNDLE}"
```

Kredensi dalam arsip konfigurasi berakhir setelah 5 menit. Transfer arsip ke alat Anda dengan drive USB yang disertakan.

Untuk mendaftarkan kamera, gunakanCreateNodeFromTemplateJobAPI API ini mengambil peta parameter template untuk nama pengguna, kata sandi, dan URL kamera. Anda dapat memformat peta ini sebagai dokumen JSON dengan menggunakan manipulasi string Bash.

Example register-camera.sh

```
if [[ $# -eq 3 ]] ; then
    NAME=$1
    USERNAME=$2
    URL=$3
else
    echo "Usage: ./register-camera.sh <stream-name> <username> <rtsp-url>"
    exit 1
fi
echo "Enter camera stream password: "
read PASSWORD
TEMPLATE='{"Username":"MY_USERNAME","Password":"MY_PASSWORD","StreamUrl": "MY_URL"}'
TEMPLATE=${TEMPLATE/MY_USERNAME/$USERNAME}
TEMPLATE=${TEMPLATE/MY_PASSWORD/$PASSWORD}
TEMPLATE=${TEMPLATE/MY_URL/$URL}
```

```
echo ${TEMPLATE}
JOB_ID=$(aws panorama create-node-from-template-job --template-type RTSP_CAMERA_STREAM
 --output-package-name ${NAME} --output-package-version "1.0" --node-name ${NAME} --
template-parameters "${TEMPLATE}" --output text)
```

Atau, Anda dapat memuat konfigurasi JSON dari file.

```
--template-parameters file://camera-template.json
```

Mengelola peralatan dengan API AWS Panorama

Anda dapat mengotomatiskan tugas manajemen alat dengan AWS Panorama API.

Lihat perangkat

Untuk mendapatkan daftar peralatan dengan ID perangkat, gunakan ListDevicesAPI.

```
$ aws panorama list-devices
    "Devices": [
        {
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
            "Name": "my-appliance",
            "CreatedTime": 1652409973.613,
            "ProvisioningStatus": "SUCCEEDED",
            "LastUpdatedTime": 1652410973.052,
            "LeaseExpirationTime": 1652842940.0
        }
    ]
}
```

Untuk mendapatkan detail lebih lanjut tentang alat, gunakan DescribeDeviceAPI.

```
$ aws panorama describe-device --device-id device-4tafxmplhtmzabv5lsacba4ere
{
    "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
    "Name": "my-appliance",
    "Arn": "arn:aws:panorama:us-west-2:123456789012:device/
device-4tafxmplhtmzabv5lsacba4ere",
    "Type": "PANORAMA_APPLIANCE",
    "DeviceConnectionStatus": "ONLINE",
    "CreatedTime": 1648232043.421,
    "ProvisioningStatus": "SUCCEEDED",
    "LatestSoftware": "4.3.55",
    "CurrentSoftware": "4.3.45",
    "SerialNumber": "GFXMPL0013023708",
    "Tags": {},
    "CurrentNetworkingStatus": {
        "Ethernet0Status": {
            "IpAddress": "192.168.0.1/24",
            "ConnectionStatus": "CONNECTED",
            "HwAddress": "8C:XM:PL:60:C5:88"
        },
```

```
        "Ethernet1Status": {
            "IpAddress": "--",
            "ConnectionStatus": "NOT_CONNECTED",
            "HwAddress": "8C:XM:PL:60:C5:89"
        }
    },
    "LeaseExpirationTime": 1652746098.0
}
```

Perbarui Perangkat Lunak Perlengkapan

Jika`LatestSoftware` versi lebih baru dari`CurrentSoftware`, Anda dapat meng-upgrade perangkat. Gunakan CreateJobForDevicesAPI untuk membuat pekerjaan pembaruan over-the-air (OTA).

```
$ aws panorama create-job-for-devices --device-ids device-4tafxmplhtmzabv5lsacba4ere \
  --device-job-config '{"OTAJobConfig": {"ImageVersion": "4.3.55"}}' --job-type OTA
{
    "Jobs": [
        {
            "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere"
        }
    ]
}
```

Dalam skrip, Anda dapat mengisi bidang versi gambar di file konfigurasi pekerjaan dengan manipulasi string Bash.

Example check-updates.sh

```
apply_update() {
    DEVICE_ID=$1
    NEW_VERSION=$2
    CONFIG='{"OTAJobConfig": {"ImageVersion": "NEW_VERSION"}}'
    CONFIG=${CONFIG/NEW_VERSION/$NEW_VERSION}
    aws panorama create-job-for-devices --device-ids ${DEVICE_ID} --device-job-config
 "${CONFIG}" --job-type OTA
}
```

Alat mengunduh versi perangkat lunak yang ditentukan dan memperbarui itu sendiri. Perhatikan kemajuan pembaruan dengan DescribeDeviceJobAPI.

```
$ aws panorama describe-device-job --job-id device-4tafxmplhtmzabv5lsacba4ere-0
{
    "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
    "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
    "DeviceArn": "arn:aws:panorama:us-west-2:559823168634:device/
device-4tafxmplhtmzabv5lsacba4ere",
    "DeviceName": "my-appliance",
    "DeviceType": "PANORAMA_APPLIANCE",
    "ImageVersion": "4.3.55",
    "Status": "REBOOTING",
    "CreatedTime": 1652410232.465
}
```

Untuk mendapatkan daftar semua pekerjaan yang berjalan, gunakan ListDevicesJobs.

```
$ aws panorama list-devices-jobs
{
    "DeviceJobs": [
        {
            "DeviceName": "my-appliance",
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
            "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
            "CreatedTime": 1652410232.465
        }
    ]
}
```

Untuk contoh skrip yang memeriksa dan menerapkan pembaruan, lihat check-updates.sh di GitHub repositori panduan ini.

Perlengkapan boot ulang

Untuk me-reboot alat, gunakan CreateJobForDevicesAPI.

```
$ aws panorama create-job-for-devices --device-ids device-4tafxmplhtmzabv5lsacba4ere --
job-type REBOOT
{
    "Jobs": [
        {
            "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere"
        }
```

```
    ]
}
```

Dalam skrip, Anda bisa mendapatkan daftar perangkat dan memilih salah satu untuk reboot secara interaktif.

Example reboot-device.sh - penggunaan

```
$ ./reboot-device.sh
Getting devices...
0: device-53amxmplyn3gmj72epzanacniy     my-se70-1
1: device-6talxmpl5mmik6qh5moba6jium     my-manh-24
Choose a device
1
Reboot device device-6talxmpl5mmik6qh5moba6jium? (y/n)y
{
    "Jobs": [
        {
            "DeviceId": "device-6talxmpl5mmik6qh5moba6jium",
            "JobId": "device-6talxmpl5mmik6qh5moba6jium-8"
        }
    ]
}
```

Mengotomatiskan penerapan aplikasi

Untuk menerapkan aplikasi, Anda menggunakan AWS Panorama Application CLI danAWS Command Line Interface. Setelah membangun wadah aplikasi, Anda mengunggahnya dan aset lainnya ke titik akses Amazon S3. Anda kemudian menyebarkan aplikasi dengan`CreateApplicationInstance`API.

Untuk konteks dan instruksi lebih lanjut untuk menggunakan skrip yang ditampilkan, ikuti petunjuk di`contoh aplikasi README`.

Bagian

- Bangun wadah
- Unggah kontainer dan daftarkan node
- Deploy aplikasi
- Pantau penyebaran

Bangun wadah

Untuk membangun wadah aplikasi, gunakan`build-container`perintah. Perintah ini membangun wadah Docker dan menyimpannya sebagai sistem file terkompresi di`assets`folder.

Example 3-build-container.sh

```
CODE_PACKAGE=SAMPLE_CODE
ACCOUNT_ID=$(aws sts get-caller-identity --output text --query 'Account')
panorama-cli build-container --container-asset-name code_asset --package-path packages/
${ACCOUNT_ID}-${CODE_PACKAGE}-1.0
```

Anda juga dapat menggunakan penyelesaian baris perintah untuk mengisi argumen path dengan mengetikkan bagian dari jalur, dan kemudian menekanTAB.

```
$ panorama-cli build-container --package-path packages/TAB
```

Unggah kontainer dan daftarkan node

Untuk mengunggah aplikasi, gunakan`package-application`perintah. Perintah ini mengunggah aset dari`assets`folder ke titik akses Amazon S3 yang dikelola AWS Panorama.

Example 4-package-app.sh

```
panorama-cli package-application
```

AWS Panorama Application CLI mengunggah aset kontainer dan deskriptor yang direferensikan oleh konfigurasi paket (`package.json`) di setiap paket, dan mendaftarkan paket sebagai node di AWS Panorama. Anda kemudian merujuk ke node ini dalam manifes aplikasi Anda (`graph.json`) untuk menyebarkan aplikasi.

Deploy aplikasi

Untuk menyebarkan aplikasi, Anda menggunakanCreateApplicationInstanceAPI. Tindakan ini mengambil parameter berikut, antara lain.

- `ManifestPayload`- Manifes aplikasi (`graph.json`) yang mendefinisikan node aplikasi, paket, tepi, dan parameter.
- `ManifestOverridesPayload`- Manifes kedua yang menimpa parameter di yang pertama. Manifes aplikasi dapat dianggap sebagai sumber daya statis di sumber aplikasi, di mana manifes override menyediakan setelan waktu penerapan yang menyesuaikan penerapan.
- `DefaultRuntimeContextDevice`- Perangkat target.
- `RuntimeRoleArn`— ARN peran IAM yang digunakan aplikasi untuk mengakses layanan dan sumber daya AWS.
- `ApplicationInstanceIdToReplace`- ID instance aplikasi yang ada untuk dihapus dari perangkat.

Manifes dan override payload adalah dokumen JSON yang harus disediakan sebagai nilai string yang bersarang di dalam dokumen lain. Untuk melakukan ini, skrip memuat manifes dari file sebagai string dan menggunakanalat jquntuk membangun dokumen bersarang.

Example 5-deploy.sh— menyusun manifes

```
GRAPH_PATH="graphs/my-app/graph.json"
OVERRIDE_PATH="graphs/my-app/override.json"
# application manifest
GRAPH=$(cat ${GRAPH_PATH} | tr -d '\n' | tr -d '[:blank:]')
MANIFEST="$(jq --arg value "${GRAPH}" '.PayloadData="\($value)"' <<< {})"
# manifest override
```

```
OVERRIDE=$(cat ${OVERRIDE_PATH} | tr -d '\n' | tr -d '[:blank:]')
MANIFEST_OVERRIDE="$(jq --arg value "${OVERRIDE}" '.PayloadData="\($value)"' <<< {})"
```

Skrip deploy menggunakanListDevicesAPI untuk mendapatkan daftar perangkat terdaftar di Wilayah saat ini, dan menyimpan pilihan pengguna ke file lokal untuk penyebaran berikutnya.

Example 5-deploy.sh— menemukan perangkat

```
    echo "Getting devices..."
    DEVICES=$(aws panorama list-devices)
    DEVICE_NAMES=($((echo ${DEVICES} | jq -r '.Devices |=sort_by(.LastUpdatedTime) |
 [.Devices[].Name] | @sh') | tr -d \'\"))
    DEVICE_IDS=($((echo ${DEVICES} | jq -r '.Devices |=sort_by(.LastUpdatedTime) |
 [.Devices[].DeviceId] | @sh') | tr -d \'\"))
    for (( c=0; c<${#DEVICE_NAMES[@]}; c++ ))
    do
        echo "${c}: ${DEVICE_IDS[${c}]}     ${DEVICE_NAMES[${c}]}"
    done
    echo "Choose a device"
    read D_INDEX
    echo "Deploying to device ${DEVICE_IDS[${D_INDEX}]}"
    echo -n ${DEVICE_IDS[${D_INDEX}]} > device-id.txt
    DEVICE_ID=$(cat device-id.txt)
```

Peran aplikasi dibuat oleh skrip lain (1-create-role.sh). Skrip deploy mendapatkan ARN dari peran iniAWS CloudFormation. Jika aplikasi sudah diterapkan ke perangkat, skrip mendapatkan ID dari instance aplikasi itu dari file lokal.

Example 5-deploy.sh- peran ARN dan argumen pengganti

```
# application role
STACK_NAME=panorama-${NAME}
ROLE_ARN=$(aws cloudformation describe-stacks --stack-name panorama-${PWD##*/} --query
 'Stacks[0].Outputs[?OutputKey==`roleArn`].OutputValue' --output text)
ROLE_ARG="--runtime-role-arn=${ROLE_ARN}"

# existing application instance id
if [ -f "application-id.txt" ]; then
    EXISTING_APPLICATION=$(cat application-id.txt)
    REPLACE_ARG="--application-instance-id-to-replace=${EXISTING_APPLICATION}"
    echo "Replacing application instance ${EXISTING_APPLICATION}"
```

```
fi
```

Akhirnya, skrip menyatukan semua bagian untuk membuat instance aplikasi dan menyebarkan aplikasi ke perangkat. Layanan merespons dengan ID instance yang disimpan skrip untuk digunakan nanti.

Example 5-deploy.sh— menyebarkan aplikasi

```
APPLICATION_ID=$(aws panorama create-application-instance ${REPLACE_ARG} --manifest-
payload="${MANIFEST}" --default-runtime-context-device=${DEVICE_ID} --name=${NAME}
 --description="command-line deploy" --tags client=sample --manifest-overrides-
payload="${MANIFEST_OVERRIDE}" ${ROLE_ARG} --output text)
echo "New application instance ${APPLICATION_ID}"
echo -n $APPLICATION_ID > application-id.txt
```

Pantau penyebaran

Untuk memantau penyebaran, gunakanListApplicationInstancesAPI. Skrip monitor mendapatkan ID perangkat dan ID instance aplikasi dari file di direktori aplikasi dan menggunakannya untuk membuat perintah CLI. Kemudian panggilan dalam satu lingkaran.

Example 6-monitor-deployment.sh

```
APPLICATION_ID=$(cat application-id.txt)
DEVICE_ID=$(cat device-id.txt)
QUERY="ApplicationInstances[?ApplicationInstanceId==\`APPLICATION_ID\`]"
QUERY=${QUERY/APPLICATION_ID/$APPLICATION_ID}
MONITOR_CMD="aws panorama list-application-instances --device-id ${DEVICE_ID} --query
 ${QUERY}"
MONITOR_CMD=${MONITOR_CMD/QUERY/$QUERY}
while true; do
    $MONITOR_CMD
    sleep 60
done
```

Saat penerapan selesai, Anda dapat melihat log dengan memanggil AmazonCloudWatchLog API. Skrip log tampilan menggunakanCloudWatchLog`GetLogEvents`API.

Example view-logs.sh

```
GROUP="/aws/panorama/devices/MY_DEVICE_ID/applications/MY_APPLICATION_ID"
```

```
GROUP=${GROUP/MY_DEVICE_ID/$DEVICE_ID}
GROUP=${GROUP/MY_APPLICATION_ID/$APPLICATION_ID}
echo "Getting logs for group ${GROUP}."
#set -x
while true
do
    LOGS=$(aws logs get-log-events --log-group-name ${GROUP} --log-stream-name
 code_node --limit 150)
    readarray -t ENTRIES < <(echo $LOGS | jq -c '.events[].message')
    for ENTRY in "${ENTRIES[@]}"; do
        echo "$ENTRY" | tr -d \"
    done
    sleep 20
done
```

Mengelola aplikasi dengan AWS Panorama API

Anda dapat memantau dan mengelola aplikasi dengan AWS Panorama API.

Melihat aplikasi

Untuk mendapatkan daftar aplikasi yang berjalan pada alat, gunakan ListApplicationInstancesAPI.

```
$ aws panorama list-application-instances
    "ApplicationInstances": [
        {
            "Name": "aws-panorama-sample",
            "ApplicationInstanceId": "applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq",
            "DefaultRuntimeContextDevice": "device-4tafxmplhtmzabv5lsacba4ere",
            "DefaultRuntimeContextDeviceName": "my-appliance",
            "Description": "command-line deploy",
            "Status": "DEPLOYMENT_SUCCEEDED",
            "HealthStatus": "RUNNING",
            "StatusDescription": "Application deployed successfully.",
            "CreatedTime": 1661902051.925,
            "Arn": "arn:aws:panorama:us-east-2:123456789012:applicationInstance/
applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq",
            "Tags": {
                "client": "sample"
            }
        },
    ]
}
```

Untuk mendapatkan rincian lebih lanjut tentang node instance aplikasi, gunakan ListApplicationInstanceNodeInstancesAPI.

```
$ aws panorama list-application-instance-node-instances --application-instance-id
 applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq
{
    "NodeInstances": [
        {
            "NodeInstanceId": "code_node",
            "NodeId": "SAMPLE_CODE-1.0-fd3dxmpl-interface",
            "PackageName": "SAMPLE_CODE",
            "PackageVersion": "1.0",
```

```
            "PackagePatchVersion":
 "fd3dxmpl2bdfa41e6fe1be290a79dd2c29cf014eadf7416d861ce7715ad5e8a8",
            "NodeName": "interface",
            "CurrentStatus": "RUNNING"
        },
        {
            "NodeInstanceId": "camera_node_override",
            "NodeId": "warehouse-floor-1.0-9eabxmpl-warehouse-floor",
            "PackageName": "warehouse-floor",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "9eabxmple89f0f8b2f2852cca2a6e7971aa38f1629a210d069045e83697e42a7",
            "NodeName": "warehouse-floor",
            "CurrentStatus": "RUNNING"
        },
        {
            "NodeInstanceId": "output_node",
            "NodeId": "hdmi_data_sink-1.0-9c23xmpl-hdmi0",
            "PackageName": "hdmi_data_sink",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "9c23xmplc4c98b92baea4af676c8b16063d17945a3f6bd8f83f4ff5aa0d0b394",
            "NodeName": "hdmi0",
            "CurrentStatus": "RUNNING"
        },
        {
            "NodeInstanceId": "model_node",
            "NodeId": "SQUEEZENET_PYTORCH-1.0-5d3cabda-interface",
            "PackageName": "SQUEEZENET_PYTORCH",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96",
            "NodeName": "interface",
            "CurrentStatus": "RUNNING"
        }
    ]
}
```

Kelola aliran kamera kamera

Anda dapat menjeda dan melanjutkan node aliran kamera dengan SignalApplicationInstanceNodeInstancesAPI.

```
$ aws panorama signal-application-instance-node-instances --application-instance-id
 applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq \
       --node-signals '[{"NodeInstanceId": "camera_node_override", "Signal":
 "PAUSE"}]'
{
    "ApplicationInstanceId": "applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq"
}
```

Dalam skrip, Anda bisa mendapatkan daftar node dan memilih salah satu untuk menjeda atau melanjutkan secara interaktif.

Example pause-camera.sh - penggunaan

```
my-app$ ./pause-camera.sh

Getting nodes...
0: SAMPLE_CODE              RUNNING
1: warehouse-floor          RUNNING
2: hdmi_data_sink           RUNNING
3: entrance-north           PAUSED
4: SQUEEZENET_PYTORCH       RUNNING
Choose a node
1
Signalling node warehouse-floor
+ aws panorama signal-application-instance-node-instances --application-instance-id
 applicationInstance-r3a7xmplcbmpjqeds7vj4b6pjy --node-signals '[{"NodeInstanceId":
 "warehouse-floor", "Signal": "PAUSE"}]'
{
    "ApplicationInstanceId": "applicationInstance-r3a7xmplcbmpjqeds7vj4b6pjy"
}
```

Dengan menjeda dan melanjutkan node kamera, Anda dapat melakukan siklus melalui aliran kamera dalam jumlah yang lebih besar daripada yang dapat diproses secara bersamaan. Untuk melakukan ini, petakan beberapa stream kamera ke node input yang sama di manifes override Anda.

Pada contoh berikut, manifes override memetakan dua aliran kamera,`warehouse-floor` dan`entrance-north` ke node input (`camera_node`) yang sama. `warehouse-floor`Aliran aktif saat aplikasi dimulai dan`entrance-north` node menunggu sinyal menyala.

Example mengesampingkan-multicam.json

```
    "nodeGraphOverrides": {
```

```
        "nodes": [
            {
                "name": "warehouse-floor",
                "interface": "123456789012::warehouse-floor.warehouse-floor",
                "launch": "onAppStart"
            },
            {
                "name": "entrance-north",
                "interface": "123456789012::entrance-north.entrance-north",
                "launch": "onSignal"
            },
        ...
        "packages": [
            {
                "name": "123456789012::warehouse-floor",
                "version": "1.0"
            },
            {
                "name": "123456789012::entrance-north",
                "version": "1.0"
            }
        ],
        "nodeOverrides": [
            {
                "replace": "camera_node",
                "with": [
                    {
                        "name": "warehouse-floor"
                    },
                    {
                        "name": "entrance-north"
                    }
                ]
            }
```

Untuk detail tentang penerapan dengan API, lihatMengotomatiskan penerapan aplikasi.

Menggunakan titik akhir VPC

Jika Anda bekerja di VPC tanpa akses internet, Anda dapat membuat titik akhir VPC untuk digunakan dengan AWS Panorama. Endpoint VPC memungkinkan klien yang berjalan di subnet pribadi terhubung ke layanan AWS tanpa koneksi internet.

Untuk detail tentang port dan titik akhir yang digunakan oleh AWS Panorama Appliance, lihat. ???

Bagian-bagian

- Membuat titik akhir VPC
- Menghubungkan alat ke subnet pribadi
- Contoh AWS CloudFormation template

Membuat titik akhir VPC

Untuk membuat koneksi pribadi antara VPC dan AWS Panorama, buat titik akhir VPC. Titik akhir VPC tidak diperlukan untuk menggunakan AWS Panorama. Anda hanya perlu membuat titik akhir VPC jika Anda bekerja di VPC tanpa akses internet. Saat AWS CLI atau SDK mencoba terhubung ke AWS Panorama, lalu lintas dirutekan melalui titik akhir VPC.

Buat titik akhir VPC untuk AWS Panorama menggunakan pengaturan berikut:

- Nama layanan – **`com.amazonaws.`*`us-west-2`*`.panorama`**
- Jenis - Antarmuka

Titik akhir VPC menggunakan nama DNS layanan untuk mendapatkan lalu lintas dari klien AWS SDK tanpa konfigurasi tambahan apa pun. Untuk informasi selengkapnya tentang penggunaan titik akhir VPC, lihat Titik akhir VPC Antarmuka di Panduan Pengguna VPC Amazon.

Menghubungkan alat ke subnet pribadi

AWS Panorama Appliance dapat terhubung AWS melalui koneksi VPN pribadi dengan AWS Site-to-Site VPN atau. AWS Direct Connect Dengan layanan ini, Anda dapat membuat subnet pribadi yang meluas ke pusat data Anda. Alat terhubung ke subnet pribadi dan mengakses AWS layanan melalui titik akhir VPC.

Site-to-Site VPN AWS Direct Connect dan merupakan layanan untuk menghubungkan pusat data Anda ke Amazon VPC dengan aman. Dengan Site-to-Site VPN, Anda dapat menggunakan perangkat

jaringan yang tersedia secara komersial untuk terhubung. AWS Direct Connectmenggunakan AWS perangkat untuk terhubung.

- Site-to-Site VPN — Apa itu? AWS Site-to-Site VPN
- AWS Direct Connect- Apa ituAWS Direct Connect?

Setelah Anda menghubungkan jaringan lokal Anda ke subnet pribadi di VPC, buat titik akhir VPC untuk layanan berikut.

- Layanan Penyimpanan Sederhana Amazon - AWS PrivateLinkuntuk Amazon S3
- AWS IoT Core— Menggunakan AWS IoT Core dengan titik akhir VPC antarmuka (bidang data dan penyedia kredensi)
- Amazon Elastic Container Registry - Titik akhir VPC antarmuka Amazon Elastic Container Registry
- Amazon CloudWatch - Menggunakan CloudWatch dengan titik akhir VPC antarmuka
- Amazon CloudWatch Logs - Menggunakan CloudWatch Log dengan titik akhir VPC antarmuka

Alat tidak memerlukan konektivitas ke layanan AWS Panorama. Ini berkomunikasi dengan AWS Panorama melalui saluran pesan di. AWS IoT

Selain titik akhir VPC, Amazon S3 dan AWS IoT memerlukan penggunaan zona host pribadi Amazon Route 53. Zona host pribadi merutekan lalu lintas dari subdomain, termasuk subdomain untuk jalur akses Amazon S3 dan topik MQTT, ke titik akhir VPC yang benar. Untuk informasi tentang zona yang dihosting pribadi, lihat Bekerja dengan zona yang dihosting pribadi di Panduan Pengembang Amazon Route 53.

Untuk contoh konfigurasi VPC dengan titik akhir VPC dan zona host pribadi, lihat. Contoh AWS CloudFormation template

Contoh AWS CloudFormation template

GitHub Repositori untuk panduan ini menyediakan AWS CloudFormation templat yang dapat Anda gunakan untuk membuat sumber daya untuk digunakan dengan AWS Panorama. Template membuat VPC dengan dua subnet pribadi, subnet publik, dan titik akhir VPC. Anda dapat menggunakan subnet pribadi di VPC untuk meng-host sumber daya yang terisolasi dari internet. Sumber daya di subnet publik dapat berkomunikasi dengan sumber daya pribadi, tetapi sumber daya pribadi tidak dapat diakses dari internet.

Example vpc-endpoint.yml—Subnet pribadi

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  vpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 172.31.0.0/16
      EnableDnsHostnames: true
      EnableDnsSupport: true
      Tags:
        - Key: Name
          Value: !Ref AWS::StackName
  privateSubnetA:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref vpc
      AvailabilityZone:
        Fn::Select:
         - 0
         - Fn::GetAZs: ""
      CidrBlock: 172.31.3.0/24
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub  ${AWS::StackName}-subnet-a
  ...
```

`vpc-endpoint.yml`Template menunjukkan cara membuat titik akhir VPC untuk AWS Panorama. Anda dapat menggunakan titik akhir ini untuk mengelola sumber daya AWS Panorama dengan SDK atauAWS. AWS CLI

Example vpc-endpoint.yml-titik akhir VPC

```
  panoramaEndpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub com.amazonaws.${AWS::Region}.panorama
      VpcId: !Ref vpc
      VpcEndpointType: Interface
      SecurityGroupIds:
      - !GetAtt vpc.DefaultSecurityGroup
      PrivateDnsEnabled: true
```

```
      SubnetIds:
      - !Ref privateSubnetA
      - !Ref privateSubnetB
      PolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal: "*"
          Action:
            - "panorama:*"
          Resource:
            - "*"
```

`PolicyDocument`Ini adalah kebijakan izin berbasis sumber daya yang mendefinisikan panggilan API yang dapat dilakukan dengan titik akhir. Anda dapat mengubah kebijakan untuk membatasi tindakan dan sumber daya yang dapat diakses melalui titik akhir. Untuk informasi selengkapnya, lihat Mengontrol Akses ke Layanan dengan titik akhir VPC dalam Panduan Pengguna Amazon VPC.

`vpc-appliance.yml`Template menunjukkan cara membuat titik akhir VPC dan zona host pribadi untuk layanan yang digunakan oleh AWS Panorama Appliance.

Example vpc-appliance.yl - Titik akhir titik akses Amazon S3 dengan zona host pribadi

```
  s3Endpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub com.amazonaws.${AWS::Region}.s3
      VpcId: !Ref vpc
      VpcEndpointType: Interface
      SecurityGroupIds:
      - !GetAtt vpc.DefaultSecurityGroup
      PrivateDnsEnabled: false
      SubnetIds:
      - !Ref privateSubnetA
      - !Ref privateSubnetB
...
  s3apHostedZone:
    Type: AWS::Route53::HostedZone
    Properties:
      Name: !Sub s3-accesspoint.${AWS::Region}.amazonaws.com
      VPCs:
        - VPCId: !Ref vpc
          VPCRegion: !Ref AWS::Region
```

```
  s3apRecords:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref s3apHostedZone
      Name: !Sub "*.s3-accesspoint.${AWS::Region}.amazonaws.com"
      Type: CNAME
      TTL: 600
      # first DNS entry, split on :, second value
      ResourceRecords:
      - !Select [1, !Split [":", !Select [0, !GetAtt s3Endpoint.DnsEntries ] ] ]
```

Templat sampel menunjukkan pembuatan sumber daya Amazon VPC dan Route 53 dengan sampel VPC. Anda dapat menyesuaikan ini untuk kasus penggunaan Anda dengan menghapus sumber daya VPC dan mengganti referensi ke subnet, grup keamanan, dan ID VPC dengan ID sumber daya Anda.

Contoh aplikasi, skrip, dan templat

The GitHub repositori untuk panduan ini menyediakan contoh aplikasi, skrip, dan templat untukAWS Panoramaperangkat. Gunakan sampel ini untuk mempelajari praktik terbaik dan mengotomatiskan alur kerja pengembangan.

Bagian

- Aplikasi sampel
- Skrip utilitas
- AWS CloudFormation templat
- Lebih banyak sampel dan alat

Aplikasi sampel

Contoh aplikasi menunjukkan penggunaanAWS Panoramafitur dan tugas visi komputer umum. Contoh aplikasi ini mencakup skrip dan templat yang mengotomatiskan pengaturan dan penerapan. Dengan konfigurasi minimal, Anda dapat menyebarkan dan memperbarui aplikasi dari baris perintah.

- aws-panorama-sample— Visi komputer dasar dengan model klasifikasi. GunakanAWS SDK for Python (Boto)untuk mengunggah metrik ke CloudWatch, metode preprocessing dan inferensi instrumen, dan konfigurasikan logging.
- debug-server—Buka port masukpada perangkat dan meneruskan lalu lintas ke wadah kode aplikasi. Gunakan multithreading untuk menjalankan kode aplikasi, server HTTP, dan klien HTTP secara bersamaan.
- model khusus— Ekspor model dari kode dan kompilasi dengan SageMaker Neo untuk menguji kompatibilitas denganAWS PanoramaAlat. Bangun secara lokal dalam pengembangan Python, dalam wadah Docker, atau di instans Amazon EC2. Ekspor dan kompilasi semua model aplikasi bawaan di Keras untuk yang spesifik TensorFlow atau versi Python.

Untuk aplikasi sampel lainnya, kunjungi juga aws-panorama-samplesrepositori.

Skrip utilitas

Skrip di`util-scripts`mengelola direktoriAWS Panoramasumber daya atau mengotomatiskan alur kerja pengembangan.

- provision-device.sh— Menyediakan perangkat.
- check-updates.sh— Periksa dan terapkan pembaruan perangkat lunak alat.
- reboot-device.sh— Reboot perangkat.
- register-camera.sh— Daftarkan kamera.
- deregister-camera.sh— Hapus simpul kamera.
- view-logs.sh— Lihat log untuk contoh aplikasi.
- pause-camera.sh— Jeda atau lanjutkan aliran kamera.
- push.sh— Membangun, mengunggah, dan menyebarkan aplikasi.
- rename-package.sh— Ganti nama paket node. Memperbarui nama direktori, file konfigurasi, dan manifes aplikasi.
- samplify.sh— Ganti ID akun Anda dengan contoh ID akun, dan pulihkan konfigurasi cadangan untuk menghapus konfigurasi lokal.
- update-model-config.sh— Tambahkan kembali model ke aplikasi setelah memperbarui file deskriptor.
- cleanup-patches.sh— Deregister versi patch lama dan hapus manifes mereka dari Amazon S3.

Untuk detail penggunaan, lihatREADME.

AWS CloudFormation templat

GunakanAWS CloudFormationtemplate di`cloudformation-templates`direktori untuk membuat sumber daya untukAWS Panoramaaplikasi.

- alarm-aplikasi.yl— Buat alarm yang memantau aplikasi untuk kesalahan. Jika instance aplikasi menimbulkan kesalahan atau berhenti berjalan selama 5 menit, alarm akan mengirimkan email notifikasi.
- alarm-device.yml— Buat alarm yang memonitor konektivitas perangkat. Jika perangkat berhenti mengirim metrik selama 5 menit, alarm akan mengirimkan email pemberitahuan.

- aplikasi-role.yml.— Buat peran aplikasi. Peran tersebut mencakup izin untuk mengirim metrik CloudWatch. Tambahkan izin ke pernyataan kebijakan untuk operasi API lain yang digunakan aplikasi Anda.
- vpc-appliance.yml— Buat VPC dengan akses layanan subnet pribadi untukAWS PanoramaAlat. Untuk menghubungkan alat ke VPC, gunakanAWS Direct ConnectatauAWS Site-to-Site VPN.
- vpc-endpoint.yml— Buat VPC dengan akses layanan subnet pribadi keAWS Panoramalayanan. Sumber daya di dalam VPC dapat terhubung keAWS Panoramauntuk memantau dan mengelolaAWS Panoramasumber daya tanpa terhubung ke internet.

The`create-stack.sh`script dalam direktori ini menciptakanAWS CloudFormationtumpukan. Dibutuhkan sejumlah variabel argumen. Argumen pertama adalah nama template, dan argumen yang tersisa adalah penggantian untuk parameter dalam template.

Misalnya, perintah berikut membuat peran aplikasi.

```
$ ./create-stack.sh application-role
```

Lebih banyak sampel dan alat

Theaws-panorama-samplesrepositori memiliki lebih banyak aplikasi sampel dan alat yang berguna.

- Aplikasi— Contoh aplikasi untuk berbagai arsitektur model dan kasus penggunaan.
- Validasi aliran kamera— Validasi aliran kamera.
- PanoJupyter— Jalankan JupyterLab pada sebuahAWS PanoramaAlat.
- Sideloading— Perbarui kode aplikasi tanpa membangun atau menyebarkan wadah aplikasi.

TheAWSkomunitas juga telah mengembangkan alat dan panduan untukAWS Panorama. Lihat proyek open source berikut di GitHub.

- cookiecutter-panorama- Template Cookiecutter untukAWS Panoramaaplikasi.
- ransel— Modul Python untuk mengakses detail lingkungan runtime, pembuatan profil, dan opsi output video tambahan.

PemantauanAWS Panoramasumber daya dan aplikasi

Anda dapat memonitorAWS Panoramasumber daya diAWS Panoramakonsol dan dengan AmazonCloudWatch. ParameterAWS PanoramaAppliance terhubung keAWSCloud melalui internet untuk melaporkan status dan status kamera yang terhubung. Saat aktif, alat juga mengirimkan log keCloudWatchLog secara waktu nyata.

Alat mendapat izin untuk menggunakanAWS IoT,CloudWatchLog, dan layanan AWS lainnya dari peran layanan yang Anda buat pertama kali Anda menggunakanAWS Panoramakonsol. Untuk informasi selengkapnya, lihat Peran layanan AWS Panorama dan sumber daya lintas layanan.

Untuk bantuan mengatasi masalah kesalahan tertentu, lihatPemecahan Masalah.

Topik

- Pemantauan di konsol AWS Panorama
- Melihat log AWS Panorama
- Memantau peralatan dan aplikasi dengan AmazonCloudWatch

Pemantauan di konsol AWS Panorama

Anda dapat menggunakan Konsol AWS Panorama untuk memantau AWS Panorama Appliance dan kamera. Konsol menggunakanAWS IoTuntuk memantau keadaan alat.

Untuk memantau alat Anda di konsol AWS Panorama

1. BukaKonsol AWS Panorama.
2. Buka konsol AWS PanoramaHalaman Perangkat.
3. Pilih alat.
4. Untuk melihat status instance aplikasi, pilih dari daftar.
5. Untuk melihat status antarmuka jaringan alat, pilihPengaturan.

Status keseluruhan alat muncul di bagian atas halaman. Jika statusnyaOnline, maka alat terhubung keAWSdan mengirim update status reguler.

Melihat log AWS Panorama

AWS Panorama melaporkan aplikasi dan peristiwa sistem ke Amazon CloudWatch Log. Saat mengalami masalah, Anda dapat menggunakan log peristiwa untuk membantu men-debug aplikasi AWS Panorama atau memecahkan masalah konfigurasi aplikasi.

Untuk melihat log CloudWatch Beberapa catatan

1. BukaLog halaman grup dari CloudWatch Konsol log.
2. Temukan log aplikasi dan alat AWS Panorama dalam grup berikut:

   - Log perangkat—`/aws/panorama/devices/device-id`
   - Log aplikasi—`/aws/panorama/devices/device-id/applications/instance-id`

Saat Anda menyediakan kembali alat setelah memperbarui perangkat lunak sistem, Anda juga dapatmelihat log pada drive USB penyediaan.

Bagian

- Melihat log perangkat
- Melihat log aplikasi
- Mengonfigurasi log aplikasi
- Melihat log penyediaan
- Egressing log dari perangkat

Melihat log perangkat

AWS Panorama Appliance membuat grup log untuk perangkat, dan grup untuk setiap instans aplikasi yang Anda terapkan. Log perangkat berisi informasi tentang status aplikasi, peningkatan perangkat lunak, dan konfigurasi sistem.

Log perangkat -**`/aws/panorama/devices/device-id`**

- `occ_log`- Output dari proses pengontrol. Proses ini mengoordinasikan penerapan aplikasi dan laporan tentang status node setiap instance aplikasi.
- `ota_log`- Output dari proses yang berkoordinasi over-the-air (OTA) peningkatan perangkat lunak.

- `syslog`- Output dari proses syslog perangkat, yang menangkap pesan yang dikirim antar proses.
- `kern_log`- Acara dari kernel Linux perangkat.
- `logging_setup_logs`- Output dari proses yang mengkonfigurasi CloudWatch Agen log.
- `cloudwatch_agent_logs`- Output dari CloudWatch Agen log.
- `shadow_log`- Output dariAWS IoTbayangan perangkat.

Melihat log aplikasi

Grup log instance aplikasi berisi aliran log untuk setiap simpul, dinamai simpul.

Log aplikasi —**`/aws/panorama/devices/`*`device-id`*`/applications/`*`instance-id`***

- Kode— Keluaran dari kode aplikasi Anda dan AWS Panorama Application SDK. Agregat log aplikasi dari`/opt/aws/panorama/logs`.
- Model- Output dari proses yang mengoordinasikan permintaan inferensi dengan model.
- Aliran- Output dari proses yang memecahkan kode video dari aliran kamera.
- Tampilan- Output dari proses yang membuat output video untuk port HDMI.
- `mds`- Log dari server metadata alat.
- `console_output`- Menangkap output standar dan aliran kesalahan dari wadah kode.

Jika Anda tidak melihat log in CloudWatch Log, konfirmasikan bahwa Anda berada di AWS Region yang benar. Jika ya, mungkin ada masalah dengan koneksi alat ke AWS atau dengan izin aktifalatAWS Identity and Access Management(IAM) peran.

Mengonfigurasi log aplikasi

Mengonfigurasi pencatat Python untuk menulis file log`/opt/aws/panorama/logs`. Alat mengalirkan log dari lokasi ini ke CloudWatch Log. Untuk menghindari penggunaan terlalu banyak ruang disk, gunakan ukuran file maksimum 10 MiB dan jumlah cadangan 1. Contoh berikut ini menunjukkan metode yang menciptakan pencatat.

Example application.py— Konfigurasi Pencatat

```
def get_logger(name=__name__,level=logging.INFO):
```

```
    logger = logging.getLogger(name)
    logger.setLevel(level)
    LOG_PATH = '/opt/aws/panorama/logs'
    handler = RotatingFileHandler("{}/app.log".format(LOG_PATH), maxBytes=10000000,
 backupCount=1)
    formatter = logging.Formatter(fmt='%(asctime)s %(levelname)-8s %(message)s',
                                  datefmt='%Y-%m-%d %H:%M:%S')
    handler.setFormatter(formatter)
    logger.addHandler(handler)
    return logger
```

Inisialisasi logger di lingkup global dan gunakan di seluruh kode aplikasi Anda.

Example application.py— Inisialisasi pencatat

```
def main():
    try:
        logger.info("INITIALIZING APPLICATION")
        app = Application()
        logger.info("PROCESSING STREAMS")
        while True:
            app.process_streams()
            # turn off debug logging after 150 loops
            if logger.getEffectiveLevel() == logging.DEBUG and app.frame_num == 150:
                logger.setLevel(logging.INFO)
    except:
        logger.exception('Exception during processing loop.')

logger = get_logger(level=logging.INFO)
main()
```

Melihat log penyediaan

Selama penyediaan, AWS Panorama Appliance menyalin log ke drive USB yang Anda gunakan untuk mentransfer arsip konfigurasi ke alat. Gunakan log ini untuk memecahkan masalah penyediaan pada peralatan dengan versi perangkat lunak terbaru.

> **Important**
>
> Log penyediaan tersedia untuk peralatan yang diperbarui ke perangkat lunak versi 4.3.23 atau yang lebih baru.

Log aplikasi

- `/panorama/occ.log`- Log perangkat lunak pengontrol AWS Panorama.
- `/panorama/ota_agent.log`— AWS Panorama over-the-air agen pembaruan log.
- `/panorama/syslog.log`- Log sistem Linux.
- `/panorama/kern.log`- Log kernel Linux.

Egressing log dari perangkat

Jika log perangkat dan aplikasi Anda tidak muncul di CloudWatch Log, Anda dapat menggunakan drive USB untuk mendapatkan gambar log terenkripsi dari perangkat. Tim layanan AWS Panorama dapat mendekripsi log atas nama Anda dan membantu dalam debugging.

Prasyarat

Untuk mengikuti prosedur Anda membutuhkan perangkat keras berikut:

- Drive USB- Drive memori flash USB berformat FAT32 dengan penyimpanan minimal 1 GB, untuk mentransfer file log dari AWS Panorama Appliance.

Untuk keluar log dari perangkat

1. Siapkan drive USB dengan`managed_logs`folder di dalam sebuah`panorama`folder

   ```
   /
    ### panorama
        ### managed_logs
   ```

2. Connect drive USB ke perangkat.
3. Power offAWS Panorama Appliance.
4. Nyalakan AWS Panorama Appliance.
5. Perangkat menyalin log ke perangkat. Status LEDberkedip birusementara ini sedang berlangsung.
6. File log kemudian dapat ditemukan di dalam`managed_logs`direktori dengan format`panorama_device_log_`*`v1_dd_hh_mm`*`.img`

Anda tidak dapat mendekripsi gambar log sendiri. Bekerja dengan dukungan pelanggan, manajer akun teknis untuk AWS Panorama, atau arsitek solusi untuk berkoordinasi dengan tim layanan.

Memantau peralatan dan aplikasi dengan AmazonCloudWatch

Saat alat online, AWS Panorama mengirimkan metrik ke AmazonCloudWatch. Anda dapat membuat grafik dan dasbor dengan metrik ini diCloudWatchkonsol untuk memantau aktivitas alat, dan mengatur alarm yang memberi tahu Anda saat perangkat offline atau aplikasi mengalami kesalahan.

Untuk melihat metrik dalam konsol CloudWatch

1. BukaHalaman Metrik konsol AWS Panorama(`PanoramaDeviceMetrics`namespace).
2. Memiilih skema dimensi.
3. Pilih metrik untuk menambahkannya ke grafik.
4. Untuk memilih statistik yang berbeda dan menyesuaikan grafik, gunakan opsi pada tab Metrik bergrafik. Secara default, grafik menggunakan statistik `Average` untuk semua metrik.

> **Harga**
>
> CloudWatchmemiliki tingkat Selalu Gratis. Di luar ambang tingkat gratis, CloudWatch membebankan biaya untuk metrik, RDS, alarm, log, dan wawasan. Untuk detailnya, lihat CloudWatch harga.

Untuk informasi lebih lanjut tentang CloudWatch, lihat Panduan Pengguna Amazon CloudWatch.

Bagian

- Menggunakan metrik perangkat
- Menggunakan metrik aplikasi
- Mengonfigurasi alarm

Menggunakan metrik perangkat

Saat alat online, ia akan mengirimkan metrik ke AmazonCloudWatch. Anda dapat menggunakan metrik ini untuk memantau aktivitas perangkat dan memicu alarm jika perangkat offline.

- `DeviceActive`— Dikirim secara berkala ketika perangkat aktif.

  Dimensi -`DeviceId`dan`DeviceName`.

MelihatDeviceActivemetrik denganAveragestatistik.

Menggunakan metrik aplikasi

Saat aplikasi mengalami kesalahan, aplikasi akan mengirimkan metrik ke AmazonCloudWatch. Anda dapat menggunakan metrik ini untuk memicu alarm jika aplikasi berhenti berjalan.

- `ApplicationErrors`— Jumlah kesalahan aplikasi yang direkam.

  Dimensi -`ApplicationInstanceName`dan`ApplicationInstanceId`.

Melihat metrik aplikasi denganSumstatistik.

Mengonfigurasi alarm

Untuk mendapatkan notifikasi saat metrik melebihi ambang batas, buat alarm. Misalnya, Anda dapat membuat alarm yang mengirimkan notifikasi saat jumlah`ApplicationErrors`metrik tetap di 1 selama 20 menit.

Untuk membuat alarm

1. BukaAmazonCloudWatchlaman Alarm.
2. Pilih Create alarm (Buat alarm).
3. PilihPilih metrikdan menemukan metrik untuk perangkat Anda, seperti`ApplicationErrors`untuk`applicationInstance-gk75xmplqbqtenlnmz4ehiu7xa`,`my-application`.
4. Ikuti petunjuk untuk mengkonfigurasi kondisi, tindakan, dan nama untuk alarm.

Untuk detail langkah-langkahnya, lihatBuatCloudWatchalarmdiAmazonCloudWatchPanduan Pengguna.

Pemecahan Masalah

Topik berikut memberikan saran pemecahan masalah untuk kesalahan dan masalah yang mungkin Anda temui saat menggunakan AWS Panorama konsol, alat, atau SDK. Jika Anda menemukan masalah yang tidak tercantum di sini, gunakan tombol Berikan umpan balik di halaman ini untuk melaporkannya.

Anda dapat menemukan log untuk alat Anda di konsol Amazon CloudWatch Logs. Alat mengunggah log dari kode aplikasi Anda, perangkat lunak alat, dan AWS IoT proses saat dibuat. Untuk informasi selengkapnya, lihat Melihat log AWS Panorama.

Penyediaan

Masalah: (macOS) Komputer saya tidak mengenali drive USB yang disertakan dengan adaptor USB-C.

Ini dapat terjadi jika Anda mencolokkan drive USB ke adaptor USB-C yang sudah terhubung ke komputer Anda. Coba lepaskan adaptor dan sambungkan kembali dengan drive USB yang sudah terpasang.

Masalah: Penyediaan gagal saat saya menggunakan drive USB saya sendiri.

Masalah: Penyediaan gagal saat saya menggunakan port USB 2.0 alat.

AWS PanoramaAlat ini kompatibel dengan perangkat memori flash USB antara 1 dan 32 GB, tetapi tidak semuanya kompatibel. Beberapa masalah telah diamati saat menggunakan port USB 2.0 untuk penyediaan. Untuk hasil yang konsisten, gunakan drive USB yang disertakan dengan port USB 3.0 (di sebelah port HDMI).

Untuk Lenovo ThinkEdge® SE70, drive USB tidak disertakan dengan alat. Gunakan drive USB 3.0 dengan penyimpanan minimal 1 GB.

Konfigurasi alat

Masalah: Alat menunjukkan layar kosong saat boot up.

Setelah menyelesaikan urutan boot awal, yang memakan waktu sekitar satu menit, alat menampilkan layar kosong selama satu menit atau lebih saat memuat model Anda dan memulai aplikasi Anda. Selain itu, alat tidak mengeluarkan video jika Anda menghubungkan layar setelah dihidupkan.

Masalah: Alat tidak merespons saat saya menahan tombol daya untuk mematikannya.

Alat membutuhkan waktu hingga 10 detik untuk mati dengan aman. Anda perlu menahan tombol daya hanya selama 1 detik untuk memulai urutan shutdown. Untuk daftar lengkap operasi tombol, lihatTombol dan lampu AWS Panorama Appliance.

Masalah: Saya perlu membuat arsip konfigurasi baru untuk mengubah pengaturan atau mengganti sertifikat yang hilang.

AWS Panoramatidak menyimpan sertifikat perangkat atau konfigurasi jaringan setelah Anda mengunduhnya, dan Anda tidak dapat menggunakan kembali arsip konfigurasi. Hapus alat menggunakan AWS Panorama konsol dan buat yang baru dengan arsip konfigurasi baru.

Konfigurasi aplikasi

Masalah: Ketika saya menjalankan beberapa aplikasi, saya tidak dapat mengontrol mana yang menggunakan output HDMI.

Saat Anda menerapkan beberapa aplikasi yang memiliki node keluaran, aplikasi yang dimulai paling baru menggunakan output HDMI. Jika aplikasi ini berhenti berjalan, aplikasi lain dapat menggunakan output. Untuk memberikan hanya satu akses aplikasi ke output, hapus node output dan tepi yang sesuai dari manifes aplikasi aplikasi lain dan redeploy.

Masalah: Output aplikasi tidak muncul di log

Konfigurasikan logger Python untuk menulis file log ke. `/opt/aws/panorama/logs` Ini ditangkap dalam aliran log untuk node kontainer kode. Output standar dan aliran kesalahan ditangkap dalam aliran log terpisah yang disebut`console-output`. Jika Anda menggunakan`print`, gunakan `flush=True` opsi untuk menjaga pesan agar tidak macet di buffer keluaran.

Kesalahan: You've reached the maximum number of versions for package SAMPLE_CODE. Deregister unused package versions and try again.

Sumber: AWS Panorama layanan

Setiap kali Anda menerapkan perubahan ke aplikasi, Anda mendaftarkan versi patch yang mewakili konfigurasi paket dan file aset untuk setiap paket yang digunakannya. Gunakan skrip patch pembersihan untuk membatalkan pendaftaran versi tambalan yang tidak digunakan.

Aliran kamera

Kesalahan: liveMedia0: Failed to get SDP description: Connection to server failed: Connection timed out (-115)

Kesalahan: liveMedia0: Failed to get SDP description: 404 Not Found; with the result code: 404

Kesalahan: liveMedia0: Failed to get SDP description: DESCRIBE send() failed: Broken pipe; with the result code: -32

Sumber: Log simpul kamera

Alat tidak dapat terhubung ke aliran kamera aplikasi. Ketika ini terjadi, output video kosong atau macet pada frame yang terakhir diproses sementara aplikasi menunggu bingkai video dari SDK AWS Panorama Aplikasi. Perangkat lunak alat mencoba terhubung ke aliran kamera dan mencatat kesalahan batas waktu di log simpul kamera. Verifikasi bahwa URL aliran kamera Anda benar dan lalu lintas RTSP dapat dirutekan antara kamera dan alat dalam jaringan Anda. Untuk informasi selengkapnya, lihat Menghubungkan AWS Panorama Appliance ke jaringan Anda.

Kesalahan: ERROR finalizeInterface(35) Camera credential fetching for port [username] failed

Sumber: log OCC

AWS Secrets ManagerRahasia dengan kredensi aliran kamera tidak dapat ditemukan. Hapus aliran kamera dan buat ulang.

Kesalahan: Camera did not provide an H264 encoded stream

Sumber: Log simpul kamera

Aliran kamera memiliki pengkodean selain H.264, seperti H.265. Menerapkan ulang aplikasi dengan aliran kamera H.264. Untuk detail tentang kamera yang didukung, lihatKamera yang didukung.

Keamanan di AWS Panorama

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda akan mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan Anda. Model tanggung jawab bersama menggambarkan ini sebagai keamanan dari cloud dan keamanan di dalam cloud:

- Keamanan cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan layanan AWS di dalam AWS Cloud. AWS juga memberi layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari program kepatuhan AWS. Untuk mempelajari program kepatuhan yang berlaku di AWS Panorama, lihatAWSLayanan dalam Lingkup oleh Program Kepatuhan.
- Keamanan di cloud – Tanggung jawab Anda ditentukan menurut layanan AWS yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta hukum dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS Panorama. Topik berikut akan menunjukkan kepada Anda cara mengkonfigurasi AWS Panorama untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan layanan AWS lain yang membantu Anda memantau dan mengamankan sumber daya AWS Panorama.

Topik

- Fitur-fitur keamanan AWS Panorama
- Praktik terbaik keamanan AWS Panorama
- Perlindungan data di AWS Panorama
- Manajemen identitas dan akses untuk AWS Panorama
- Validasi kepatuhan untuk AWS Panorama
- Keamanan infrastruktur di AWS Panorama
- Perangkat lunak lingkungan runtime di AWS Panorama

Fitur-fitur keamanan AWS Panorama

Untuk melindungi Andaaplikasi, model, dan perangkat keras terhadap kode berbahaya dan eksploitasi lainnya, AWS Panorama Appliance mengimplementasikan serangkaian fitur keamanan yang luas. Ini termasuk namun tidak terbatas pada berikut.

- Enkripsi disk penuh— Alat mengimplementasikan Linux unified key setup (LUKS2) enkripsi full-disk. Semua perangkat lunak sistem dan data aplikasi dienkripsi dengan kunci yang spesifik untuk perangkat Anda. Bahkan dengan akses fisik ke perangkat, penyerang tidak dapat memeriksa isi penyimpanannya.
- Tata letak memori— Untuk melindungi terhadap serangan yang menargetkan kode eksekusi yang dimuat ke dalam memori, AWS Panorama Appliance menggunakan pengacakan tata letak ruang alamat (ASLR). ASLR mengacak lokasi kode sistem operasi seperti yang dimuat ke memori. Hal ini mencegah penggunaan eksploitasi yang mencoba untuk menimpa atau menjalankan bagian kode tertentu dengan memprediksi di mana ia disimpan pada saat runtime.
- Lingkungan eksekusi tepercaya— Alat ini menggunakan lingkungan eksekusi tepercaya (TEE) berdasarkan ARM TrustZone, dengan penyimpanan terisolasi, memori, dan sumber daya pemrosesan. Kunci dan data sensitif lainnya yang tersimpan di zona kepercayaan hanya dapat diakses oleh aplikasi tepercaya, yang berjalan dalam sistem operasi terpisah dalam TEE. Perangkat lunak AWS Panorama Appliance berjalan di lingkungan Linux yang tidak tepercaya bersama kode aplikasi. Ini hanya dapat mengakses operasi kriptografi dengan membuat permintaan untuk aplikasi yang aman.
- Penyediaan aman— Ketika Anda menyediakan alat, kredensi (kunci, sertifikat, dan materi kriptografi lainnya) yang Anda transfer ke perangkat hanya berlaku untuk waktu yang singkat. Alat ini menggunakan kredensi berumur pendek untuk terhubung keAWS IoTdan meminta sertifikat untuk dirinya sendiri yang berlaku untuk waktu yang lebih lama. Layanan AWS Panorama menghasilkan kredensyal dan mengenkripsinya dengan kunci yang dikodekan pada perangkat. Hanya perangkat yang meminta sertifikat yang dapat mendekripsi dan berkomunikasi dengan AWS Panorama.
- Boot aman— Saat perangkat dinyalakan, setiap komponen perangkat lunak diautentikasi sebelum dijalankan. Boot ROM, perangkat lunak hardcoded dalam prosesor yang tidak dapat dimodifikasi, menggunakan kunci enkripsi hardcoded untuk mendekripsi bootloader, yang memvalidasi kernel lingkungan eksekusi tepercaya, dan sebagainya.

- Kernel yang ditandatangani— Modul kernel ditandatangani dengan kunci enkripsi asimetris. Kernel sistem operasi mendekripsi tanda tangan dengan kunci publik dan memverifikasi bahwa itu cocok dengan tanda tangan modul sebelum memuat modul ke memori.
- dm-kejujuran- Mirip dengan bagaimana modul kernel divalidasi, alat menggunakan Linux Device Mapperdm-verityfitur untuk memverifikasi integritas gambar perangkat lunak alat sebelum memasangnya. Jika perangkat lunak alat dimodifikasi, itu tidak akan berjalan.
- Pencegahan rollback— Ketika Anda memperbarui perangkat lunak alat, alat meniup sekering elektronik pada SoC (sistem pada chip). Setiap versi perangkat lunak mengharapkan peningkatan jumlah sekering yang akan ditiup, dan tidak dapat berjalan jika lebih banyak ditiup.

Praktik terbaik keamanan AWS Panorama

Ingatlah praktik terbaik berikut saat menggunakan alat AWS Panorama.

- Secara fisik mengamankan alat— Pasang alat di rak server tertutup atau ruang aman. Batasi akses fisik ke perangkat ke personil yang berwenang.
- Mengamankan koneksi jaringan alat— Connect alat ke router yang membatasi akses ke sumber daya internal dan eksternal. Alat perlu terhubung ke kamera, yang bisa berada di jaringan internal yang aman. Hal ini juga perlu terhubung keAWS. Gunakan port Ethernet kedua hanya untuk redundansi fisik, dan konfigurasikan router untuk mengizinkan hanya lalu lintas yang diperlukan.

  Gunakan salah satu konfigurasi jaringan yang disarankan untuk merencanakan tata letak jaringan Anda. Untuk informasi selengkapnya, lihat Menghubungkan AWS Panorama Appliance ke jaringan Anda.
- Format drive USB— Setelah menyediakan alat, menghapus drive USB dan format itu. Alat tidak menggunakan drive USB setelah mendaftar dengan layanan AWS Panorama. Format drive untuk menghapus kredensi sementara, file konfigurasi, dan log penyediaan.
- Tetap terbaru alat— Terapkan pembaruan perangkat lunak alat pada waktu yang tepat. Saat Anda melihat alat di konsol AWS Panorama, konsol akan memberi tahu Anda jika pembaruan perangkat lunak tersedia. Untuk informasi selengkapnya, lihat Mengelola AWS Panorama Appliance.

  DenganDescribeDeviceOperasi API, Anda dapat mengotomatisasi memeriksa pembaruan dengan membandingkan`LatestSoftware`dan`CurrentSoftware`bidang. Bila versi perangkat lunak terbaru berbeda dari versi saat ini, terapkan pembaruan dengan konsol atau dengan menggunakanCreateJobforDevicesoperasi.
- Jika Anda berhenti menggunakan alat, atur ulang— Sebelum Anda memindahkan alat keluar dari pusat data aman Anda, atur ulang sepenuhnya. Dengan alat dimatikan dan terpasang, tekan tombol power dan reset secara bersamaan selama 5 detik. Ini menghapus kredensi akun, aplikasi, dan log dari alat.

  Untuk informasi selengkapnya, lihat Tombol dan lampu AWS Panorama Appliance.
- Batasi akses ke AWS Panorama dan layanan AWS lainnya— TheAWSPanoramaFullAccessmenyediakan akses ke semua operasi AWS Panorama API dan, jika perlu, akses ke layanan lain. Jika memungkinkan, kebijakan membatasi akses ke sumber daya berdasarkan konvensi penamaan. Sebagai contoh, ia menyediakan akses keAWS Secrets Managerrahasia yang memiliki nama dimulai dengan`panorama`. Untuk pengguna yang

memerlukan akses hanya-baca, atau akses ke kumpulan sumber daya yang lebih spesifik, gunakan kebijakan terkelola sebagai titik awal untuk kebijakan yang paling tidak istimewa Anda.

Untuk informasi selengkapnya, lihat Kebijakan IAM berbasis identitas untuk AWS AWS AWS AWS Panorama.

Perlindungan data di AWS Panorama

Model tanggung jawab AWS bersama model tanggung berlaku untuk perlindungan data di AWS Panorama. Sebagaimana diuraikan dalam model ini, AWS bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk memelihara kendali atas isi yang dihost pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS.

Untuk tujuan perlindungan data, sebaiknya lindungi kredensial Akun AWS dan siapkan untuk masing-masing pengguna AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya AWS. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pengelogan aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama semua kontrol keamanan bawaan dalam Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi yang lebih lengkap tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-2.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan AWS Panorama atau lainnya Layanan AWS menggunakan konsol, APIAWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Bagian-bagian

- Enkripsi dalam bergerak
- Alat AWS Panorama
- Aplikasi
- Layanan lainnya

Enkripsi dalam bergerak

Titik akhir AWS Panorama API mendukung koneksi aman hanya melalui HTTPS. Saat Anda mengelola sumber daya AWS Panorama dengan AWS SDK, atau AWS Management Console AWS Panorama API, semua komunikasi dienkripsi dengan Transport Layer Security (TLS). Komunikasi antara AWS Panorama Appliance dan AWS juga dienkripsi dengan TLS. Komunikasi antara AWS Panorama Appliance dan kamera melalui RTSP tidak dienkripsi.

Untuk daftar lengkap titik akhir API, lihat Wilayah AWS dan titik akhir di. Referensi Umum AWS

Alat AWS Panorama

AWS Panorama Appliance memiliki port fisik untuk Ethernet, video HDMI, dan penyimpanan USB. Slot kartu SD, Wi-Fi, dan Bluetooth tidak dapat digunakan. Port USB hanya digunakan selama penyediaan untuk mentransfer arsip konfigurasi ke alat.

Isi arsip konfigurasi, yang mencakup sertifikat penyediaan alat dan konfigurasi jaringan, tidak dienkripsi. AWS Panorama tidak menyimpan file-file ini; file tersebut hanya dapat diambil saat Anda mendaftarkan alat. Setelah Anda mentransfer arsip konfigurasi ke alat, hapus dari komputer dan perangkat penyimpanan USB Anda.

Seluruh sistem file alat dienkripsi. Selain itu, alat ini menerapkan beberapa perlindungan tingkat sistem, termasuk perlindungan rollback untuk pembaruan perangkat lunak yang diperlukan, kernel dan bootloader yang ditandatangani, dan verifikasi integritas perangkat lunak.

Saat Anda berhenti menggunakan alat, lakukan reset penuh untuk menghapus data aplikasi Anda dan mengatur ulang perangkat lunak alat.

Aplikasi

Anda mengontrol kode yang Anda terapkan ke alat Anda. Validasi semua kode aplikasi untuk masalah keamanan sebelum menerapkannya, terlepas dari sumbernya. Jika Anda menggunakan

pustaka pihak ketiga dalam aplikasi Anda, pertimbangkan dengan cermat kebijakan lisensi dan dukungan untuk pustaka tersebut.

Aplikasi CPU, memori, dan penggunaan disk tidak dibatasi oleh perangkat lunak alat. Aplikasi yang menggunakan terlalu banyak sumber daya dapat berdampak negatif pada aplikasi lain dan pengoperasian perangkat. Uji aplikasi secara terpisah sebelum menggabungkan atau menyebarkan ke lingkungan produksi.

Aset aplikasi (kode dan model) tidak diisolasi dari akses dalam akun, perangkat, atau lingkungan build Anda. Gambar kontainer dan arsip model yang dihasilkan oleh AWS Panorama Application CLI tidak dienkripsi. Gunakan akun terpisah untuk beban kerja produksi dan hanya izinkan akses sesuai kebutuhan.

Layanan lainnya

Untuk menyimpan model dan wadah aplikasi Anda dengan aman di Amazon S3, AWS Panorama menggunakan enkripsi sisi server dengan kunci yang dikelola Amazon S3. Untuk informasi selengkapnya, lihat Melindungi data menggunakan enkripsi di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Kredensyal aliran kamera dienkripsi saat istirahat. AWS Secrets Manager Peran IAM alat memberinya izin untuk mengambil rahasia untuk mengakses nama pengguna dan kata sandi aliran.

AWS Panorama Appliance mengirimkan data log ke Amazon CloudWatch Logs. CloudWatch Log mengenkripsi data ini secara default, dan dapat dikonfigurasi untuk menggunakan kunci yang dikelola pelanggan. Untuk informasi selengkapnya, lihat Mengenkripsi data CloudWatch log di Log menggunakan AWS KMS Panduan Pengguna Amazon CloudWatch Logs.

Manajemen identitas dan akses untuk AWS Panorama

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke sumber daya AWS secara aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya AWS Panorama. IAM adalah layanan Layanan AWS yang dapat Anda gunakan tanpa dikenakan biaya tambahan.

Topik

- Audiens
- Mengautentikasi dengan identitas
- Mengelola akses menggunakan kebijakan
- Cara AWS Panorama bekerja dengan IAM
- Contoh kebijakan berbasis identitas AWS Panorama
- AWSkebijakan terkelola untuk AWS Panorama
- Menggunakan peran terkait layanan untuk AWS Panorama
- Cross-service bingung wakil pencegahan
- Memecahkan masalah identitas dan akses AWS Panorama

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di AWS Panorama.

Pengguna layanan — Jika Anda menggunakan layanan AWS Panorama untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur AWS Panorama untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di AWS Panorama, lihat. Memecahkan masalah identitas dan akses AWS Panorama

Administrator layanan — Jika Anda bertanggung jawab atas sumber daya AWS Panorama di perusahaan Anda, Anda mungkin memiliki akses penuh ke AWS Panorama. Tugas Anda adalah menentukan fitur dan sumber daya AWS Panorama mana yang harus diakses pengguna layanan

Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM Anda untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan AWS Panorama, lihat. Cara AWS Panorama bekerja dengan IAM

Administrator IAM — Jika Anda administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke AWS Panorama. Untuk melihat contoh kebijakan berbasis identitas AWS Panorama yang dapat Anda gunakan di IAM, lihat. Contoh kebijakan berbasis identitas AWS Panorama

Mengautentikasi dengan identitas

Autentikasi adalah cara Anda untuk masuk ke AWS menggunakan kredensial identitas Anda. Anda harus terautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk ke AWS sebagai identitas terfederasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. Pengguna AWS IAM Identity Center Pengguna (Pusat Identitas IAM), autentikasi Single Sign-On perusahaan Anda, dan kredensial Google atau Facebook Anda merupakan contoh identitas terfederasi. Saat Anda masuk sebagai identitas gabungan, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil suatu peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal akses AWS. Untuk informasi selengkapnya tentang cara masuk ke AWS, lihat Cara masuk ke Akun AWS dalam Panduan Pengguna AWS Sign-In.

Jika Anda mengakses AWS secara terprogram, AWS memberikan Kit Pengembangan Perangkat Lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan peralatan AWS, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang cara menggunakan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani permintaan API AWS dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Sebagai contoh, AWS menyarankan Anda menggunakan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Menggunakan autentikasi multi-faktor (MFA) di AWS dalam Panduan Pengguna IAM.

Pengguna root Akun AWS

Ketika membuat Akun AWS, Anda memulai dengan satu identitas masuk yang memiliki akses penuh ke semua Layanan AWS dan sumber daya di akun tersebut. Identitas ini disebut pengguna root Akun AWS dan diakses dengan cara masuk menggunakan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar tugas lengkap yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya andalkan kredensial temporer, dan bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM, sebaiknya rotasikan kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan kumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran tersebut dimaksudkan untuk dapat diambil oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, silakan lihat Kapan harus membuat pengguna IAM (bukan peran) dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM merupakan identitas dalam Akun AWS Anda yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara dalam AWS Management Console dengan berganti peran. Anda dapat mengambil peran dengan cara memanggil operasi API AWS CLI atau AWS atau menggunakan URL kustom.

Untuk informasi selengkapnya tentang metode untuk menggunakan peran, lihat Menggunakan peran IAM dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas gabungan, Anda dapat membuat peran dan menentukan izin untuk peran tersebut. Saat identitas terfederasi diautentikasi, identitas tersebut dikaitkan dengan peran dan diberikan izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga dalam Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda mengonfigurasi sekumpulan izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengaitkan izin yang ditetapkan ke peran dalam IAM. Untuk informasi tentang rangkaian izin, lihat Rangkaian izin dalam Panduan Pengguna AWS IAM Identity Center.
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (pengguna utama tepercaya) dengan akun berbeda untuk mengakses sumber daya yang ada di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, pada beberapa Layanan AWS, Anda dapat menyertakan kebijakan secara langsung ke sumber daya (bukan menggunakan peran sebagai proksi). Untuk mempelajari perbedaan antara kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya dalam Panduan Pengguna IAM.
- Akses lintas layanan – Sebagian Layanan AWS menggunakan fitur di Layanan AWS lainnya. Contoh, ketika Anda melakukan panggilan dalam layanan, umumnya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Suatu layanan mungkin melakukan hal tersebut menggunakan izin pengguna utama panggilan, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses maju (FAS) – Ketika Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan di AWS, Anda akan dianggap sebagai seorang pengguna utama. Saat menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian dilanjutkan oleh tindakan lain pada layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya diajukan saat layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan

tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Meneruskan sesi akses.

- Peran IAM – Peran layanan adalah peran IAM yang diambil layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
- Peran terkait layanan – Peran terkait layanan adalah tipe peran layanan yang terkait dengan Layanan AWS. Layanan tersebut dapat mengambil peran untuk melakukan sebuah tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

- Aplikasi yang berjalan di Amazon EC2 – Anda dapat menggunakan peran IAM untuk mengelola kredensial sementara untuk aplikasi yang berjalan di instans EC2 dan mengajukan permintaan API AWS CLI atau AWS. Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan peran AWS ke instans EC2 dan menyediakannya bagi semua aplikasinya, Anda dapat membuat profil instans yang dilampirkan ke instans tersebut. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2 dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat Kapan harus membuat peran IAM (bukan pengguna) dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengendalikan akses di AWS dengan membuat kebijakan dan melampirkannya ke identitas atau sumber daya AWS. Kebijakan adalah objek di AWS yang, ketika terkait dengan identitas atau sumber daya, akan menentukan izinnya. AWS mengevaluasi kebijakan-kebijakan tersebut ketika seorang pengguna utama (pengguna, pengguna root, atau sesi peran) mengajukan permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan di AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, silakan lihat Gambaran Umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses terhadap apa. Artinya, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat menjalankan peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk operasi. Sebagai contoh, anggap saja Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut dapat memperoleh informasi peran dari AWS Management Console, AWS CLI, atau API AWS.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan pengguna dan peran, di sumber daya mana, dan dengan ketentuan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan terkelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Akun AWS Anda. Kebijakan terkelola meliputi kebijakan yang dikelola AWS dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan inline, lihat Memilih antara kebijakan terkelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus menentukan pengguna utama dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan yang dikelola AWS dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat Gambaran umum daftar kontrol akses (ACL) di Panduan Developer Layanan Penyimpanan Ringkas Amazon.

Tipe kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda berdasarkan tipe kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan di mana Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan secara eksplisit terhadap salah satu kebijakan ini akan mengesampingkan izin tersebut. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) – SCP adalah kebijakan JSON yang menentukan izin maksimum untuk sebuah organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan mengelola beberapa akun Akun AWS yang dimiliki bisnis Anda secara terpusat. Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas dalam akun anggota, termasuk setiap Pengguna root akun AWS. Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat Cara kerja SCP dalam Panduan Pengguna AWS Organizations.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit di salah satu kebijakan ini akan membatalkan izin tersebut. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Jika beberapa jenis kebijakan diberlakukan untuk satu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan ketika ada beberapa jenis kebijakan, lihat Logika evaluasi kebijakan dalam Panduan Pengguna IAM.

Cara AWS Panorama bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke AWS Panorama, Anda harus memahami fitur IAM apa yang tersedia untuk digunakan dengan AWS Panorama. Untuk mendapatkan tampilan tingkat tinggi tentang cara AWS Panorama dan layanan AWS lainnya bekerja dengan IAM, AWSlihat layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Untuk ikhtisar izin, kebijakan, dan peran yang digunakan oleh AWS Panorama, lihat. Izin AWS Panorama

Contoh kebijakan berbasis identitas AWS Panorama

Secara default, pengguna dan peran IAM tidak memiliki izin untuk membuat atau memodifikasi sumber daya AWS Panorama. Mereka juga tidak dapat melakukan tugas menggunakan API AWS Management Console, AWS CLI, or AWS. Administrator IAM harus membuat kebijakan IAM yang memberikan izin kepada pengguna dan peran untuk melakukan operasi API tertentu pada sumber daya yang diperlukan. Administrator kemudian harus melampirkan kebijakan tersebut ke pengguna IAM atau grup yang memerlukan izin tersebut.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan di tab JSON dalam Panduan Pengguna IAM.

Topik

- Praktik terbaik kebijakan
- Menggunakan konsol AWS Panorama
- Izinkan pengguna melihat izin mereka sendiri

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya AWS Panorama di akun Anda. Tindakan ini dikenai biaya untuk Akun AWS Anda. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulai menggunakan kebijakan yang dikelola AWS dan beralih ke izin dengan hak akses paling rendah – Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan yang dikelola AWS yang memberikan izin untuk banyak kasus penggunaan umum. Kebijakan ini ada di Akun AWS Anda. Sebaiknya Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan AWS yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat kebijakan yang dikelola AWS atau kebijakan yang dikelola AWS untuk fungsi pekerjaan di Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukan ini dengan menentukan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, juga dikenal sebagai izin hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk menerapkan izin, lihat Kebijakan dan izin di IAM di Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Misalnya, Anda dapat menulis syarat kebijakan untuk menentukan bahwa semua pengajuan harus dikirim menggunakan SSL. Anda juga dapat menggunakan kondisi untuk memberi akses ke tindakan layanan jika digunakan melalui Layanan AWS yang spesifk, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Syarat di Panduan Pengguna IAM.
- Menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan berfungsi – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat validasi kebijakan Analizer Akses IAM di Panduan Pengguna IAM.
- Wajibkan autentikasi multi-faktor (MFA) – Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Akun AWS Anda, aktifkan MFA untuk keamanan tambahan. Untuk mewajibkan MFA saat operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses API yang dilindungi MFA di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.

Menggunakan konsol AWS Panorama

Untuk mengakses konsol AWS Panorama, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang sumber daya AWS Panorama di AWS akun Anda. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tersebut tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna IAM atau peran) dengan kebijakan tersebut.

Lihat informasi yang lebih lengkap di Kebijakan IAM berbasis identitas untuk AWS AWS AWS AWS Panorama

Izinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan para pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan pada konsol atau menggunakan AWS CLI atau AWS API secara terprogram.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
```

```
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSkebijakan terkelola untuk AWS Panorama

SebuahAWSkebijakan terkelola adalah kebijakan mandiri yang dibuat dan dikelola olehAWS.AWSkebijakan terkelola dirancang untuk memberikan izin untuk banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwaAWSkebijakan terkelola mungkin tidak memberikan izin paling sedikit hak istimewa untuk kasus penggunaan spesifik Anda karena tersedia untuk semuaAWSpelanggan untuk digunakan. Kami menyarankan Anda mengurangi izin lebih lanjut dengan mendefinisikankebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalamAWSkebijakan yang dikelola. JikaAWSmemperbarui izin yang didefinisikan dalamAWSkebijakan terkelola, pembaruan mempengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan.AWSkemungkinan besar akan memperbaruiAWSkebijakan terkelola saat baruLayanan AWSdiluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

AWS Panorama menyediakan kebijakan terkelola berikut. Untuk konten lengkap dan riwayat perubahan setiap kebijakan, lihat halaman tertaut di konsol IAM.

- AWSPanoramaFullAccess— Menyediakan akses penuh ke AWS Panorama, titik akses AWS Panorama di Amazon S3, kredensi alat diAWS Secrets Manager, dan log alat di AmazonCloudWatch. Termasuk izin untuk membuatperan terkait layananuntuk AWS Panorama.
- AWSPanoramaServiceLinkedRolePolicy— Memungkinkan AWS Panorama mengelola sumber daya di AWS IoT, AWS Secrets Manager, dan AWS Panorama.
- AWSPanoramaApplianceServiceRolePolicy— Memungkinkan AWS Panorama Appliance untuk mengunggah log keCloudWatch, dan untuk mendapatkan objek dari titik akses Amazon S3 yang dibuat oleh AWS Panorama.

Pembaruan AWS PanoramaAWSkebijakan terkelola

Tabel berikut menjelaskan pembaruan kebijakan terkelola untuk AWS Panorama.

Perubahan	Deskripsi	Tanggal
AWSPanoramaFullAccess — Permbaruan ke kebijakan yang sudah ada	Izin yang ditambahkan ke kebijakan pengguna untuk memungkinkan pengguna melihat grup log diCloudWatchLog konsol.	2022-01-13
AWSPanoramaFullAccess — Permbaruan ke kebijakan yang sudah ada	Menambahkan izin ke kebijakan pengguna untuk memungkinkan pengguna mengelola AWS Panoramaperan terkait layanan, dan untuk mengakses sumber daya AWS Panorama di layanan lain termasuk IAM, Amazon S3,CloudWatch, dan Manajer Rahasia.	2021-10-20
AWSPanoramaApplianceServiceRolePolicy – Kebijakan baru	Kebijakan baru untuk peran layanan AWS Panorama Appliance	2021-10-20
AWSPanoramaServiceLinkedRolePolicy – Kebijakan baru	Kebijakan baru untuk peran terkait layanan AWS Panorama.	2021-10-20
AWS Panorama mulai melacak perubahan	AWS Panorama mulai melacak perubahan nyaAWSkebijakan yang dikelola.	2021-10-20

Menggunakan peran terkait layanan untuk AWS Panorama

AWS Panorama menggunakan AWS Identity and Access Management peran yang terkait dengan layanan (IAM) . Peran yang terkait dengan layanan adalah tipe IAM role unik yang terkait langsung ke layanan. Peran yang terhubung dengan layanan ditentukan sebelumnya oleh AWS Panorama dan mencakup semua izin yang diperlukan layanan untuk menghubungi layanan AWS lainnya atas nama Anda.

Peran yang terhubung dengan layanan memudahkan pengaturan AWS Panorama karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. AWS Panorama menentukan izin peran yang terhubung dengan layanan, dan kecuali ditentukan sebaliknya, hanya AWS Panorama yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya AWS Panorama karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat layanan AWS yang bekerja bersama IAM dan mencari layanan yang memuat Ya di dalam kolom Peran tertaut layanan. Pilih Yes (Ya) bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Bagian

- Izin peran terkait layanan untuk AWS Panorama
- Membuat peran yang terhubung dengan layanan untuk AWS Panorama
- Mengedit peran terkait layanan untuk AWS Panorama
- Menghapus peran terkait layanan untuk AWS Panorama
- Wilayah yang didukung untuk peran yang terhubung dengan layanan AWS Panorama

Izin peran terkait layanan untuk AWS Panorama

AWS Panoramamenggunakan peran yang ditautkan dengan layananAWSServicerOleforawSpanorama— Memungkinkan AWS Panorama mengelola sumber daya di AWS IoT, AWS Secrets Manager, dan AWS Panorama

Peran tertaut layanan AWSServiceRoleForAWSSpanorama memercayai layanan berikut untuk mengambil peran tersebut:

- `panorama.amazonaws.com`

Kebijakan izin peran mengizinkanAWS Panoramauntuk menyelesaikan tindakan berikut:

- PemantauanAWS Panoramasumber daya
- KelolaAWS IoTsumber daya untukAWS PanoramaAlat
- AksesAWS Secrets Managerrahasia untuk mendapatkan kredensi kamera

Untuk daftar lengkap izin,lihat kebijakan awspanoramaservicelInkedRolePolicydi konsol IAM.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat Izin peran yang terkait dengan layanan dalam Panduan Pengguna IAM.

Membuat peran yang terhubung dengan layanan untuk AWS Panorama

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda mendaftarkan alat diAWS Management Console, yangAWS CLI, atauAWSAPIAWS Panoramamenciptakan peran yang ditautkan dengan layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda mendaftarkan alat,AWS Panoramamenciptakan peran yang ditautkan dengan layanan lagi untuk Anda.

Mengedit peran terkait layanan untuk AWS Panorama

AWS Panoramatidak mengizinkan Anda mengedit peran yang terhubung dengan layanan AWSServiceRoleForAWSSpanorama. Setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi lebih lanjut, lihat Mengedit peran terkait layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk AWS Panorama

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, kami menyarankan Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran yang terhubung dengan layanan sebelum menghapusnya secara manual.

Untuk menghapusAWS Panoramasumber daya yang digunakan oleh AWSServiceRoleForawSpanorama, gunakan prosedur di bagian berikut dari panduan ini.

- Hapus versi dan aplikasi
- Membatalkan pendaftaran alat

> **Note**
>
> Jika layanan AWS Panorama menggunakan peran tersebut ketika Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus peran yang ditautkan dengan layanan AWSServiceRoleForAWSSpanorama, konsol IAM,AWS CLI, atauAWSAPI Untuk informasi lebih lanjut, lihat Menghapus peran terkait layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran yang terhubung dengan layanan AWS Panorama

AWS Panorama mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihatAWSWilayah dan titik akhir.

Cross-service bingung wakil pencegahan

Masalah deputi yang bingung adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memaksa entitas yang lebih istimewa untuk melakukan tindakan tersebut. MasukAWS, peniruan lintas layanan dapat mengakibatkan masalah wakil bingung. Peniruan lintas layanan dapat terjadi ketika satu layanan (layanan panggilan) panggilan layanan lain (yangdisebut layanan). Layanan panggilan dapat dimanipulasi untuk menggunakan izin untuk bertindak atas sumber daya pelanggan lain dengan cara yang seharusnya tidak memiliki izin untuk mengakses. Untuk mencegah hal ini,AWSmenyediakan alat yang membantu Anda melindungi data Anda untuk semua layanan dengan prinsipal layanan yang telah diberikan akses ke sumber daya di akun Anda.

Sebaiknya gunakan`aws:SourceArn`dan`aws:SourceAccount`kunci konteks kondisi global dalam kebijakan sumber daya untuk membatasi izin yangAWS Panoramamemberikan layanan lain untuk sumber daya. Jika Anda menggunakan kedua kunci konteks kondisi

global,`aws:SourceAccount`nilai dan akun di`aws:SourceArn`nilai harus menggunakan ID akun yang sama bila digunakan dalam pernyataan kebijakan yang sama.

Nilai dari`aws:SourceArn`harus menjadi ARN dariAWS Panoramaperangkat

Cara paling efektif untuk melindungi dari masalah wakil bingung adalah dengan menggunakan`aws:SourceArn`kunci konteks kondisi global dengan ARN penuh sumber daya. Jika Anda tidak mengetahui ARN penuh dari sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan`aws:SourceArn`kunci kondisi konteks global dengan wildcard (`*`) untuk bagian yang tidak diketahui dari ARN. Misalnya, `arn:aws:`*`servicename`*`::`*`123456789012`*`:*`.

Untuk petunjuk tentang mengamankan peran layanan yangAWS Panoramamenggunakan untuk memberikan izin kepadaAWS PanoramaAppliance, lihatMengamankan peran alat.

Memecahkan masalah identitas dan akses AWS Panorama

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan AWS Panorama dan IAM.

Topik

- Saya tidak berwenang untuk melakukan tindakan di AWS Panorama
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar AWS akun saya untuk mengakses sumber daya AWS Panorama saya

Saya tidak berwenang untuk melakukan tindakan di AWS Panorama

Jika AWS Management Console memberi tahu bahwa Anda tidak diotorisasi untuk melakukan tindakan, Anda harus menghubungi administrator untuk mendapatkan bantuan. Administrator adalah orang yang memberikan nama pengguna dan kata sandi kepada Anda.

Contoh kesalahan berikut terjadi ketika pengguna `mateojackson` IAM mencoba menggunakan konsol untuk melihat detail tentang alat tetapi tidak memiliki `panorama:DescribeAppliance` izin.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 panorama:DescribeAppliance on resource: my-appliance
```

Dalam hal ini, Mateo meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan dia mengakses sumber daya `my-appliance` menggunakan tindakan `panorama:DescribeAppliance`.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan bahwa Anda tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke AWS Panorama.

Sebagian Layanan AWS mengizinkan Anda untuk memberikan peran yang sudah ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait-layanan. Untuk melakukan tindakan tersebut, Anda harus memiliki izin untuk memberikan peran pada layanan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di AWS Panorama. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda membutuhkan bantuan, hubungi administrator AWS Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar AWS akun saya untuk mengakses sumber daya AWS Panorama saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau pengguna di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi pengguna akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa hal berikut:

- Untuk mengetahui apakah AWS Panorama mendukung fitur-fitur ini, lihat. Cara AWS Panorama bekerja dengan IAM

- Untuk mempelajari cara memberikan akses ke sumber daya di seluruh Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di Akun AWS lainnya yang Anda miliki dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses ke sumber daya Anda ke pihak ketiga Akun AWS, lihat Menyediakan akses ke akun Akun AWS yang dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (gabungan identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara penggunaan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Perbedaan antara peran IAM dan kebijakan berbasis sumber daya di Panduan Pengguna IAM.

Validasi kepatuhan untuk AWS Panorama

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

  > **Note**
  >
  > Tidak semua memenuhi Layanan AWS syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber

daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.

- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Pertimbangan tambahan saat orang hadir

Di bawah ini adalah beberapa praktik terbaik yang perlu dipertimbangkan saat menggunakan AWS Panorama untuk skenario di mana orang mungkin hadir:

- Pastikan bahwa Anda mengetahui dan mematuhi semua hukum dan peraturan yang berlaku untuk kasus penggunaan Anda. Ini mungkin termasuk undang-undang yang terkait dengan pemosisian dan bidang pandang kamera Anda, persyaratan pemberitahuan dan signage saat menempatkan dan menggunakan kamera, dan hak-hak orang yang mungkin ada dalam video Anda, termasuk hak privasi mereka.
- Pertimbangkan efek kamera Anda pada orang dan privasi mereka. Selain persyaratan hukum, pertimbangkan apakah akan tepat untuk menempatkan pemberitahuan di area di mana kamera Anda berada, dan apakah kamera harus ditempatkan di depan mata dan bebas dari oklusi, sehingga orang tidak terkejut bahwa mereka mungkin berada di kamera.
- Memiliki kebijakan dan prosedur yang tepat untuk pengoperasian kamera Anda dan meninjau data yang diperoleh dari kamera.
- Pertimbangkan kontrol akses yang sesuai, batasan penggunaan, dan periode retensi untuk data yang diperoleh dari kamera Anda.

Keamanan infrastruktur di AWS Panorama

Sebagai layanan terkelola, AWS Panorama dilindungi olehAWSkeamanan jaringan global. Untuk informasi tentangAWSLayanan keamanan dan bagaimanaAWSmelindungi infrastruktur, lihatAWSKeamanan Cloud. Untuk mendesain AndaAWSlingkungan menggunakan praktik terbaik untuk keamanan infrastruktur, lihatPerlindungan InfrastrukturdiPilar KeamananAWSKerangka Kerja yang Diarsiteksikan dengan Baik.

Anda menggunakanAWSmempublikasikan panggilan API untuk mengakses AWS Panorama melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Transportasi (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Cipher suite dengan perfect forward secrecy (PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Menerapkan AWS Panorama Appliance di pusat data Anda

AWS Panorama Appliance membutuhkan akses internet untuk berkomunikasiAWSlayanan. Ini juga membutuhkan akses ke jaringan internal kamera Anda. Penting untuk mempertimbangkan konfigurasi jaringan Anda dengan hati-hati dan hanya menyediakan setiap perangkat akses yang dibutuhkannya. Hati-hati jika konfigurasi Anda memungkinkan AWS Panorama Appliance bertindak sebagai jembatan ke jaringan kamera IP yang sensitif.

Anda bertanggung jawab untuk hal-hal berikut:

- Keamanan jaringan fisik dan logis AWS Panorama Appliance.
- Mengoperasikan kamera yang terpasang di jaringan dengan aman saat Anda menggunakan AWS Panorama Appliance.
- Menjaga perangkat lunak AWS Panorama Appliance dan kamera diperbarui.
- Mematuhi hukum atau peraturan yang berlaku terkait dengan konten video dan gambar yang Anda kumpulkan dari lingkungan produksi Anda, termasuk yang terkait dengan privasi.

AWS Panorama Appliance menggunakan aliran kamera RTSP yang tidak terenkripsi. Untuk informasi selengkapnya tentang menghubungkan AWS Panorama Appliance ke jaringan Anda, lihatMenghubungkan AWS Panorama Appliance ke jaringan Anda. Untuk detail tentang enkripsi, lihatPerlindungan data di AWS Panorama.

Perangkat lunak lingkungan runtime di AWS Panorama

AWS Panorama menyediakan perangkat lunak yang menjalankan kode aplikasi Anda di lingkungan berbasis Ubuntu Linux di AWS Panorama Appliance. AWS Panorama bertanggung jawab untuk menjaga perangkat lunak dalam gambar alat tetap up to date. AWS Panorama secara teratur merilis pembaruan perangkat lunak, yang dapat Anda terapkan[menggunakan konsol AWS Panorama](#).

Anda dapat menggunakan pustaka dalam kode aplikasi Anda dengan menginstalnya di aplikasi`Dockerfile`. Untuk memastikan stabilitas aplikasi di seluruh build, pilih versi tertentu dari setiap pustaka. Perbarui dependensi Anda secara teratur untuk mengatasi masalah keamanan.

Rilis

Tabel berikut menunjukkan kapan fitur dan pembaruan perangkat lunak dirilis untuk AWS Panorama layanan, perangkat lunak, dan dokumentasi. Untuk memastikan bahwa Anda memiliki akses ke semua fitur, perbarui AWS Panorama Appliance Anda ke versi perangkat lunak terbaru. Untuk informasi selengkapnya tentang rilis, lihat topik tertaut.

Perubahan	Deskripsi	Tanggal
Pembaruan perangkat lunak alat	Versi 7.0.13 adalah pembaruan versi utama yang mengubah cara alat mengelola pembaruan perangkat lunak. Jika Anda membatasi komunikasi jaringan keluar dari alat, atau menghubungkannya ke subnet VPC pribadi, Anda harus mengizinkan akses ke titik akhir dan port tambahan sebelum menerapkan pembaruan. Untuk informasi selengkapnya, lihat log perubahan.	28 Desember 2023
Pembaruan perangkat lunak alat	Versi 6.2.1 termasuk perbaikan bug. Untuk informasi selengkapnya, lihat log perubahan.	September 6, 2023
Pembaruan perangkat lunak alat	Versi 6.0.8 mencakup perbaikan bug dan peningkatan keamanan. Untuk informasi selengkapnya, lihat log perubahan.	6 Juli 2023
Pembaruan perangkat lunak alat	Versi 5.1.7 mencakup perbaikan bug dan peningkat	31 Maret 2023

	an penanganan kesalahan. Untuk informasi selengkapnya, lihat log perubahan.	
Pembaruan konsol	Anda sekarang dapat membeli AWS Panorama Appliance dari konsol manajemen. Untuk memberikan izin pengguna untuk membeli perangkat, lihat Kebijakan IAM berbasis identitas untuk AWS Panorama.	2 Februari 2023
Pembaruan perangkat lunak alat	Versi 5.0.74 mencakup perbaikan bug dan peningkatan penanganan kesalahan. Untuk informasi selengkapnya, lihat log perubahan.	23 Januari 2023
Pembaruan API	Menambahkan `AllowMajorVersionUpdate` opsi `OTAJobConfig` untuk membuat pembaruan versi utama perangkat lunak alat opt-in. Untuk informasi lebih lanjut, lihat CreateJobForDevices.	19 Januari 2023
Alat baru untuk pengembang	Alat baru, “sideloading”, tersedia di repositori sampel. AWS Panorama GitHub Anda dapat menggunakan alat ini untuk memperbarui kode aplikasi tanpa membangun dan menyebarkan wadah. Untuk informasi lebih lanjut, lihat README.	16 November 2022

Pembaruan gambar dasar aplikasi	Versi 1.2.0 menambahkan opsi batas waktu `kevideo_in.get()` , menetapkan variabel `AWS_REGION` lingkungan, dan meningkatkan penanganan kesalahan. Untuk informasi selengkapnya, lihat log perubahan.	16 November 2022
Pembaruan perangkat lunak alat	Versi 5.0.42 mencakup perbaikan bug dan pembaruan keamanan. Untuk informasi selengkapnya, lihat log perubahan.	16 November 2022
Pembaruan perangkat lunak alat	Versi 5.0.7 menambahkan dukungan untuk me-reboot peralatan dari jarak jauh dan menjeda aliran kamera dari jarak jauh. Untuk informasi selengkapnya, lihat log perubahan.	13 Oktober 2022
Pembaruan perangkat lunak alat	Versi 4.3.93 menambahkan dukungan untuk mengambil log dari perangkat offline. Untuk informasi selengkapnya, lihat log perubahan.	Agustus 24, 2022
Pembaruan perangkat lunak alat	Versi 4.3.72 mencakup perbaikan bug dan pembaruan keamanan. Untuk informasi selengkapnya, lihat log perubahan.	Juni 23, 2022

AWS PrivateLinkdukungan	AWS Panoramamendukung titik akhir VPC untuk mengelola AWS Panorama sumber daya dari subnet pribadi. Untuk informasi selengkapnya, lihat Menggunakan titik akhir VPC.	2 Juni 2022
Pembaruan perangkat lunak alat	Versi 4.3.55 meningkatkan pemanfaatan penyimpanan untuk log. `console_output` Untuk informasi selengkapnya, lihat log perubahan.	Mei 5, 2022
Lenovo ThinkEdge® SE70	Alat baru untuk AWS Panorama tersedia dari Lenovo. Lenovo ThinkEdge® SE70, didukung oleh Nvidia Jetson Xavier NX, mendukung fitur yang sama dengan Appliance. AWS Panorama Untuk informasi selengkap nya, lihat Perangkat yang kompatibel.	April 6, 2022
Pembaruan gambar dasar aplikasi	Versi 1.1.0 meningkatkan kinerja saat menjalankan utas latar belakang dan menambahkan bendera (is_cached) ke objek media yang menunjukkan apakah gambar segar. Untuk informasi selengkapnya, lihat gallery.e cr.aws.	29 Maret 2022

Pembaruan perangkat lunak alat	Versi 4.3.45 menambahkan dukungan untuk akses GPU dan port masuk. Untuk informasi selengkapnya, lihat log perubahan.	24 Maret 2022
Pembaruan perangkat lunak alat	Versi 4.3.35 meningkatkan keamanan dan kinerja. Untuk informasi selengkapnya, lihat log perubahan.	Februari 22, 2022
Kebijakan terkelola yang diperbarui	AWS Identity and Access Managementkebijakan terkelola untuk AWS Panorama telah diperbarui. Untuk detailnya, lihat kebijakan terkelola AWS.	Januari 13, 2022
Log penyediaan	Dengan perangkat lunak alat 4.3.23, alat menulis log ke drive USB selama penyediaan. Untuk informasi selengkapnya, lihat Log.	Januari 13, 2022
Konfigurasi server NTP	Anda sekarang dapat mengkonfigurasi AWS Panorama Appliance untuk menggunakan server NTP tertentu untuk sinkronisasi jam. Konfigurasikan pengaturan NTP selama pengaturan alat dengan pengaturan jaringan lainnya. Untuk informasi selengkapnya, lihat Menyiapkan.	Januari 13, 2022

Wilayah tambahan	AWS Panoramasekarang tersedia di Wilayah Asia Pasifik (Singapura) dan Asia Pasifik (Sydney).	Januari 13, 2022
Pembaruan perangkat lunak alat	Versi 4.3.4 menambahkan dukungan untuk `precisionMode` pengaturan untuk model dan memperbarui perilaku logging. Untuk informasi selengkapnya, lihat log perubahan.	November 8, 2021
Kebijakan terkelola yang diperbarui	AWS Identity and Access Managementkebijakan terkelola untuk AWS Panorama telah diperbarui. Untuk detailnya, lihat kebijakan terkelola AWS.	20 Oktober 2021
Ketersediaan umum	AWS Panoramasekarang tersedia untuk semua pelanggan di Wilayah AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia), dan Kanada (Tengah). Untuk membeli AWS Panorama Peralatan, kunjungi AWS Panorama.	20 Oktober 2021
Pratinjau	AWS Panoramatersedia melalui undangan di Wilayah AS Timur (Virginia N.) dan AS Barat (Oregon).	1 Desember 2020

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.