Panduan Developerr

AWS Panorama

AWS Panorama: Panduan Developerr

Table of Contents

Pemberitahuan akhir dukungan: Pada 31 Mei 2026, AWS akan mengakhiri dukungan untuk AWS Panorama. Setelah 31 Mei 2026, Anda tidak akan lagi dapat mengakses AWS Panorama konsol atau AWS Panorama sumber daya. Untuk informasi lebih lanjut, lihat AWS Panorama akhir dukungan.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.

Apa itu AWS Panorama?

AWS Panorama adalah layanan yang membawa visi komputer ke jaringan kamera lokal Anda. Anda menginstal AWS Panorama Appliance atau perangkat lain yang kompatibel di pusat data Anda, mendaftarkannya AWS Panorama, dan menyebarkan aplikasi visi komputer dari cloud. AWS Panorama bekerja dengan kamera jaringan protokol streaming waktu nyata (RTSP) Anda yang ada. Alat ini menjalankan aplikasi visi komputer yang aman dari AWS Mitra, atau aplikasi yang Anda buat sendiri dengan SDK AWS Panorama Aplikasi.

AWS Panorama Appliance adalah alat tepi ringkas yang menggunakan system-on-module (SOM) yang kuat yang dioptimalkan untuk beban kerja pembelajaran mesin. Alat ini dapat menjalankan beberapa model visi komputer terhadap beberapa aliran video secara paralel dan menampilkan hasilnya secara real time. Ini dirancang untuk digunakan dalam pengaturan komersial dan industri dan dinilai untuk perlindungan debu dan cairan (IP-62).

AWS Panorama Appliance memungkinkan Anda menjalankan aplikasi visi komputer mandiri di edge, tanpa mengirim gambar ke AWS Cloud. Dengan menggunakan AWS SDK, Anda dapat berintegrasi dengan layanan AWS lainnya dan menggunakannya untuk melacak data dari aplikasi dari waktu ke waktu. Dengan mengintegrasikan dengan layanan AWS lainnya, Anda dapat menggunakan AWS Panorama untuk melakukan hal berikut:

- Menganalisis 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 masa depan.
- Terima peringatan keamanan situs — Pantau area terlarang di lokasi industri. Saat aplikasi mendeteksi situasi yang berpotensi tidak aman, unggah gambar ke Amazon Simple Storage Service (Amazon S3) dan kirim pemberitahuan ke topik Simple Notification Service Amazon (Amazon SNS) sehingga penerima dapat mengambil tindakan korektif.
- Tingkatkan kontrol kualitas - Pantau output jalur perakitan untuk mengidentifikasi suku cadang yang tidak sesuai dengan persyaratan. Sorot gambar bagian yang tidak sesuai dengan teks dan kotak pembatas dan tampilkan pada monitor untuk ditinjau oleh tim kontrol kualitas Anda.
- Kumpulkan data pelatihan dan uji — Unggah gambar objek yang tidak dapat diidentifikasi oleh model visi komputer Anda, atau di mana kepercayaan model pada tebakannya adalah batas. Gunakan aplikasi tanpa server untuk membuat antrian gambar yang perlu diberi tag. Tandai gambar dan gunakan untuk melatih kembali model di Amazon SageMaker AI.

AWS Panorama menggunakan layanan AWS lainnya untuk mengelola AWS Panorama Appliance, mengakses model dan kode, serta menyebarkan aplikasi. AWS Panorama melakukan sebanyak mungkin tanpa mengharuskan Anda untuk berinteraksi dengan layanan lain, tetapi pengetahuan tentang layanan berikut dapat membantu Anda memahami cara AWS Panorama kerja.

- SageMaker AI — Anda dapat menggunakan SageMaker AI untuk mengumpulkan data pelatihan dari kamera atau sensor, membangun model pembelajaran mesin, dan melatihnya untuk visi komputer. AWS Panorama menggunakan SageMaker AI Neo untuk mengoptimalkan model agar berjalan di AWS Panorama Appliance.
- Amazon S3 — Anda menggunakan jalur akses Amazon S3 untuk menampilkan kode aplikasi, model, dan file konfigurasi untuk penerapan ke Appliance. AWS Panorama
- AWS IoT— AWS Panorama menggunakan AWS IoT layanan untuk memantau keadaan AWS Panorama Appliance, mengelola pembaruan perangkat lunak, dan menyebarkan aplikasi. Anda tidak perlu menggunakan AWS IoT secara langsung.

Untuk memulai dengan AWS Panorama Appliance dan mempelajari lebih lanjut tentang layanan ini, lanjutkan keMemulai dengan AWS Panorama.

AWS Panorama akhir dukungan

Setelah mempertimbangkan dengan cermat, kami memutuskan untuk mengakhiri dukungan untuk AWS Panorama, efektif 31 Mei 2026. AWS Panorama tidak akan lagi menerima pelanggan baru mulai 20 Mei 2025. Sebagai pelanggan lama dengan akun yang mendaftar untuk layanan sebelum 20 Mei 2025, Anda dapat terus menggunakan fitur AWS Panorama. Setelah 31 Mei 2026, Anda tidak lagi dapat menggunakan AWS Panorama.

Alternatif untuk AWS Panorama

Jika Anda tertarik dengan alternatif AWS Panorama, AWS memiliki opsi untuk pembeli dan pembangun.

Untuk out-of-the-box solusi, AWS Partner Network menawarkan solusi dari beberapa mitra. Anda dapat menelusuri solusi di AWS Solutions Library dari banyak mitra kami. Solusi mitra ini mencakup opsi untuk aplikasi perangkat keras, perangkat lunak, perangkat lunak sebagai layanan (SaaS), solusi terkelola, atau implementasi khusus berdasarkan kebutuhan Anda. Pendekatan ini memberikan solusi yang menangani kasus penggunaan Anda tanpa mengharuskan Anda memiliki

keahlian dalam visi komputer, AI, atau pengembangan aplikasi. Ini biasanya memberikan waktu yang lebih cepat untuk menilai dengan memanfaatkan keahlian khusus dari AWS Partners.

Jika Anda lebih suka membangun solusi Anda sendiri, AWS menawarkan alat dan layanan AI untuk membantu Anda mengembangkan aplikasi visi komputer berbasis AI dan mengelola aplikasi dan perangkat di edge. Amazon SageMaker menyediakan seperangkat alat untuk membuat, melatih, dan menerapkan model ML untuk kasus penggunaan Anda dengan infrastruktur, alat, dan alur kerja yang dikelola sepenuhnya. Selain memungkinkan Anda membuat model Anda sendiri, Amazon SageMaker JumpStart menawarkan algoritme visi komputer bawaan yang dapat disesuaikan dengan kasus penggunaan spesifik Anda.

Untuk mengelola perangkat dan aplikasi di edge, AWS IoT Greengrass adalah solusi yang terbukti dan aman untuk menerapkan dan memperbarui aplikasi untuk perangkat IoT. Untuk implementasi berbasis server, AWS Systems Manager menyediakan serangkaian alat untuk mengelola server dan Amazon EKS Anywhere atau ECS Anywhere dapat mengelola wadah aplikasi di server edge. Amazon menyediakan beberapa panduan untuk mengelola perangkat edge, bersama dengan sumber daya tambahan di Bagian 4 Securing Internet of Things (IoT) dengan whitepaper AWS. Pendekatan pembangun ini memberi Anda alat untuk mempercepat pengembangan AI dan manajemen perangkat Anda sambil memberikan fleksibilitas lengkap untuk membangun solusi yang memenuhi persyaratan Anda yang tepat dan terintegrasi dengan infrastruktur perangkat keras dan perangkat lunak yang ada. Ini biasanya memberikan biaya operasi yang lebih rendah untuk suatu solusi.

Migrasi dari AWS Panorama

Untuk memindahkan aplikasi yang ada dari AWS Panorama ke implementasi alternatif, Anda perlu mengganti perangkat keras yang ada, memigrasikan aplikasi dari layanan AWS Panorama, dan menerapkan manajemen tepi dan keamanan untuk solusi baru. Masing-masing area tersebut akan dieksplorasi secara rinci di bawah ini:

Penggantian Perangkat Keras

Alat AWS Panorama yang ada didasarkan pada platform Nvidia Jetson Xavier. Perangkat keras dapat diganti dengan off-the-shelf perangkat serupa berdasarkan platform Nvidia Jetson generasi saat ini yang memenuhi kebutuhan Anda, atau server tepi. Sementara sebagian besar penerapan AWS Panorama dapat diganti dengan perangkat serupa, kami telah melihat beberapa pelanggan yang menggunakan sejumlah besar kamera di satu lokasi menemukan bahwa server adalah alternatif yang lebih baik.

Migrasi Aplikasi

Aplikasi AWS Panorama perlu ditulis ulang untuk menghilangkan penggunaan panggilan API khusus AWS Panorama. Aplikasi AWS Panorama hanya mendukung input video melalui feed Real-Time Streaming Protocol (RTSP) menggunakan H.264 dan input video tersebut disediakan menggunakan node kamera di SDK perangkat AWS Panorama.

Untuk mem-port aplikasi yang ada, Anda perlu mengimplementasikan kelas aplikasi yang mirip dengan AWS Panorama sehingga kode yang ada sebagian besar dapat digunakan kembali. Contoh kode tersedia dalam file banner-code.zip yang menunjukkan contoh implementasi ini menggunakan PyAV dan OpenCV.

Ini adalah pendekatan sederhana dengan jumlah perubahan kode minimal, tetapi memiliki banyak batasan yang sama dengan implementasi berbasis AWS Panorama saat ini dalam hal jenis aliran video yang didukung.

Pilihan lain adalah merancang ulang aplikasi untuk memanfaatkan sumber daya sistem dengan lebih baik dan untuk mendukung kemampuan aplikasi baru. Untuk opsi ini, Anda menggunakan GStreameratau DeepStreammengimplementasikan pipeline media dari sumber media ke hasil inferensi dan logika bisnis, atau menggunakan implementasi runtime machine learning (ML) yang lebih kaya fitur dan berkinerja lebih baik, seperti server inferensi Nvidia Triton. Pendekatan ini membutuhkan perubahan pada lebih banyak saluran pemrosesan video, tetapi keduanya lebih efisien dan memungkinkan lebih banyak fleksibilitas untuk mendukung rentang codec, jenis kamera, dan sensor lainnya yang lebih luas.

Manajemen dan Keamanan Tepi

Terlepas dari saluran media, Anda juga harus menerapkan penyimpanan aman untuk kredensialnya, misalnya nama pengguna dan kata sandi aliran RTSP. AWS menyediakan berbagai cara untuk menyimpan parameter aplikasi dengan aman:

- Layanan AWS IoT Device Shadow digunakan untuk menyimpan parameter yang diteruskan ke aplikasi, serta untuk melacak status aplikasi pada perangkat edge.
- AWS Secrets Manager digunakan untuk menyimpan kredensil semacam itu untuk melindungi kredensil dengan lebih baik untuk mengakses aliran media.
- Jika Anda menggunakan Amazon EKS atau Amazon ECS, Anda juga dapat menggunakan penyimpanan Parameter AWS System Manager yang aman untuk kredensi dan parameter aplikasi lainnya.

Pilihannya tergantung pada persyaratan keamanan aplikasi, serta AWS produk lain yang Anda rencanakan untuk digunakan untuk mengimplementasikan aplikasi Anda.

Saat mengganti alat AWS Panorama dengan perangkat edge generik, Anda juga harus menerapkan fitur keamanan yang diperlukan untuk aplikasi Anda dan mengonfigurasi perangkat agar sesuai dengan persyaratan keamanan Anda. AWS memberikan panduan tentang hal ini dalam Pilar Keamanan AWS Well-Architected Framework. Sementara kerangka kerja terutama berfokus pada aplikasi cloud, sebagian besar prinsip juga berlaku untuk perangkat edge. Selain itu, Anda harus menggunakan fitur keamanan perangkat keras dari solusi yang dipilih, seperti integrasi keamanan perangkat keras AWS IoT Greengrass V2, dan menggunakan fitur keamanan yang disediakan oleh OS dan/atau perangkat yang dipilih, seperti enkripsi disk penuh.

Ringkasan

Meskipun AWS Panorama berencana untuk ditutup pada 31 Mei 2026, AWS menawarkan serangkaian layanan dan solusi AI/ML yang kuat dalam bentuk SageMaker alat Amazon untuk membangun model visi komputer dan layanan manajemen perangkat, seperti AWS IoT Greengrass, Amazon EKS dan Amazon ECS Anywhere dan AWS System Manager untuk mendukung pengembangan solusi serupa. AWS juga memiliki berbagai penawaran dari mitra di AWS Partner Network jika Anda lebih suka membeli daripada membangun solusi. Contoh kode dan panduan implementasi disediakan untuk membantu bermigrasi ke solusi alternatif jika Anda memilihnya. Anda harus mengeksplorasi opsi-opsi ini untuk menentukan apa yang terbaik untuk kebutuhan spesifik Anda.

Untuk detail selengkapnya, lihat sumber daya berikut:

- Panduan SageMaker Pengembang Amazon — Dokumentasi terperinci tentang cara membuat model atau bekerja dengan algoritme visi komputer bawaan yang tersedia di SageMaker JumpStart.
- Panduan Pengembang AWS IoT Core - Dokumentasi terperinci tentang cara menghubungkan dan mengelola Perangkat IoT.
- Panduan Pengembang AWS IoT Greengrass V2 — Dokumentasi terperinci tentang cara membuat, menerapkan, dan mengelola aplikasi IoT di perangkat Anda.
- Panduan Pengembang ECS Anywhere - Dokumentasi terperinci tentang menjalankan ECS di tepi.
- Panduan Praktik Terbaik EKS Anywhere - Dokumentasi terperinci tentang menjalankan EKS di tepi.

- AWS Solutions Library — Penawaran mitra dari berbagai penyedia yang menawarkan solusi visi komputer yang dibuat sebelumnya atau disesuaikan.
- Panorama FAQs - Informasi Panorama Tambahan.

Pertanyaan yang Sering Diajukan

Berapa waktu penghentian Panorama?

Pengumuman itu dibuat pada 20 Mei 2025. Setelah tanggal ini, pelanggan yang tidak aktif pada layanan tidak akan lagi memiliki akses ke Panorama. Pelanggan aktif akan dapat terus menggunakan layanan ini secara normal hingga 31 Mei 2026. Pelanggan memiliki waktu hingga saat itu untuk memindahkan aplikasi mereka ke solusi alternatif dan memigrasikan aplikasi Panorama. Setelah 31 Mei 2026, aplikasi apa pun yang mencoba mengakses layanan Panorama tidak akan berfungsi lagi dan perangkat Panorama tidak akan berfungsi lagi.

Bagaimana pelanggan yang ada akan terpengaruh?

Pelanggan yang sudah ada dapat terus menggunakan layanan ini secara normal hingga 31 Mei 2026. Setelah itu, aplikasi yang mencoba mengakses Panorama tidak akan berfungsi lagi. Perangkat Panorama juga tidak akan berfungsi lagi setelah tanggal tersebut.

Apakah pelanggan baru diterima?

Tidak. Per 20 Mei 2025, hanya pelanggan yang merupakan pengguna aktif Panorama yang akan memiliki akses ke layanan ini. Jika pelanggan memiliki aplikasi dalam layanan dari penggunaan sebelumnya yang perlu mereka akses, mereka dapat membuat kasus dengan dukungan pelanggan untuk meminta akses ke akun mereka. Jika pelanggan tidak memiliki penggunaan layanan sebelumnya, mereka tidak akan diberikan akses.

Apa alternatif yang dapat dijelajahi pelanggan?

AWS menawarkan berbagai layanan yang dapat menggantikan kemampuan Panorama. Kami menyarankan pelanggan menggunakan off-the-shelf perangkat keras dan mengelola perangkat dan aplikasi melalui kombinasi AWS IoT Core, AWS IoT Greengrass, Amazon AKS Anywhere, Amazon ECS Anywhere, dan/atau AWS System Manager yang memenuhi persyaratan mereka. AWS Partner Network juga memiliki beberapa solusi yang tersedia dari mitra dengan keahlian Computer Visions tertentu yang dapat dipertimbangkan oleh pelanggan.

Bagaimana pelanggan dapat bermigrasi dari Panorama?

Aplikasi Panorama perlu dimodifikasi untuk menghapus dependensi apa pun pada khusus Panorama APIs, yang terutama terkait dengan koneksi kamera dan streaming. AWS telah menyediakan kode contoh untuk menunjukkan cara membuat perubahan ini. Setelah dependensi tersebut dihapus, aplikasi dapat dipindahkan ke platform perangkat keras alternatif.

Jika saya mengalami masalah pada atau setelah 20 Mei 2025, dukungan apa yang akan tersedia?

AWS akan terus memberikan dukungan untuk Panorama hingga akhir periode pemberitahuan penghentian (31 Mei 2026). Untuk persyaratan dukungan apa pun, pelanggan harus memasukkan kasus dukungan melalui saluran dukungan normal mereka. AWS akan memberikan pembaruan keamanan, perbaikan bug, dan peningkatan ketersediaan.

Saya tidak dapat bermigrasi sebelum 31 Mei 2026. Bisakah tanggal diperpanjang?

Kami yakin bahwa alternatif yang tersedia untuk Panorama memungkinkan pelanggan untuk bermigrasi ke solusi alternatif pada 31 Mei 2026 dan kami tidak memiliki rencana untuk memperpanjang ketersediaan layanan melewati tanggal tersebut.

Apakah aplikasi edge saya akan terus berfungsi setelah layanan berakhir?

Tidak. Perangkat dan aplikasi Panorama bergantung pada konektivitas ke layanan cloud Panorama. Setelah layanan itu dihentikan pada 31 Mei 2026, baik aplikasi Panorama maupun perangkat Panorama tidak akan terus berfungsi.

Memulai dengan AWS Panorama

Untuk memulai AWS Panorama, pertama-tama pelajari tentang konsep layanan dan terminologi yang digunakan dalam panduan ini. Kemudian Anda dapat menggunakan AWS Panorama konsol untuk mendaftarkan AWS 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 menggunakan AWS Panorama Appliance dan kamera yang mengalirkan video melalui jaringan lokal.

> **Note**
>
> Untuk membeli AWS Panorama Appliance, kunjungi AWS Panorama konsol.

Aplikasi AWS Panorama sampel menunjukkan penggunaan AWS Panorama fitur. Ini mencakup model yang telah dilatih dengan SageMaker AI dan kode sampel yang menggunakan SDK AWS Panorama Aplikasi untuk menjalankan inferensi dan output video. Contoh aplikasi menyertakan CloudFormation template dan skrip yang menunjukkan cara mengotomatiskan alur kerja pengembangan dan penyebaran dari baris perintah.

Dua topik terakhir dalam bagian ini merinci persyaratan untuk model dan kamera, dan spesifikasi perangkat keras AWS Panorama Alat. 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

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 menerapkan 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 perangkat tepi yang ditenagai oleh Nvidia Jetson AGX Xavier. Alih-alih mengirim gambar ke AWS Cloud 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 Mengelola AWS Panorama Appliance.

Perangkat yang kompatibel

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

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

- Lenovo ThinkEdge® SE7 0 — 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 Mengelola AWS Panorama aplikasi.

Simpul

Sebuah aplikasi terdiri dari beberapa komponen yang disebut node, 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 dalam paket node yang Anda unggah ke jalur akses Amazon S3, tempat AWS Panorama Appliance dapat mengaksesnya. Manifes aplikasi adalah 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 AI atau di lingkungan pengembangan Anda. Untuk informasi selengkapnya, lihat ???.

Menyiapkan AWS Panorama Appliance

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 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.

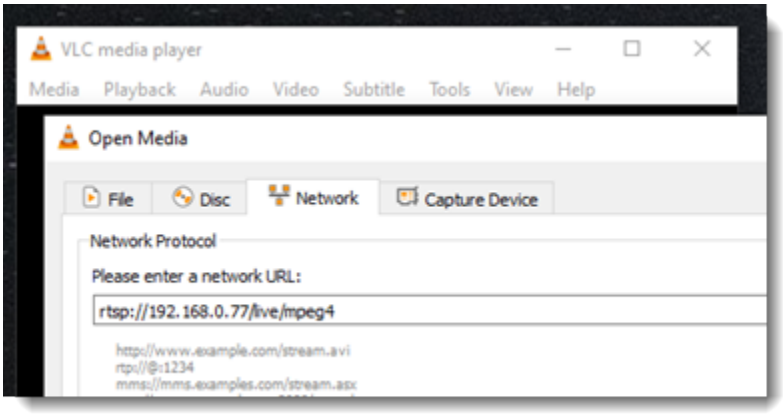
Bagian-bagian

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 yang FAT32 diformat dengan penyimpanan minimal 1 GB, untuk mentransfer arsip dengan file konfigurasi dan sertifikat ke AWS Panorama Appliance.
- Kamera — Kamera IP yang menghasilkan 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 streaming, dengan membukanya sebagai sumber media jaringan:



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

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

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

Daftarkan dan konfigurasikan AWS Panorama Appliance

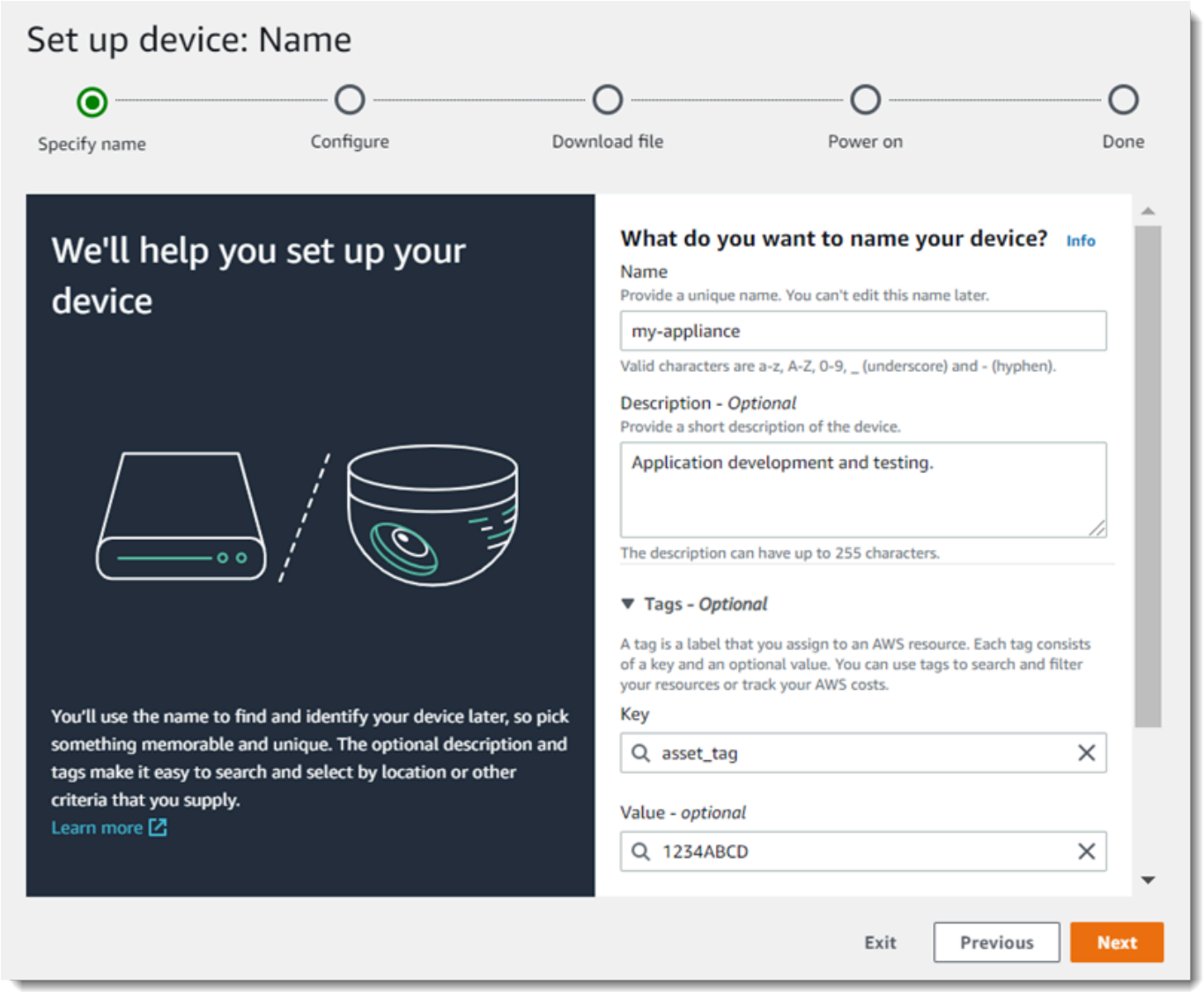
AWS Panorama Appliance adalah perangkat keras yang terhubung ke kamera berkemampuan jaringan melalui koneksi jaringan lokal. Ini menggunakan sistem operasi berbasis Linux yang mencakup AWS Panorama Application SDK dan perangkat lunak pendukung untuk menjalankan aplikasi visi komputer.

Untuk terhubung ke AWS manajemen alat dan penerapan aplikasi, alat menggunakan sertifikat perangkat. Anda menggunakan konsol AWS Panorama untuk membuat sertifikat penyediaan. Alat menggunakan sertifikat sementara ini untuk menyelesaikan pengaturan 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 dari awal.

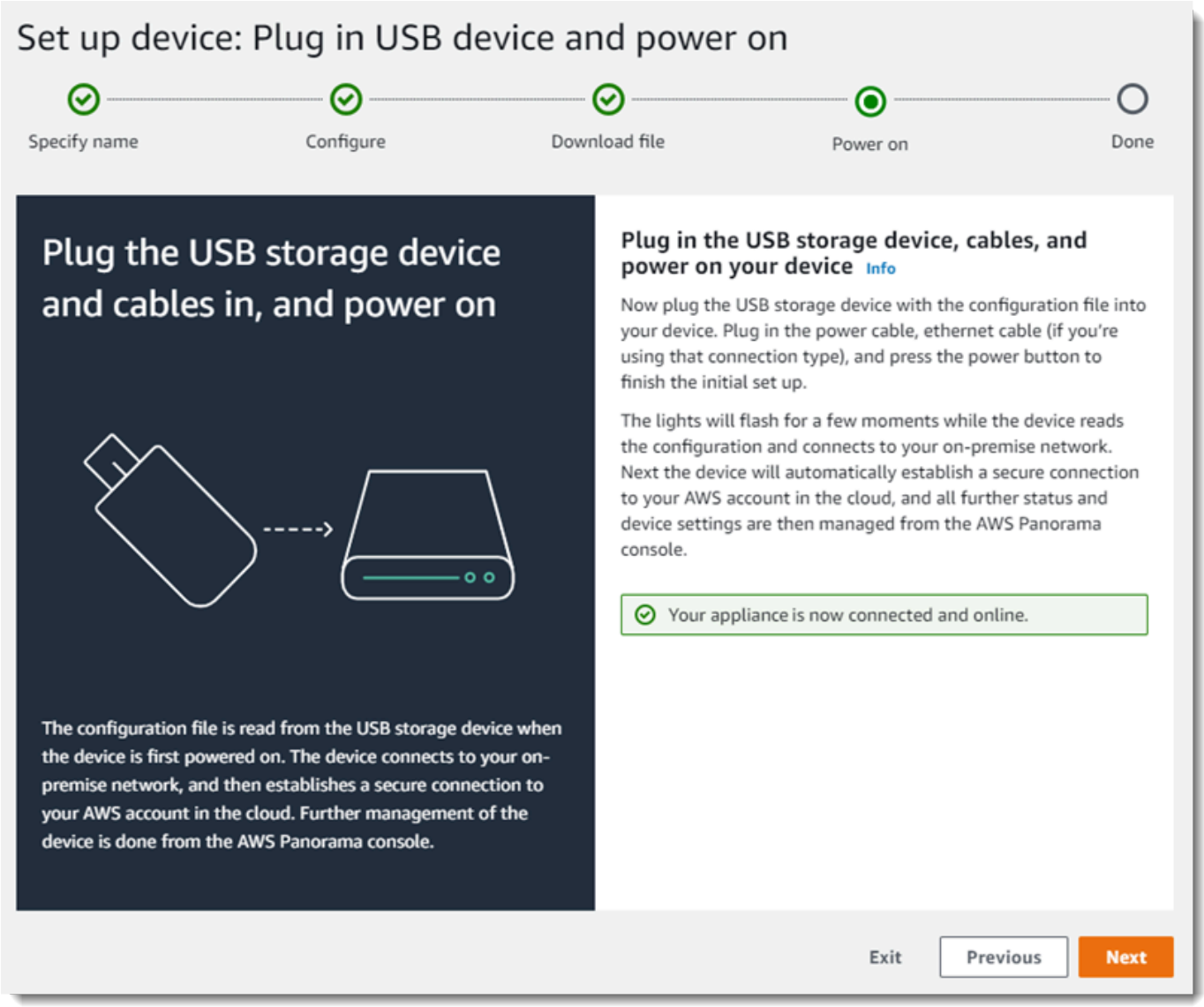
Untuk mendaftarkan alat

1. Hubungkan drive USB ke komputer Anda. Siapkan alat dengan menghubungkan jaringan dan kabel daya. Alat menyala dan menunggu drive USB terhubung.
2. Buka halaman Memulai konsol AWS Panorama.
3. Pilih Tambah perangkat.
4. Pilih Mulai penyiapan.
5. Masukkan nama dan deskripsi untuk sumber daya perangkat yang mewakili alat di AWS Panorama. Pilih Berikutnya



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 Berikutnya.
8. Salin arsip konfigurasi ke direktori root drive USB.
9. Hubungkan drive USB ke port USB 3.0 di bagian depan alat, di sebelah port HDMI.

   Saat Anda menghubungkan drive USB, alat 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. Pilih Selesai.

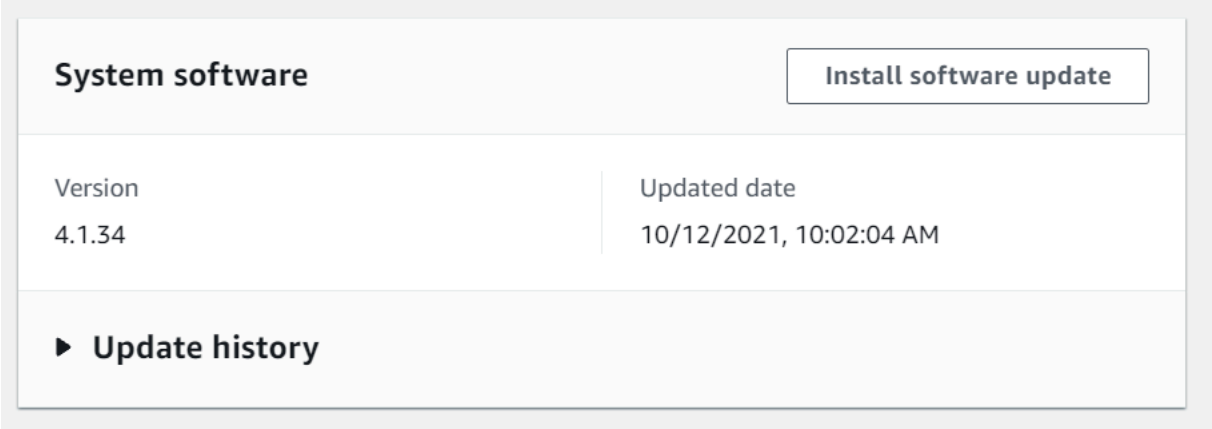
Tingkatkan perangkat lunak alat

AWS Panorama Appliance memiliki beberapa komponen perangkat lunak, termasuk sistem operasi Linux, SDK aplikasi AWS Panorama, dan pustaka dan kerangka kerja visi komputer pendukung. Untuk memastikan bahwa Anda dapat menggunakan fitur dan aplikasi terbaru dengan alat Anda, tingkatkan perangkat lunaknya setelah penyiapan 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.

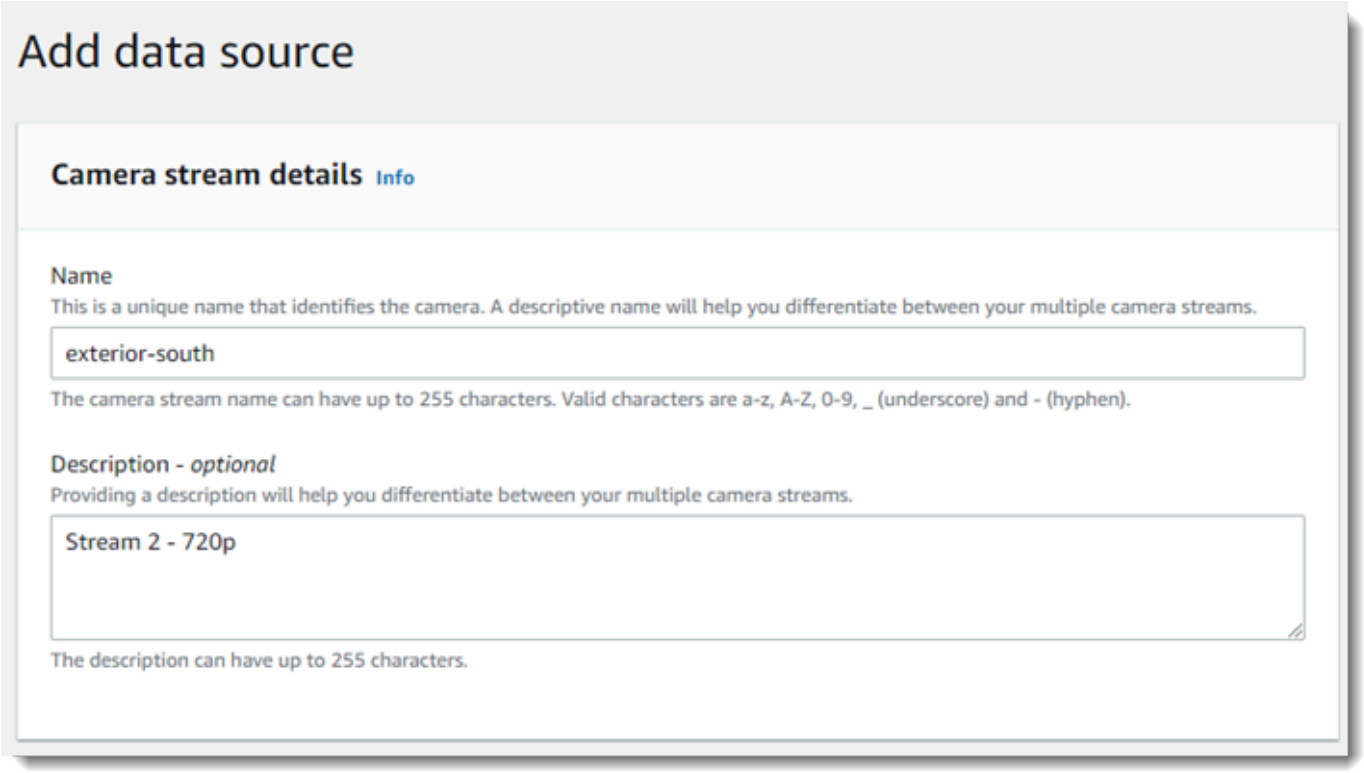
Proses upgrade bisa memakan waktu 30 menit atau lebih. Anda dapat memantau kemajuannya di konsol AWS Panorama atau pada monitor yang terhubung. Saat proses selesai, alat reboot.

Tambahkan aliran kamera

Selanjutnya, daftarkan aliran kamera dengan konsol AWS Panorama.

Untuk mendaftarkan aliran kamera

1. Buka halaman Sumber data AWS Panorama console.
2. Pilih Tambahkan sumber data.



3. Konfigurasikan pengaturan berikut.

   - Nama — Nama untuk aliran kamera.
   - Deskripsi — Deskripsi singkat tentang kamera, lokasinya, atau detail lainnya.
   - URL RTSP — URL yang menentukan alamat IP kamera dan jalur ke aliran. Sebagai contoh, `rtsp://192.168.0.77/live/mpeg4/`.
   - Kredensial — Jika aliran kamera dilindungi kata sandi, tentukan nama pengguna dan kata sandi.
4. Pilih 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 selama penyiapan, lihatPemecahan Masalah.

Untuk menerapkan aplikasi sampel, lanjutkan ke topik berikutnya.

Menerapkan aplikasi sampel AWS Panorama

Setelah menyiapkan AWS Panorama Appliance atau perangkat yang kompatibel dan memutakhirkan perangkat lunaknya, terapkan aplikasi contoh. 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 layar 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-bagian

Prasyarat

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

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

Untuk perintah panjang, kita menggunakan karakter escape (\) untuk membagi perintah di 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 menggunakan AWS Command Line Interface (AWS CLI) untuk memanggil operasi API layanan. Jika Anda sudah memilikinya AWS CLI, tingkatkan ke versi terbaru. Untuk menginstal CLI Aplikasi AWS Panorama dan, gunakan. AWS CLI`pip`

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

Unduh aplikasi sampel, dan ekstrak ke ruang kerja Anda.

- Contoh aplikasi — aws-panorama-sample.zip

Impor aplikasi sampel

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 memperbarui ini dengan nomor akun Anda, jalankan `panorama-cli import-application` perintah.

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

`SAMPLE_CODE`Paket, dalam `packages` direktori, berisi kode dan konfigurasi aplikasi, 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 jalur 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 alat Anda.

Untuk menyebarkan aplikasi

1. Buka halaman Aplikasi Penerapan konsol AWS Panorama.
2. Pilih Menyebarkan aplikasi.
3. Tempelkan konten manifes aplikasi`graphs/aws-panorama-sample/graph.json`,, ke editor teks. Pilih Berikutnya.
4. Untuk Application name (Nama aplikasi), masukkan `aws-panorama-sample`.
5. Pilih Lanjutkan untuk menyebarkan.
6. Pilih Mulai penerapan.
7. Pilih Berikutnya tanpa memilih peran.
8. Pilih Pilih perangkat, lalu pilih alat Anda. Pilih Berikutnya.
9. Pada langkah Pilih sumber data, pilih Lihat input, dan tambahkan aliran kamera Anda sebagai sumber data. Pilih Berikutnya.

10. Pada langkah Konfigurasi, pilih Berikutnya.
11. Pilih Deploy, lalu pilih Selesai.
12. Dalam daftar aplikasi yang digunakan, pilih aws-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, periksa log aplikasi dan perangkat di Amazon CloudWatch Logs.

Lihat outputnya

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

Untuk melihat log di CloudWatch Log

1. Buka halaman Grup log dari konsol CloudWatch 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 objek umum. Coba arahkan kamera Anda ke keyboard atau cangkir kopi.

Untuk kesederhanaan, aplikasi sampel menggunakan model klasifikasi ringan. Model menghasilkan array tunggal dengan probabilitas untuk masing-masing kelasnya. 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 untuk Python

Aplikasi sampel menggunakan AWS SDK for Python (Boto) untuk mengirim metrik ke Amazon CloudWatch. Untuk mengaktifkan fungsionalitas ini, buat peran yang memberikan izin aplikasi untuk mengirim metrik, dan menerapkan ulang aplikasi dengan peran yang dilampirkan.

Contoh aplikasi menyertakan CloudFormation template yang membuat 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 menerapkan kembali aplikasi

1. Buka halaman Aplikasi Penerapan konsol AWS Panorama.
2. Pilih aplikasi.
3. Pilih Ganti.
4. Selesaikan langkah-langkah untuk menyebarkan aplikasi. Dalam peran Tentukan IAM, pilih peran yang Anda buat. Namanya dimulai dengan `aws-panorama-sample-runtime`
5. Saat penerapan selesai, buka CloudWatchkonsol dan lihat metrik di namespace. `AWSPanoramaApplication` 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 halaman Aplikasi Penerapan konsol AWS Panorama.
2. Pilih aplikasi.
3. Pilih Hapus dari perangkat.

Langkah selanjutnya

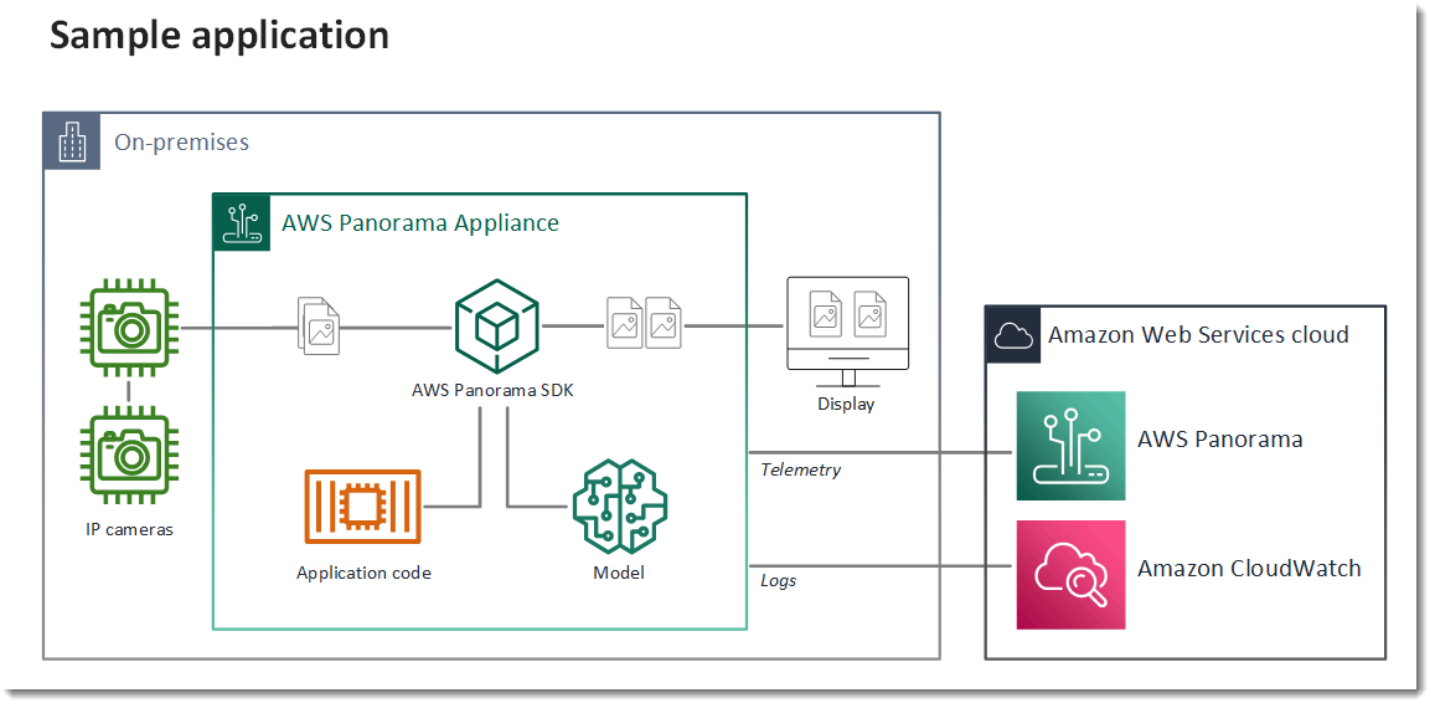
Jika Anda mengalami kesalahan saat menerapkan atau menjalankan aplikasi sampel, lihatPemecahan Masalah.

Untuk mempelajari lebih lanjut tentang fitur dan implementasi aplikasi sampel, lanjutkan ke topik berikutnya.

Mengembangkan aplikasi AWS Panorama

Anda dapat menggunakan contoh aplikasi 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 frame video dari kamera, memproses data video, dan mengirim data ke model visi komputer yang mendeteksi objek. Aplikasi menampilkan hasilnya pada layar HDMI yang terhubung ke alat.

Bagian-bagian

- Manifes aplikasi
- Membangun dengan aplikasi sampel
- Mengubah model visi komputer
- Preprocessing gambar
- Mengunggah metrik dengan SDK untuk Python

- Langkah selanjutnya

Manifes aplikasi

Manifes aplikasi adalah file bernama `graph.json` 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

```
        "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 kamera generik dan paket tampilan yang disediakan oleh layanan AWS Panorama. `abstract_rtsp_media_source`Paket ini merupakan placeholder untuk kamera yang Anda timpa selama penerapan. `hdmi_data_sink`Paket ini mewakili konektor output HDMI pada perangkat.

Node adalah antarmuka ke paket, serta parameter non-paket yang dapat memiliki nilai default yang Anda timpa pada waktu penerapan. Paket kode dan model mendefinisikan antarmuka dalam `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 tersebut adalah logika bisnis dan dibutuhkan aliran video bernama `video_in` dan nomor floating point bernama `threshold` input. Antarmuka juga menentukan bahwa kode tersebut memerlukan buffer aliran video yang diberi nama `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 dalam manifes aplikasi, `camera_node` node mewakili aliran video dari kamera. Ini termasuk dekorator yang muncul di konsol saat Anda menyebarkan aplikasi, meminta Anda untuk memilih aliran kamera.

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

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

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

```
        "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"
            }
        ]
```

Membangun dengan aplikasi sampel

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

Nama setiap paket harus unik di akun Anda. Jika Anda dan pengguna lain di akun Anda sama-sama menggunakan nama paket generik seperti `code` ataumodel, Anda mungkin mendapatkan versi paket yang salah saat menerapkan. 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 - `graphs/aws-panorama-sample/graph.json`
   - Konfigurasi Package - `packages/123456789012-SAMPLE_CODE-1.0/package.json`
   - Membangun skrip - `3-build-container.sh`

Untuk memperbarui kode aplikasi

1. Ubah 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 halaman Aplikasi Penerapan konsol AWS Panorama.
5. Pilih aplikasi.
6. Pilih Ganti.

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

Mengubah model visi komputer

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

Contoh berikut menggunakan model MXNet SSD ResNet 50 yang dapat Anda unduh dari GitHub repo panduan ini: ssd_512_resnet50_v1_voc.tar.gz

Untuk mengubah model aplikasi sampel

1. Ganti 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 - `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`). Ubah `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 agar sesuai dengan model Anda.

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

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

Nilai bentuk,`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 dan urutan array bervariasi di antara model.

5. Impor model dengan AWS Panorama Application CLI. CLI Aplikasi AWS Panorama menyalin file model dan deskriptor ke dalam `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 dalam `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)
```

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

Preprocessing gambar

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

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 standar deviasi. Akhirnya, ia menggabungkan saluran warna dan mengubahnya menjadi NumPy array yang dapat diproses model.

Example application.py - Preprocessing

```
    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 yang dapat diprediksi yang berpusat di sekitar 0. Ini cocok dengan preprocessing yang diterapkan pada gambar dalam dataset pelatihan, yang merupakan pendekatan standar tetapi dapat bervariasi per model.

Mengunggah metrik dengan SDK untuk Python

Aplikasi sampel menggunakan SDK untuk Python untuk mengunggah metrik ke Amazon. CloudWatch

Example application.py - SDK untuk 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 penerapan. Peran didefinisikan dalam `aws-panorama-sample.yml` CloudFormation template.

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 menggunakan AWS SDK dalam kode aplikasi Anda, pertama-tama ubah template untuk menambahkan izin untuk semua tindakan API yang digunakan aplikasi. Perbarui CloudFormation tumpukan dengan menjalankan `1-create-role.sh` setiap kali Anda membuat perubahan. Kemudian, terapkan perubahan pada kode aplikasi Anda.

Untuk tindakan yang mengubah atau menggunakan sumber daya yang ada, merupakan praktik terbaik untuk meminimalkan ruang lingkup 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, lihat Kunci tindakan, sumber daya, dan kondisi di 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/aws/aws-panorama-cli

Untuk kode sampel 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-sampel

Model dan kamera visi komputer yang didukung

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

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

Bagian-bagian

Model yang didukung

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

AWS Panorama mendukung versi PyTorch, Apache MXNet, dan TensorFlow yang didukung untuk perangkat edge oleh SageMaker AI Neo. Saat Anda membuat model sendiri, Anda dapat menggunakan versi kerangka kerja yang tercantum dalam catatan rilis SageMaker AI Neo. Di SageMaker AI, Anda dapat menggunakan algoritma klasifikasi gambar bawaan.

Untuk informasi selengkapnya tentang penggunaan model di AWS Panorama, lihat. Model visi komputer

Kamera yang didukung

AWS Panorama Appliance mendukung aliran video H.264 dari kamera yang mengeluarkan RTSP melalui jaringan lokal. Untuk aliran kamera yang lebih besar dari 2 megapiksel, alat ini menurunkan 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- 040W PB3
- Vivotek — 0-H IB936
- Amcrest — M-841B IP2
- Aplikasi — IPC-B850W-S-3X, IPC-D250W-S
- WGCC — Kubah PoE 4MP ONVIF

Untuk spesifikasi perangkat keras alat, lihatSpesifikasi AWS Panorama Appliance.

Spesifikasi AWS Panorama Appliance

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

Komponen	Spesifikasi
Prosesor dan GPU	Nvidia Jetson AGX Xavier dengan RAM 32GB
eternet	2x 1000 Base-T (Gigabyte)
USB	1x USB 2.0 dan 1x USB 3.0 tipe-A wanita
Keluaran 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)
Perlindungan debu dan cairan	IP-62
Kepatuhan peraturan EMI/EMC	FCC Bagian-15 (AS)
Batas sentuhan termal	IEC-62368
Suhu operasi	-20° C hingga 60° C
Kelembaban operasi	0% hingga 95% RH
Suhu penyimpanan	-20° C hingga 85° C
Kelembaban penyimpanan	Tidak terkontrol untuk suhu rendah. 90% RH pada suhu tinggi
Pendinginan	Ekstraksi panas udara paksa (kipas)
Opsi pemasangan	Rackmount atau berdiri bebas

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

Penyimpanan Wi-Fi, Bluetooth, dan kartu SD ada di alat tetapi tidak dapat digunakan.

AWS Panorama Appliance mencakup dua sekrup untuk dipasang di rak server. Anda dapat memasang 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 di beberapa AWS Wilayah, 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.
- Instans 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, atau AWS SDK 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, kunjungi konsol Service Quotas, atau lihat titik akhir dan kuota AWS Panorama di. Referensi Umum Amazon Web

AWS Panorama izin

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

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

AWS Panorama Appliance juga memiliki peran yang memberinya izin untuk mengakses AWS layanan dan sumber daya. Peran alat adalah salah satu peran layanan yang digunakan AWS Panorama layanan untuk mengakses layanan lain atas nama Anda.

Peran aplikasi adalah peran layanan terpisah yang Anda buat untuk aplikasi, untuk memberikan izin untuk menggunakan AWS layanan dengan aplikasi AWS 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) aplikasi yang mengharuskan pengguna untuk memasukkan nama pengguna mereka dalam nama aplikasi yang mereka buat. Untuk sumber daya dan kondisi yang didukung oleh setiap tindakan, lihat Kunci tindakan, sumber daya, dan kondisi AWS Panorama di Referensi Otorisasi Layanan.

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

Topik

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

Kebijakan IAM berbasis identitas untuk AWS Panorama

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

AWS Panorama menyediakan kebijakan terkelola yang memberikan akses ke tindakan AWS Panorama API 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 saat dirilis.

- AWSPanoramaFullAccess— Menyediakan akses penuh ke AWS Panorama, jalur akses AWS Panorama di Amazon S3, kredensi alat di, dan log alat di AWS Secrets Manager 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 mencakup izin untuk membeli perangkat dari konsol AWS Panorama. Untuk memberikan izin ini, tambahkan kebijakan berikut.

- ElementalAppliancesSoftwareFullAccess— Lihat kebijakan

Kebijakan terkelola memberikan izin untuk tindakan API tanpa membatasi sumber daya yang dapat dimodifikasi 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 untuk kebijakan Anda.

Membuat peran layanan

Saat pertama kali 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 ketentuan yang dapat Anda gunakan untuk membatasi cakupan izin pengguna di AWS Panorama, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS Panorama di Referensi Otorisasi Layanan.

Peran layanan AWS Panorama dan sumber daya lintas layanan

AWS Panorama menggunakan layanan AWS lainnya 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 Anda 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 ke 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, gunakan konsol IAM. Jika memungkinkan, izin peran dibatasi untuk sumber daya yang cocok dengan pola penamaan yang digunakan AWS Panorama. Misalnya, hanya `AWSServiceRoleForAWSPanorama` memberikan izin untuk layanan untuk mengakses AWS IoT sumber daya yang memiliki `panorama` nama mereka.

Bagian-bagian

- Mengamankan peran alat
- Penggunaan layanan lain

Mengamankan peran alat

AWS Panorama Appliance menggunakan `AWSPanoramaApplianceServiceRole` peran tersebut untuk mengakses sumber daya di akun Anda. Alat ini memiliki izin untuk mengunggah CloudWatch log ke Log, membaca kredensil aliran kamera dari AWS Secrets Manager, dan 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 menggunakan AWS layanan, buat peran 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. Buka peran alat di konsol IAM: AWSPanoramaApplianceServiceRole
2. Pilih Edit trust relationship (Edit Hubungan Kepercayaan).
3. Perbarui konten kebijakan, lalu pilih Perbarui kebijakan kepercayaan.

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

Example kebijakan kepercayaan — Akun khusus

JSON

```
{
  "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 lanjut, dan membiarkannya hanya mengambil peran dengan perangkat tertentu, Anda dapat menentukan perangkat dengan ARN. `aws:SourceArn`Kondisi ini membandingkan ARN alat yang ditentukan oleh AWS Panorama dengan ARN yang Anda sertakan dalam kebijakan.

Example kebijakan kepercayaan - Alat tunggal

JSON

```
{
  "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 kembali alat, Anda harus menghapus kondisi ARN sumber sementara dan kemudian menambahkannya lagi dengan ID perangkat baru.

Untuk informasi selengkapnya tentang kondisi ini, dan praktik terbaik keamanan saat layanan menggunakan peran untuk mengakses sumber daya di akun Anda, lihat Masalah deputi yang membingungkan di 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 mementaskan 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 di Panduan Pengguna IAM yang ditautkan dalam daftar ini.

Memberikan izin untuk aplikasi

Anda dapat membuat peran untuk aplikasi Anda untuk memberikan izin untuk memanggil AWS layanan. Secara default, aplikasi tidak memiliki izin apa pun. Anda membuat peran aplikasi di IAM dan menetapkannya ke aplikasi selama penerapan. Untuk memberikan aplikasi Anda hanya izin yang dibutuhkan, buat peran untuk itu dengan izin untuk tindakan API tertentu.

Contoh aplikasi mencakup CloudFormation template dan skrip yang membuat peran aplikasi. Ini adalah peran layanan yang dapat diasumsikan oleh AWS Panorama. Peran ini memberikan izin bagi aplikasi untuk menelepon untuk CloudWatch 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 atau pola API untuk nilai. `Action`

Untuk informasi selengkapnya tentang izin di AWS Panorama, lihat. AWS Panorama izin

Mengelola AWS Panorama Appliance

AWS Panorama Appliance adalah perangkat keras yang menjalankan aplikasi Anda. Anda menggunakan AWS Panorama konsol 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 layar terlampir.

Setelah menyiapkan alat Anda atau perangkat lain yang kompatibel, Anda mendaftarkan kamera untuk digunakan dengan aplikasi. Anda mengelola aliran kamera di AWS Panorama konsol. Saat Anda menerapkan aplikasi, Anda memilih aliran kamera mana yang dikirim alat untuk diproses.

Untuk tutorial yang memperkenalkan AWS Panorama Appliance dengan contoh aplikasi, lihatMemulai dengan AWS Panorama.

Topik

Mengelola Alat Panorama AWS

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

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

Untuk mendaftarkan alat dengan AWS Panorama API, lihat. Otomatiskan pendaftaran perangkat

Bagian-bagian

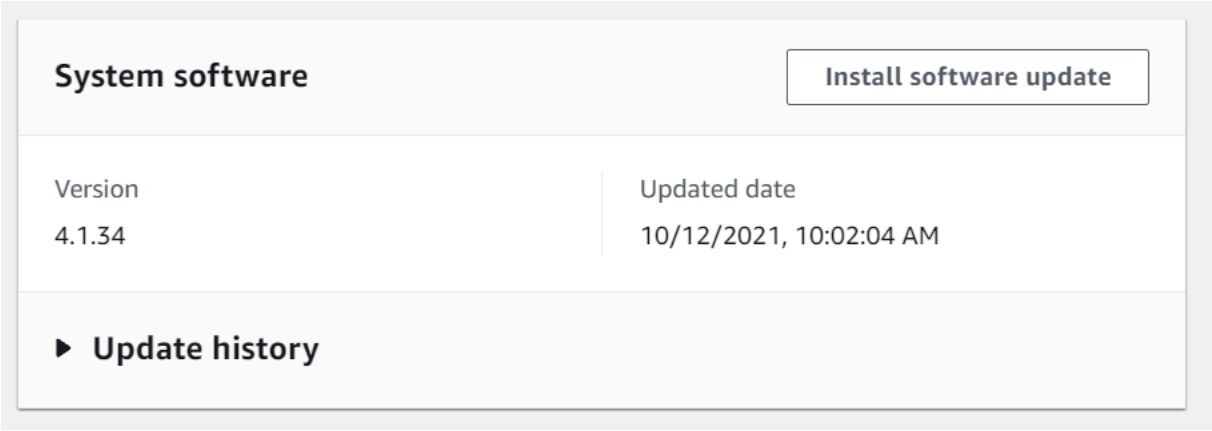
- Perbarui perangkat lunak alat
- Deregister alat
- Nyalakan ulang alat
- Setel ulang alat

Perbarui 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.

Deregister alat

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

Untuk menghapus alat

1. Buka halaman Perangkat konsol AWS Panorama.
2. Pilih nama alat.
3. Pilih 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 dideregistrasi tidak dapat terhubung ke AWS layanan dan tidak dapat didaftarkan lagi hingga disetel ulang.

Nyalakan ulang alat

Anda dapat me-reboot alat dari jarak jauh.

Untuk me-reboot alat

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

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

Setel ulang alat

Untuk menggunakan alat di Wilayah yang berbeda atau dengan akun yang berbeda, 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 daya 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 atur ulang.

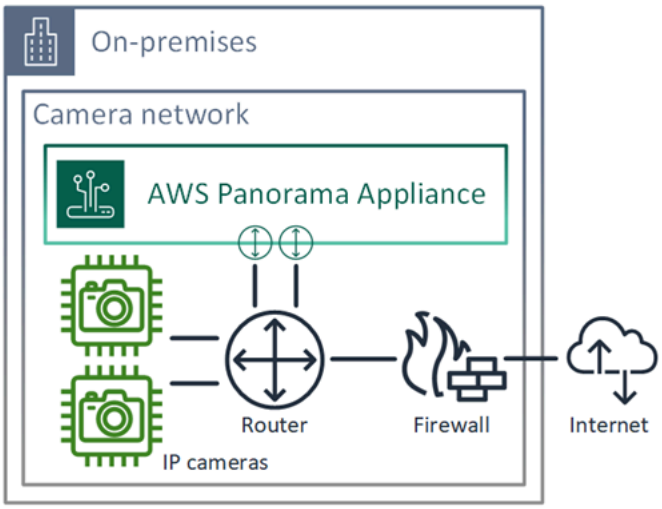
Menghubungkan AWS Panorama Appliance ke jaringan Anda

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

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.

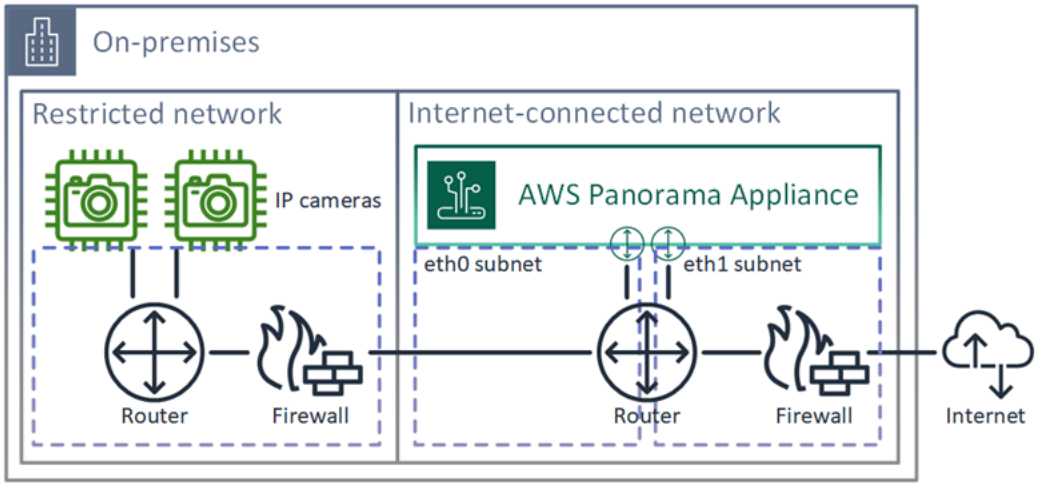


Untuk detail tentang port dan titik akhir yang perlu diakses oleh alat, lihat Mengkonfigurasi akses layanan danMengkonfigurasi akses jaringan lokal.

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 IoT kredensi (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 atau Direct Connect 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 Service

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

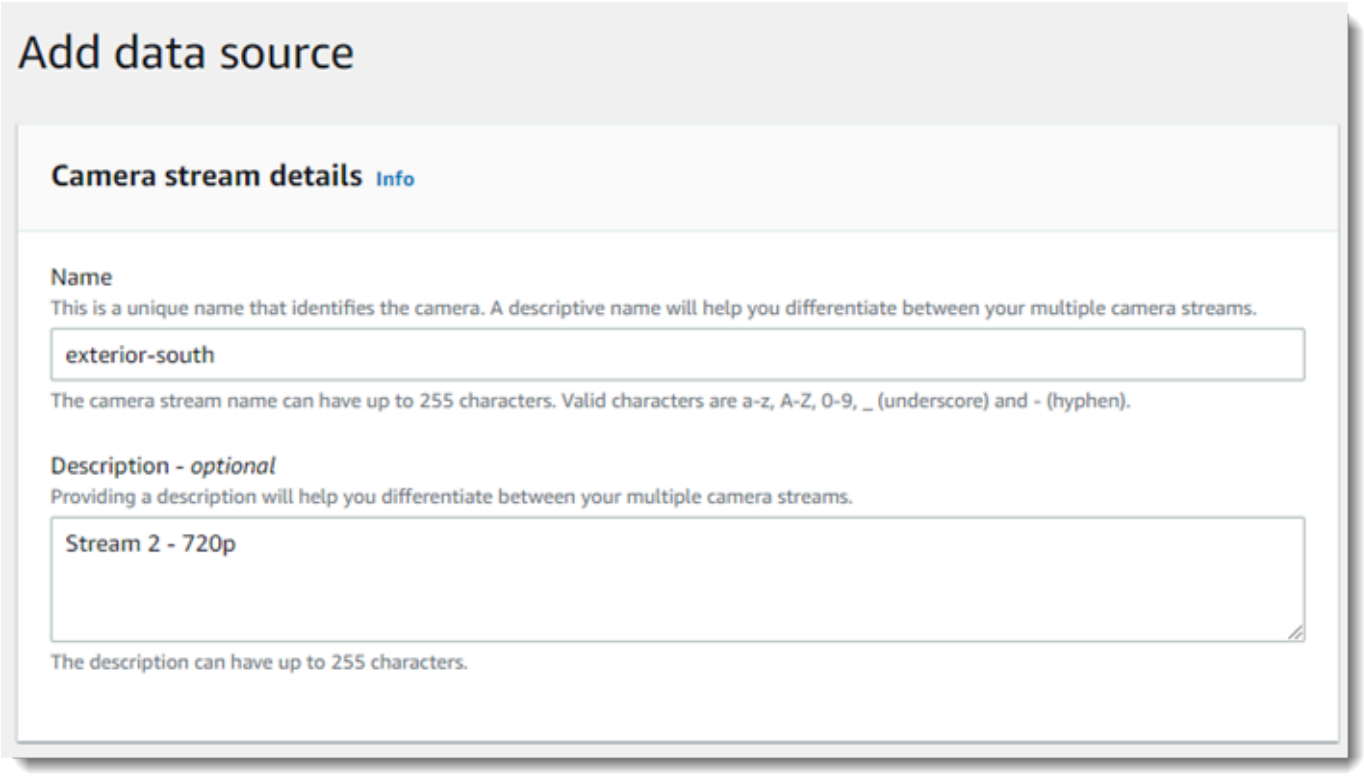
Mengelola aliran kamera di AWS Panorama

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**
>
> Aplikasi dapat terhubung ke aliran kamera apa pun yang dapat dirutekan dari jaringan lokal yang terhubung dengannya. 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 halaman Sumber data AWS Panorama console.
2. Pilih Tambahkan sumber data.



3. Konfigurasikan pengaturan berikut.

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

4. Pilih Simpan.

Untuk mendaftarkan aliran kamera dengan AWS Panorama API, lihat. Otomatiskan pendaftaran perangkat

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

Menghapus aliran

Anda dapat menghapus aliran kamera di konsol AWS Panorama.

Untuk menghapus aliran kamera

1. Buka halaman Sumber data AWS Panorama console.
2. Pilih aliran kamera.
3. Pilih Hapus sumber data.

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

Mengelola aplikasi di AWS Panorama Appliance

Aplikasi adalah kombinasi dari kode, model, dan konfigurasi. Dari halaman Perangkat di konsol AWS Panorama, Anda dapat mengelola aplikasi di alat.

Untuk mengelola aplikasi di AWS Panorama Appliance

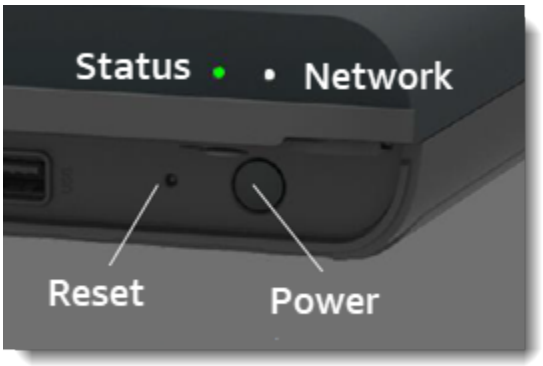
1. Buka halaman Perangkat konsol AWS Panorama.
2. Pilih alat.

Halaman Aplikasi Deployed menunjukkan aplikasi yang telah digunakan ke alat.

Gunakan opsi di halaman ini untuk menghapus aplikasi yang digunakan 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

LEDs Perubahan warna dan kedipan untuk menunjukkan status. Kedipan lambat adalah setiap tiga detik sekali. Sebuah kedipan cepat sekali per detik.

Status status LED

- Hijau berkedip cepat — Alat sedang boot.
- Hijau pekat — Alat beroperasi secara normal.
- Biru berkedip lambat — Alat ini menyalin file konfigurasi dan mencoba mendaftar. AWS IoT
- Biru berkedip cepat - Alat ini menyalin gambar log ke drive USB.
- Merah berkedip 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.

Lampu jaringan

LED jaringan memiliki status berikut:

Status LED jaringan

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

- Merah solid — Kabel Ethernet tidak terhubung.

Tombol daya dan atur ulang

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 membutuhkan 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 membutuhkan banyak tombol, tekan dan tahan kedua tombol secara bersamaan.

   - Reset penuh - Tekan lama daya dan reset.

     Mengembalikan versi perangkat lunak minimum dan menghapus semua file konfigurasi dan aplikasi.
   - Kembalikan versi perangkat lunak terbaru — Reset tekan sebentar.

     Menerapkan kembali pembaruan perangkat lunak terbaru ke alat.
   - Kembalikan versi perangkat lunak minimum - Reset tekan lama.

     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 siap, lampu status berkedip hijau.

Menyetel ulang alat tidak menghapusnya dari layanan AWS Panorama. Untuk informasi selengkapnya, lihat Deregister alat.

Mengelola AWS Panorama 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.

Topik

Menyebarkan aplikasi

Untuk menerapkan aplikasi, Anda menggunakan AWS Panorama Application CLI mengimpornya ke akun Anda, membangun wadah, 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 penelusurannya.

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

Bagian-bagian

- Instal CLI Aplikasi AWS Panorama
- Impor 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 CLI Aplikasi AWS Panorama dan AWS 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, lihatMenyiapkan lingkungan pengembangan di Windows.

Impor aplikasi

Jika Anda bekerja dengan contoh aplikasi 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 Package dimulai dengan ID akun akun tempat mereka digunakan. Saat Anda menyebarkan 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 kredenal 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 di 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` dengan AWS 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 image 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 gambar sistem file.

```
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 gambar Docker bernama `code_asset` dan mengekspor sistem file ke arsip di folder. `.tar.gz` `assets` CLI menarik image dasar aplikasi dari Amazon Elastic Container Registry (Amazon ECR) Registry ECR), seperti yang ditentukan dalam Dockerfile aplikasi.

Selain arsip kontainer, CLI membuat 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. Nama-nama ini digunakan oleh alat selama proses penyebaran.

Example packages/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 sampel.

Example skrip util/ .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 di direktori paket model tidak diterapkan sampai Anda mengimpornya kembali 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 sistem file kontainer, dan file deskriptornya, 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 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 jalur akses Amazon S3 yang khusus untuk akun Anda. AWS Panorama mengelola jalur 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 Aplikasi Penerapan konsol AWS Panorama.
2. Pilih Menyebarkan aplikasi.
3. Tempelkan konten manifes aplikasi`graph.json`,, ke editor teks. Pilih Berikutnya.
4. Masukkan nama dan descroption.
5. Pilih Lanjutkan untuk menyebarkan.
6. Pilih Mulai penerapan.
7. Jika aplikasi Anda menggunakan peran, pilih dari menu tarik-turun. Pilih Berikutnya.
8. Pilih Pilih perangkat, lalu pilih alat Anda. Pilih Berikutnya.

9. Pada langkah Pilih sumber data, pilih Lihat input, dan tambahkan aliran kamera Anda sebagai sumber data. Pilih Berikutnya.
10. Pada langkah Konfigurasi, konfigurasikan pengaturan khusus aplikasi apa pun yang ditentukan oleh pengembang. Pilih Berikutnya.
11. Pilih Deploy, lalu pilih Selesai.
12. Dalam daftar aplikasi yang digunakan, pilih aplikasi untuk memantau statusnya.

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

Mengotomatiskan penerapan aplikasi

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

Untuk informasi selengkapnya, 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-bagian

Perbarui atau salin aplikasi

Untuk memperbarui aplikasi, gunakan opsi Ganti. Saat Anda mengganti aplikasi, Anda dapat memperbarui kode atau modelnya.

Untuk memperbarui aplikasi

1. Buka halaman Aplikasi Penerapan konsol AWS Panorama.
2. Pilih aplikasi.
3. Pilih Ganti.
4. Ikuti petunjuk untuk membuat versi atau aplikasi baru.

Ada juga opsi Klon yang bertindak mirip dengan Ganti, tetapi tidak menghapus versi lama aplikasi. Anda dapat menggunakan opsi ini untuk menguji perubahan pada aplikasi tanpa menghentikan versi yang sedang berjalan, atau untuk menerapkan ulang versi yang telah Anda hapus.

Hapus versi dan aplikasi

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

Untuk menghapus aplikasi

1. Buka halaman Aplikasi Penerapan konsol AWS Panorama.
2. Pilih aplikasi.
3. Pilih Hapus dari perangkat.

Konfigurasi Package

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 termasuk file biner (gambar dan model kontainer) dan file deskriptor, yang diunduh oleh AWS Panorama Appliance selama penerapan. Untuk mendaftarkan aset paket, Anda menyediakan file konfigurasi paket terpisah yang mendefinisikan paket, asetnya, dan antarmukanya.

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 yang 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"
                    }
                ]
            }
        ]
    }
}
```

`assets`Bagian ini menentukan nama artefak yang diunggah oleh AWS Panorama Application CLI ke Amazon S3. Jika Anda mengimpor contoh aplikasi atau aplikasi dari pengguna lain, bagian ini bisa kosong atau merujuk ke aset yang tidak ada di akun Anda. Saat 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 graphs/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 manifes aplikasi dan dokumen override didefinisikan dalam skema JSON. Anda dapat menggunakan skema JSON untuk memvalidasi dokumen konfigurasi Anda sebelum menerapkan. Skema JSON tersedia di repositori panduan ini. GitHub

- Skema JSON -/sumber daya aws-panorama-developer-guide

Node aplikasi

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

Dalam contoh berikut, `code_node` dan `model_node` lihat kode sampel dan paket model yang disertakan dengan aplikasi sampel. `camera_node`menggunakan paket yang disediakan oleh AWS Panorama untuk membuat placeholder untuk aliran 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

Edges memetakan output dari satu node ke input yang lain. Dalam contoh berikut, tepi pertama memetakan output dari node aliran kamera ke input node kode aplikasi. Nama-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"
            },
```

Dalam kode aplikasi Anda, Anda menggunakan `outputs` atribut `inputs` and untuk mendapatkan gambar dari aliran input, dan mengirim gambar ke aliran output.

Example application.py - Video masukan 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)
```

Node abstrak

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

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

  Nama Package - `panorama::abstract_rtsp_media_source`

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

  Nama Package - `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 input. Node output, yang merujukh`dmi_data_sink`, memberikan akses kode aplikasi ke buffer video yang merupakan output dari port HDMI alat.

Example graph.json - Node abstrak

```
{
    "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 penerapan. Parameter dapat memiliki nilai default dan dekorator, yang menginstruksikan pengguna aplikasi cara mengkonfigurasinya.

Jenis parameter

- `string`— Sebuah string. Misalnya, `DEBUG`.
- `int32`— Sebuah bilangan bulat. Sebagai contoh, `20`.
- `float32`— Nomor floating point. Sebagai contoh, `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 penerapan dengan penggantian. Untuk informasi selengkapnya, lihat Konfigurasi waktu penerapan dengan penggantian.

Konfigurasi waktu penerapan dengan penggantian

Anda mengonfigurasi parameter dan node abstrak selama penerapan. 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 overrides memiliki struktur yang mirip dengan manifes aplikasi. Untuk parameter dengan tipe dasar, Anda mendefinisikan 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 digantikannya.

Example menggantikan.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 penggantian untuk satu parameter string dan simpul kamera abstrak. Ini `nodeOverrides` memberi tahu AWS Panorama node mana dalam dokumen ini yang mengesampingkan yang dalam manifes aplikasi.

Membangun AWS Panorama 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.

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

Proses kode aplikasi masih gambar dari aliran kamera, mengirimkannya ke model, dan memproses hasilnya. Sebuah model dapat mendeteksi beberapa objek dan mengembalikan bentuk dan lokasinya. Kode dapat menggunakan informasi ini untuk menambahkan teks atau grafik ke video, atau untuk mengirim hasil ke AWS layanan untuk penyimpanan atau pemrosesan lebih lanjut.

Untuk mendapatkan gambar dari aliran, berinteraksi dengan model, dan video keluaran, kode aplikasi menggunakan SDK AWS Panorama Aplikasi. Aplikasi SDK adalah pustaka Python yang mendukung model yang dihasilkan PyTorch dengan, MXNet Apache, dan. TensorFlow

Topik

Model visi komputer

Model visi komputer adalah program perangkat lunak yang dilatih untuk mendeteksi objek dalam gambar. Sebuah model belajar mengenali satu set objek dengan terlebih dahulu menganalisis gambar objek tersebut melalui pelatihan. Model visi komputer mengambil gambar sebagai input dan output informasi tentang objek yang dideteksi, seperti jenis objek dan lokasinya. AWS Panorama mendukung model visi komputer yang dibangun dengan PyTorch, Apache MXNet, dan. TensorFlow

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

Bagian-bagian

Menggunakan model dalam kode

Sebuah model mengembalikan satu atau lebih hasil, yang dapat mencakup probabilitas untuk kelas terdeteksi, informasi lokasi, dan data lainnya.Contoh berikut menunjukkan bagaimana 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.

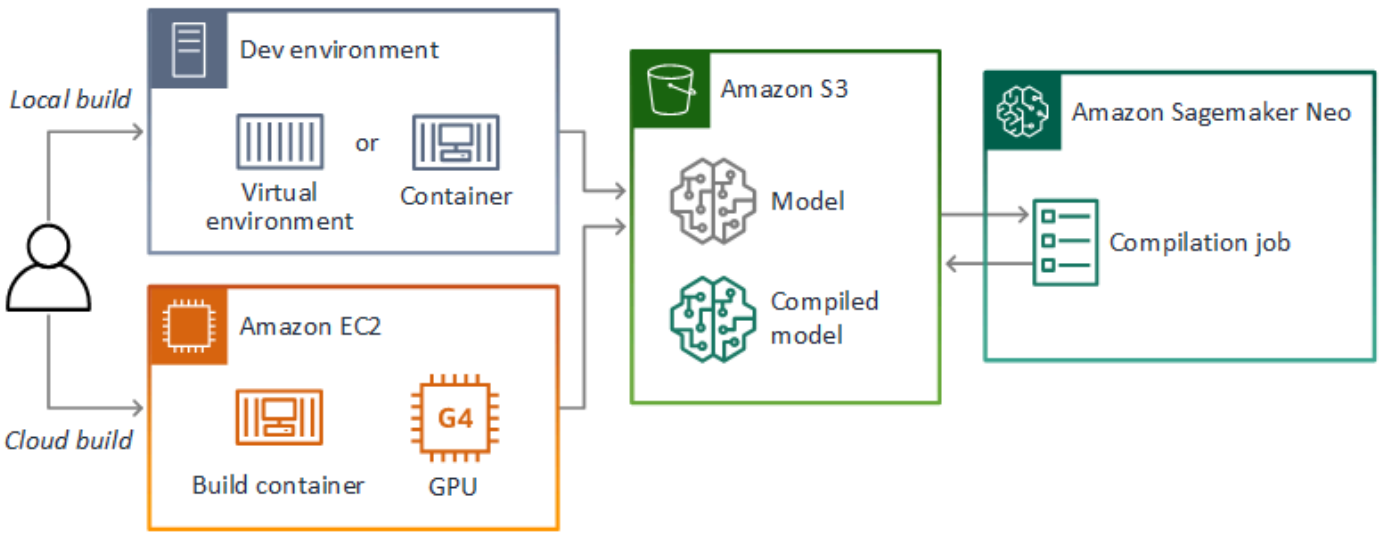
Membangun model khusus

Anda dapat menggunakan model yang Anda buat PyTorch, Apache MXNet, dan TensorFlow dalam aplikasi AWS Panorama. Sebagai alternatif untuk membangun dan melatih model dalam SageMaker AI, 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 AI Neo, lihat Kerangka Kerja yang Didukung di Panduan Pengembang Amazon SageMaker AI.

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

- Aplikasi sampel model kustom



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

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

Mengemas 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 mayor dan minor versi kerangka kerja saja. Untuk daftar versi yang didukung PyTorch, Apache MXNet, dan TensorFlow versi, lihat Kerangka kerja 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 dengan cahaya rendah atau cahaya latar.
- Resolusi — Ukuran input model biasanya ditetapkan pada resolusi antara 224 dan 512 piksel lebar dalam rasio aspek persegi. Sebelum Anda meneruskan bingkai video ke model, Anda dapat menurunkan atau memotongnya agar sesuai dengan ukuran yang diperlukan.
- Distorsi gambar — Panjang fokus kamera dan bentuk lensa dapat menyebabkan gambar menunjukkan distorsi jauh dari pusat bingkai. Posisi kamera juga menentukan fitur subjek mana yang terlihat. Misalnya, kamera overhead dengan lensa sudut lebar akan menunjukkan bagian atas subjek saat berada di tengah bingkai, dan tampilan miring dari 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 varians 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 berbeda.

Membangun gambar aplikasi

AWS Panorama Appliance menjalankan aplikasi sebagai sistem file kontainer yang diekspor dari gambar yang Anda buat. Anda menentukan dependensi dan resource aplikasi Anda di 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`— Atur direktori kerja pada gambar. `/panorama`digunakan untuk kode aplikasi dan file terkait. Pengaturan ini hanya bertahan selama pembuatan dan tidak memengaruhi direktori kerja untuk aplikasi Anda saat runtime ()`/`.
- `COPY`— Menyalin file dari jalur lokal ke jalur pada gambar. `COPY . .`menyalin file di 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 pembuatan. 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 di dalamnya. `requirements.txt`

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

Menentukan dependensi

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

Example Paket/123456789012-sample_code-1.0/requirements.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 citra

Saat Anda membuat gambar untuk paket aplikasi Anda dengan AWS Panorama Application CLI, CLI berjalan di direktori paket. `docker build` Ini membangun gambar aplikasi yang berisi kode aplikasi Anda. CLI kemudian membuat wadah, mengekspor sistem file-nya, 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 pengidentifikasi unik untuk aset saat disimpan adalah Amazon S3.

Memanggil layanan AWS dari kode aplikasi Anda

Anda dapat menggunakan layanan AWS 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 menjalankan fungsi Lambda untuk diproses lebih lanjut. Sebagian besar layanan AWS memiliki API publik yang dapat Anda gunakan dengan AWS SDK.

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

Bagian-bagian

- Menggunakan Amazon S3
- Menggunakan topik AWS IoT MQTT

Menggunakan Amazon S3

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 topik AWS IoT MQTT

Anda dapat menggunakan SDK for Python (Boto3) untuk mengirim pesan ke topik MQTT di. AWS IoT Dalam contoh berikut, aplikasi memposting ke topik yang dinamai sesuai nama benda alat, yang dapat Anda temukan di AWS IoT konsol.

```
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 pengenal lain pilihan Anda. Untuk mempublikasikan pesan, aplikasi memerlukan izin untuk meneleponiot:Publish.

Untuk memantau antrian MQTT

1. Buka halaman Uji AWS IoT konsol.
2. Untuk topik Langganan, masukkan nama topik. Misalnya, `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. Dalam kode aplikasi, Anda menggunakan AWS Panorama Application SDK untuk memuat model visi komputer, menjalankan inferensi, dan mengeluarkan video ke monitor.

> **Note**
>
> Untuk memastikan bahwa Anda memiliki akses ke fungsionalitas terbaru SDK Aplikasi AWS Panorama, tingkatkan perangkat lunak alat.

Untuk detail tentang kelas yang didefinisikan SDK aplikasi dan metodenya, lihat Referensi SDK aplikasi.

Bagian-bagian

- Menambahkan teks dan kotak untuk output video

Menambahkan teks dan kotak untuk output video

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

Setiap objek dalam `video_in` array adalah gambar dari aliran kamera yang terhubung ke alat. Jenis objek ini adalah`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 setiap 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 utas

Anda dapat menjalankan logika aplikasi Anda pada thread pemrosesan dan menggunakan utas lain untuk proses latar belakang lainnya. Misalnya, Anda dapat membuat thread yang melayani lalu lintas HTTP untuk debugging, atau thread yang memantau hasil inferensi dan mengirimkan data ke. AWS

Untuk menjalankan beberapa thread, Anda menggunakan modul threading dari 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 menjalankan tiga thread.

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. `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 thread harus menangani kesalahannya sendiri. Jika kesalahan tidak tertangkap dan dicatat, utas 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 `self` objek aplikasi. Untuk memulai ulang loop pemrosesan aplikasi, utas debugger memanggil metode. `stop` Metode ini menetapkan `terminate` atribut, yang memberi sinyal thread lain untuk dimatikan.

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

```
    # 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 di samping kode aplikasi Anda. Untuk melayani lalu lintas eksternal, Anda memetakan port di 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 tambahan untuk mengamankan alat Anda dari lalu lintas eksternal dan mengamankan komunikasi antara klien resmi dan alat.
> Kode contoh yang disertakan dengan panduan ini adalah untuk tujuan demonstrasi dan tidak menerapkan otentikasi, otorisasi, atau enkripsi.

Anda dapat membuka port dalam kisaran 8000—9000 pada alat. Port ini, ketika dibuka, dapat menerima lalu lintas dari klien yang dapat dirutekan. Saat 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 di port alat yang Anda tentukan dan respons keluar pada port singkat acak.

Mengkonfigurasi port masuk

Anda menentukan pemetaan port di tiga tempat dalam konfigurasi aplikasi Anda. Paket kode`package.json`, Anda menentukan port yang simpul kode mendengarkan di 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 pada `code_node` wadah.

Example graphs/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 ke API. CreateApplicationInstance Anda harus menyediakan konfigurasi ini pada waktu penerapan untuk mengonfirmasi bahwa Anda ingin membuka port pada alat.

Example graphs/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 digunakan oleh aplikasi lain, Anda dapat menggunakan dokumen override untuk memilih port yang berbeda.

Melayani lalu lintas

Dengan port terbuka pada wadah, Anda dapat membuka soket atau menjalankan server untuk menangani permintaan yang masuk. `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 alat 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` jalur untuk mengambil beberapa informasi tentang aplikasi. Ini juga menerima permintaan POST `/restart` untuk me-restart aplikasi.

Untuk mendemonstrasikan fungsionalitas ini, aplikasi sampel menjalankan klien HTTP pada utas terpisah. Klien memanggil `/status` jalur melalui jaringan lokal segera setelah startup, dan memulai ulang 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 memulai ulang 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 menyebarkan aplikasi sampel, lihat instruksi di GitHub repositori panduan ini.

Menggunakan GPU

Anda dapat mengakses prosesor grafis (GPU) di AWS Panorama Appliance untuk menggunakan pustaka yang dipercepat GPU, atau menjalankan model pembelajaran mesin dalam kode aplikasi Anda. Untuk mengaktifkan akses GPU, Anda menambahkan akses GPU sebagai persyaratan ke konfigurasi paket setelah membangun wadah kode aplikasi Anda.

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

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

Example package.json dengan blok persyaratan

```
{
    "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.

Untuk 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-samples GitHub repositori.

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 contoh aplikasi.

Bagian-bagian

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 berikut. PowerShell

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

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

Instal Docker

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

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

> **Note**
>
> Docker Desktop hanya mengaktifkan 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 bawah Sumber Daya, integrasi WSL.

Konfigurasikan Ubuntu

Anda sekarang dapat menjalankan perintah Docker di mesin virtual Ubuntu Anda. Untuk membuka terminal baris perintah, jalankan distribusi dari menu mulai. Pertama kali Anda menjalankannya, Anda mengonfigurasi 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 mengkonfigurasi AWS CLI.

```
$ aws configure
```

   Jika Anda tidak memiliki kunci akses, Anda dapat membuatnya di konsol IAM.

Terakhir, unduh dan impor aplikasi sampel.

Untuk mendapatkan aplikasi sampel

1. Unduh dan ekstrak aplikasi sampel.

```
$ 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 mengunggah 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
```

CLI Aplikasi AWS Panorama mengunggah paket dan mendaftarkannya dengan layanan AWS Panorama. Anda sekarang dapat menerapkan aplikasi sampel dengan konsol AWS Panorama.

Langkah selanjutnya

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

Untuk mengakses sistem file mesin virtual, buka File explorer dan masukkan `\\wsl$` di bilah navigasi. Direktori ini berisi link 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, navigasikan ke `/mnt/c` direktori. Misalnya, Anda dapat membuat daftar file di direktori unduhan 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, kunjungi code.visualstudio.com. Setelah instalasi, tambahkan ekstensi Remote WSL.

Terminal Windows adalah alternatif dari terminal Ubuntu standar tempat Anda menjalankan perintah. Ini mendukung banyak tab dan dapat menjalankan PowerShell, Command Prompt, dan terminal untuk berbagai Linux lainnya yang Anda instal. Ini mendukung salin dan tempel dengan Ctrl+C dan`Ctrl+V`, dapat diklik URLs, dan peningkatan bermanfaat lainnya. Untuk menginstal Terminal Windows, kunjungi microsoft.com.

AWS Panorama API

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

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

Bagian-bagian

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

Otomatiskan pendaftaran perangkat

Untuk menyediakan alat, gunakan ProvisionDeviceAPI. Respons termasuk file ZIP dengan konfigurasi perangkat dan kredenal sementara. Dekode 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 kedaluwarsa setelah 5 menit. Transfer arsip ke alat Anda dengan drive USB yang disertakan.

Untuk mendaftarkan kamera, gunakan CreateNodeFromTemplateJobAPI. 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
```

Kelola peralatan dengan AWS Panorama API

Anda dapat mengotomatiskan tugas manajemen alat dengan AWS Panorama API.

Lihat perangkat

Untuk mendapatkan daftar peralatan dengan perangkat IDs, 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 selengkapnya tentang suatu 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
}
```

Tingkatkan perangkat lunak alat

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 dirinya sendiri. Tonton 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 sedang 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.

Peralatan reboot

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 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 dan. AWS Command Line Interface Setelah membangun wadah aplikasi, Anda mengunggahnya dan aset lainnya ke jalur akses Amazon S3. Anda kemudian menerapkan aplikasi dengan CreateApplicationInstanceAPI.

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

Bagian-bagian

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

Bangun wadahnya

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

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 jalur dengan mengetikkan bagian dari jalur, lalu menekan. TAB

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

Unggah wadah dan daftarkan node

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

Example 4-package-app.sh

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

CLI Aplikasi AWS Panorama mengunggah kontainer dan aset 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 menggunakan CreateApplicationInstanceAPI. Tindakan ini mengambil parameter berikut, antara lain.

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

Muatan manifes dan override 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 menggunakan alat jq untuk membangun dokumen bersarang.

Example 5-deploy.sh - tulis 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 penerapan menggunakan ListDevicesAPI untuk mendapatkan daftar perangkat terdaftar di Wilayah saat ini, dan menyimpan pilihan pengguna ke file lokal untuk penerapan 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 penerapan mendapatkan ARN dari peran ini. AWS CloudFormation Jika aplikasi sudah disebarkan ke perangkat, skrip mendapatkan ID 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 menempatkan semua bagian bersama-sama 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 penerapan, gunakan ListApplicationInstancesAPI. Skrip monitor mendapatkan ID perangkat dan ID instance aplikasi dari file di direktori aplikasi dan menggunakannya untuk membuat perintah CLI. Kemudian memanggil 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 Amazon CloudWatch Logs API. Skrip view logs menggunakan CloudWatch Logs `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
```

Kelola aplikasi dengan AWS Panorama API

Anda dapat memantau dan mengelola aplikasi dengan AWS Panorama API.

Lihat 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 detail 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

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 memutar melalui sejumlah besar aliran kamera daripada yang dapat diproses secara bersamaan. Untuk melakukan ini, petakan beberapa aliran kamera ke simpul input yang sama di manifes penggantian Anda.

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

Example timpalan-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 dan Direct Connect 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. Direct Connect menggunakan AWS perangkat untuk terhubung.

- Site-to-Site VPN — Apa itu AWS Site-to-Site VPN?
- Direct Connect- Apa itu AWS 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 PrivateLink untuk 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 atau AWS . 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 VPC IDs dengan sumber daya Anda. IDs

Contoh aplikasi, skrip, dan templat

GitHub Repositori untuk panduan ini menyediakan contoh aplikasi, skrip, dan templat untuk perangkat. AWS Panorama Gunakan sampel ini untuk mempelajari praktik terbaik dan mengotomatiskan alur kerja pengembangan.

Bagian-bagian

- Aplikasi sampel
- Skrip utilitas
- CloudFormation template
- Lebih banyak sampel dan alat

Aplikasi sampel

Contoh aplikasi menunjukkan penggunaan AWS Panorama fitur dan tugas visi komputer umum. Contoh aplikasi ini mencakup skrip dan templat yang mengotomatiskan penyiapan dan penerapan. Dengan konfigurasi minimal, Anda dapat menyebarkan dan memperbarui aplikasi dari baris perintah.

- aws-panorama-sample— Visi komputer dasar dengan model klasifikasi. Gunakan metrik AWS SDK for Python (Boto) untuk mengunggah ke CloudWatch, metode preprocessing dan inferensi instrumen, dan konfigurasikan logging.
- debug-server — Buka port masuk pada perangkat dan teruskan 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 AI Neo untuk menguji kompatibilitas dengan Appliance. AWS Panorama Bangun secara lokal dalam pengembangan Python, dalam wadah Docker, atau di instance Amazon. EC2 Ekspor dan kompilasi semua model aplikasi bawaan di Keras untuk versi tertentu TensorFlow atau Python.

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

Skrip utilitas

Skrip dalam `util-scripts` direktori mengelola AWS Panorama sumber 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, meng-upload, 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, lihat README.

CloudFormation template

Gunakan CloudFormation template dalam `cloudformation-templates` direktori untuk membuat sumber daya untuk AWS Panorama aplikasi.

- alarm-application.yml-Buat alarm yang memonitor 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.

- application-role.yml-Buat peran aplikasi. Peran tersebut mencakup izin untuk mengirim metrik ke CloudWatch. Tambahkan izin ke pernyataan kebijakan untuk operasi API lain yang digunakan aplikasi Anda.
- vpc-appliance.yml-Buat VPC dengan akses layanan subnet pribadi untuk Appliance. AWS Panorama Untuk menghubungkan alat ke VPC, gunakan AWS Direct Connect atau. AWS Site-to-Site VPN
- vpc-endpoint.yml-Buat VPC dengan akses layanan subnet pribadi ke layanan. AWS Panorama Sumber daya di dalam VPC dapat terhubung AWS Panorama untuk memantau dan mengelola AWS Panorama sumber daya tanpa terhubung ke internet.

`create-stack.sh`Skrip dalam direktori ini membuat CloudFormation tumpukan. 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

aws-panorama-samplesRepositori memiliki lebih banyak contoh aplikasi dan alat yang berguna.

- Aplikasi — Contoh aplikasi untuk berbagai arsitektur model dan kasus penggunaan.
- Validasi aliran kamera - Validasi aliran kamera.
- PanoJupyter— Jalankan JupyterLab di AWS Panorama Appliance.
- Sideloading - Perbarui kode aplikasi tanpa membangun atau menyebarkan wadah aplikasi.

AWS Komunitas juga telah mengembangkan alat dan panduan untuk AWS Panorama. Lihat proyek open source berikut di GitHub.

- cookiecutter-panorama - Template Cookiecutter untuk aplikasi. AWS Panorama
- ransel — Modul Python untuk mengakses detail lingkungan runtime, pembuatan profil, dan opsi keluaran video tambahan.

Memantau AWS Panorama sumber daya dan aplikasi

Anda dapat memantau AWS Panorama sumber daya di AWS Panorama konsol dan dengan Amazon CloudWatch. AWS Panorama Appliance terhubung ke AWS Cloud melalui internet untuk melaporkan status dan status kamera yang terhubung. Saat aktif, alat juga mengirimkan log ke CloudWatch Log secara real time.

Alat mendapat izin untuk menggunakan AWS IoT, CloudWatch Log, dan layanan AWS lainnya dari peran layanan yang Anda buat saat pertama kali menggunakan AWS Panorama konsol. Untuk informasi selengkapnya, lihat Peran layanan AWS Panorama dan sumber daya lintas layanan.

Untuk bantuan memecahkan masalah kesalahan tertentu, lihat. Pemecahan Masalah

Topik

Pemantauan di konsol AWS Panorama

Anda dapat menggunakan konsol AWS Panorama untuk memantau Peralatan dan kamera AWS Panorama Anda. Konsol digunakan AWS IoT untuk memantau keadaan alat.

Untuk memantau alat Anda di konsol AWS Panorama

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

Status keseluruhan alat muncul di bagian atas halaman. Jika statusnya Online, maka alat terhubung ke AWS dan mengirim pembaruan status reguler.

Melihat log AWS Panorama

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

Untuk melihat log di CloudWatch Log

1. Buka halaman Grup log dari konsol CloudWatch 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 dapat melihat log pada drive USB penyediaan.

Bagian-bagian

Melihat log perangkat

AWS Panorama Appliance membuat grup log untuk perangkat, dan grup untuk setiap instance 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`— Keluaran dari proses yang mengkoordinasikan over-the-air (OTA) peningkatan perangkat lunak.

- `syslog`— Output dari proses syslog perangkat, yang menangkap pesan yang dikirim antar proses.
- `kern_log`— Peristiwa dari kernel Linux perangkat.
- `logging_setup_logs`— Output dari proses yang mengkonfigurasi agen CloudWatch Log.
- `cloudwatch_agent_logs`— Output dari agen CloudWatch Log.
- `shadow_log`— Output dari bayangan AWS IoT perangkat.

Melihat log aplikasi

Grup log instance aplikasi berisi aliran log untuk setiap node, dinamai menurut node.

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 mengkoordinasikan permintaan inferensi dengan model.
- Stream — Output dari proses yang menerjemahkan 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 CloudWatch log di Log, konfirmasikan bahwa Anda berada di Wilayah AWS yang benar. Jika ya, mungkin ada masalah dengan koneksi alat ke AWS atau dengan izin pada peran alat AWS Identity and Access Management (IAM).

Mengkonfigurasi log aplikasi

Konfigurasikan logger Python untuk menulis file log ke. `/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 menunjukkan metode yang menciptakan logger.

Example application.py - konfigurasi Logger

```
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 logger

```
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`— Log agen over-the-air pembaruan AWS Panorama.
- `/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 memerlukan perangkat keras berikut:

- USB drive — Drive memori flash USB FAT32 yang diformat 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 `panorama` folder.

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

2. Hubungkan drive USB ke perangkat.
3. Matikan AWS Panorama Appliance.
4. Nyalakan AWS Panorama Appliance.
5. Perangkat menyalin log ke perangkat. LED status berkedip biru saat 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 Amazon CloudWatch

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

Untuk melihat metrik di konsol CloudWatch

1. Buka halaman AWS Panorama console Metrics (`PanoramaDeviceMetrics`namespace).
2. Pilih 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**
>
> CloudWatch memiliki tingkat Selalu Gratis. Di luar ambang batas tingkat gratis, CloudWatch biaya untuk metrik, dasbor, alarm, log, dan wawasan. Untuk detailnya, lihat CloudWatch harga.

Untuk informasi selengkapnya CloudWatch, lihat Panduan CloudWatch Pengguna Amazon.

Bagian-bagian

- Menggunakan metrik perangkat
- Menggunakan metrik aplikasi
- Mengkonfigurasi alarm

Menggunakan metrik perangkat

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

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

  Dimensi — `DeviceId` dan`DeviceName`.

Lihat `DeviceActive` metrik dengan `Average` statistik.

Menggunakan metrik aplikasi

Ketika aplikasi menemukan kesalahan, ia mengirimkan metrik ke Amazon. CloudWatch Anda dapat menggunakan metrik ini untuk memicu alarm jika aplikasi berhenti berjalan.

- `ApplicationErrors`— Jumlah kesalahan aplikasi yang direkam.

  Dimensi — `ApplicationInstanceName` dan`ApplicationInstanceId`.

Lihat metrik aplikasi dengan `Sum` statistik.

Mengkonfigurasi alarm

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

Untuk membuat alarm

1. Buka halaman Alarm CloudWatch konsol Amazon.
2. Pilih Buat alarm.
3. Pilih Pilih metrik dan temukan metrik untuk perangkat Anda, seperti `ApplicationErrors` untuk`applicationInstance-gk75xmplqbqtenlnmz4ehiu7xa`,`my-application`.
4. Ikuti petunjuk untuk mengonfigurasi kondisi, tindakan, dan nama alarm.

Untuk petunjuk terperinci, lihat Membuat CloudWatch alarm di Panduan CloudWatch Pengguna Amazon.

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 Panorama Alat 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® SE7 0, 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 Panorama tidak 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 Manager Rahasia 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 adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

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

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS Panorama. Topik berikut menunjukkan cara mengonfigurasi 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 Anda.

Topik

Fitur keamanan AWS Panorama Appliance

Untuk melindungi aplikasi, model, dan perangkat keras Anda dari kode berbahaya dan eksploitasi lainnya, AWS Panorama Appliance mengimplementasikan serangkaian fitur keamanan yang luas. Ini termasuk tetapi tidak terbatas pada yang berikut.

- Enkripsi disk penuh - Alat ini mengimplementasikan penyetelan kunci terpadu Linux (LUKS2) enkripsi disk penuh. Semua perangkat lunak sistem dan data aplikasi dienkripsi dengan kunci yang khusus untuk perangkat Anda. Bahkan dengan akses fisik ke perangkat, penyerang tidak dapat memeriksa isi penyimpanannya.
- Pengacakan tata letak memori — Untuk melindungi dari serangan yang menargetkan kode yang dapat dieksekusi yang dimuat ke dalam memori, AWS Panorama Appliance menggunakan pengacakan tata letak ruang alamat (ASLR). ASLR mengacak lokasi kode sistem operasi saat dimuat ke dalam memori. Ini mencegah penggunaan eksploitasi yang mencoba menimpa atau menjalankan bagian kode tertentu dengan memprediksi di mana ia disimpan saat runtime.
- Lingkungan eksekusi tepercaya — Alat ini menggunakan lingkungan eksekusi tepercaya (TEE) berdasarkan ARM TrustZone, dengan penyimpanan, memori, dan sumber daya pemrosesan yang terisolasi. Kunci dan data sensitif lainnya yang disimpan di zona kepercayaan hanya dapat diakses oleh aplikasi tepercaya, yang berjalan dalam sistem operasi terpisah di dalam TEE. Perangkat lunak AWS Panorama Appliance berjalan di lingkungan Linux yang tidak tepercaya bersama kode aplikasi. Itu hanya dapat mengakses operasi kriptografi dengan membuat permintaan ke aplikasi yang aman.
- Penyediaan aman — Saat Anda menyediakan alat, kredensi (kunci, sertifikat, dan materi kriptografi lainnya) yang Anda transfer ke perangkat hanya berlaku untuk waktu yang singkat. Alat menggunakan kredensil berumur pendek untuk terhubung AWS IoT dan meminta sertifikat untuk dirinya sendiri yang valid untuk waktu yang lebih lama. Layanan AWS Panorama menghasilkan kredensil dan mengenkripsi mereka dengan kunci yang di-hardcode 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 berjalan. ROM boot, perangkat lunak yang di-hardcode dalam prosesor yang tidak dapat dimodifikasi, menggunakan kunci enkripsi hardcode 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 dalam memori.
- dm-verity — Mirip dengan bagaimana modul kernel divalidasi, alat menggunakan `dm-verity` fitur Linux Device Mapper untuk memverifikasi integritas gambar perangkat lunak alat sebelum memasangnya. Jika perangkat lunak alat dimodifikasi, itu tidak akan berjalan.
- Pencegahan rollback — Saat 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 yang meledak.

Praktik terbaik keamanan AWS Panorama Appliance

Perhatikan praktik terbaik berikut saat menggunakan alat AWS Panorama.

- Amankan alat secara fisik — Pasang alat di rak server tertutup atau ruang aman. Batasi akses fisik ke perangkat ke personel yang berwenang.
- Amankan koneksi jaringan alat — Hubungkan alat ke router yang membatasi akses ke sumber daya internal dan eksternal. Alat perlu terhubung ke kamera, yang dapat berada di jaringan internal yang aman. Itu juga perlu terhubung ke AWS. Gunakan port Ethernet kedua hanya untuk redundansi fisik, dan konfigurasikan router untuk memungkinkan 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, lepaskan drive USB dan format. Alat tidak menggunakan drive USB setelah mendaftar dengan layanan AWS Panorama. Format drive untuk menghapus kredensi sementara, file konfigurasi, dan log penyediaan.
- Tetap perbarui alat — Terapkan pembaruan perangkat lunak alat tepat waktu. Saat Anda melihat alat di konsol AWS Panorama, konsol akan memberi tahu Anda jika pembaruan perangkat lunak tersedia. Untuk informasi selengkapnya, lihat Mengelola Alat Panorama AWS.

  Dengan operasi DescribeDeviceAPI, Anda dapat mengotomatiskan pemeriksaan pembaruan dengan membandingkan `LatestSoftware` dan `CurrentSoftware` bidang. Ketika versi perangkat lunak terbaru berbeda dari versi saat ini, terapkan pembaruan dengan konsol atau dengan menggunakan CreateJobForDevicesoperasi.
- Jika Anda berhenti menggunakan alat, setel ulang — Sebelum Anda memindahkan alat keluar dari pusat data aman Anda, setel ulang sepenuhnya. Dengan alat dimatikan dan dicolokkan, tekan tombol daya 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 — The AWSPanoramaFullAccessmenyediakan 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. Misalnya, ia menyediakan akses ke AWS Secrets Manager rahasia 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 hak istimewa paling tidak Anda miliki.

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

Perlindungan data di AWS Panorama

Model tanggung jawab AWS bersama model tanggung berlaku untuk perlindungan data di AWS Panorama. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host 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, kami menyarankan Anda melindungi Akun AWS kredensyal dan mengatur pengguna individu dengan 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 pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

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, API AWS CLI, atau. AWS SDKs 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 bergerak

Titik akhir AWS Panorama API mendukung koneksi aman hanya melalui HTTPS. Saat Anda mengelola sumber daya AWS Panorama dengan AWS SDK, atau Konsol Manajemen AWS 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 AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya AWS Panorama. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda berdasarkan peran Anda:

- Pengguna layanan - minta izin dari administrator Anda jika Anda tidak dapat mengakses fitur (lihat Memecahkan masalah identitas dan akses AWS Panorama)
- Administrator layanan - tentukan akses pengguna dan mengirimkan permintaan izin (lihat Cara AWS Panorama bekerja dengan IAM)
- Administrator IAM - tulis kebijakan untuk mengelola akses (lihat Contoh kebijakan berbasis identitas AWS Panorama)

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi sebagai Pengguna root akun AWS, pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk sebagai identitas federasi menggunakan kredensil dari sumber identitas seperti AWS IAM Identity Center (Pusat Identitas IAM), autentikasi masuk tunggal, atau kredensional. Google/Facebook Untuk informasi selengkapnya tentang cara masuk, lihat Cara masuk ke Akun AWS Anda dalam Panduan Pengguna AWS Sign-In .

Untuk akses terprogram, AWS sediakan SDK dan CLI untuk menandatangani permintaan secara kriptografis. Untuk informasi selengkapnya, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang disebut pengguna Akun AWS root yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Untuk tugas yang memerlukan kredensial pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Pengguna dan grup IAM

Pengguna IAM adalah identitas dengan izin khusus untuk satu orang atau aplikasi. Sebaiknya gunakan kredensial sementara alih-alih pengguna IAM dengan kredensial jangka panjang. Untuk informasi selengkapnya, lihat Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara di Panduan Pengguna IAM.

Grup IAM menentukan kumpulan pengguna IAM dan mempermudah pengelolaan izin untuk pengguna dalam jumlah besar. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dengan izin khusus yang menyediakan kredensial sementara. Anda dapat mengambil peran dengan beralih dari pengguna ke peran IAM (konsol) atau dengan memanggil operasi AWS CLI atau AWS API. Untuk informasi selengkapnya, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM berguna untuk akses pengguna terfederasi, izin pengguna IAM sementara, akses lintas akun, akses lintas layanan, dan aplikasi yang berjalan di Amazon EC2. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan menentukan izin saat dikaitkan dengan identitas atau sumber daya. AWS mengevaluasi kebijakan ini ketika kepala sekolah membuat permintaan. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Menggunakan kebijakan, administrator menentukan siapa yang memiliki akses ke apa dengan mendefinisikan principal mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Administrator IAM membuat kebijakan IAM dan menambahkannya ke peran, yang kemudian dapat diambil oleh pengguna. Kebijakan IAM mendefinisikan izin terlepas dari metode yang Anda gunakan untuk melakukan operasinya.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang Anda lampirkan ke identitas (pengguna, grup, atau peran). Kebijakan ini mengontrol tindakan apa yang bisa dilakukan oleh identitas tersebut, terhadap sumber daya yang mana, dan dalam kondisi apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan yang dikelola pelanggan dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat berupa kebijakan inline (disematkan langsung ke dalam satu identitas) atau kebijakan terkelola (kebijakan mandiri yang dilampirkan pada banyak identitas). Untuk mempelajari cara memilih antara kebijakan terkelola dan kebijakan inline, lihat Pilih 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. Contohnya termasuk 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. Anda harus menentukan principal dalam kebijakan berbasis sumber daya.

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

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat Ringkasan daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang dapat menetapkan izin maksimum yang diberikan oleh jenis kebijakan yang lebih umum:

- Batasan izin – Menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM. Untuk informasi selengkapnya, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCPs) — Tentukan izin maksimum untuk organisasi atau unit organisasi di AWS Organizations. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam Panduan Pengguna AWS Organizations .
- Kebijakan kontrol sumber daya (RCPs) — Tetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda. Untuk informasi selengkapnya, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan lanjutan yang diteruskan sebagai parameter saat membuat sesi sementara untuk peran atau pengguna terfederasi. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di 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, AWS lihat layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

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

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 Konsol Manajemen AWS, AWS CLI, atau AWS API. 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
- Mengizinkan 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 membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam 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 melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – 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 dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

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

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

Mengizinkan pengguna melihat izin mereka sendiri

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

```
{
    "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": "*"
        }
    ]
}
```

AWS kebijakan terkelola untuk AWS Panorama

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan 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, jalur akses AWS Panorama di Amazon S3, kredensi alat di, dan log alat di AWS Secrets Manager Amazon. CloudWatch Termasuk izin untuk membuat peran terkait layanan untuk 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 ke CloudWatch, dan untuk mendapatkan objek dari titik akses Amazon S3 yang dibuat oleh AWS Panorama.

AWS Panorama memperbarui kebijakan terkelola AWS

Tabel berikut menjelaskan pembaruan kebijakan terkelola untuk AWS Panorama.

Perubahan	Deskripsi	Tanggal
AWSPanoramaApplianceServiceRolePolicy — Perbarui ke kebijakan yang ada	Ganti StringLike kondisi dengan ArnLike untuk menulis ARNs.	2024-12-10
AWSPanoramaFullAccess — Perbarui ke kebijakan yang ada	Ganti StringLike kondisi dengan ArnLike untuk menulis ARNs.	2024-12-10
AWSPanoramaFullAccess — Perbarui ke kebijakan yang ada	Menambahkan izin ke kebijakan pengguna untuk memungkinkan pengguna melihat grup log di konsol CloudWatch Log.	2022-01-13
AWSPanoramaFullAccess — Perbarui ke kebijakan yang ada	Menambahkan izin ke kebijakan pengguna untuk memungkinkan pengguna mengelola peran terkait layanan AWS Panorama, dan mengakses sumber daya AWS Panorama di layanan lain termasuk IAM, Amazon S3,, dan Secrets Manager. CloudWatch	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

Perubahan	Deskripsi	Tanggal
AWS Panorama mulai melacak perubahan	AWS Panorama mulai melacak perubahan untuk kebijakan yang AWS dikelola.	2021-10-20

Menggunakan peran terkait layanan untuk AWS Panorama

AWS Panorama menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran terkait layanan adalah jenis unik peran IAM yang ditautkan langsung ke. AWS Panorama Peran terkait layanan telah ditentukan sebelumnya oleh AWS Panorama dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran terkait layanan membuat pengaturan AWS Panorama lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. AWS Panorama mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya AWS Panorama dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut 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 terkait layanan, silakan lihat layanan AWS yang bisa digunakan dengan IAM dan carilah layanan yang memiliki opsi Ya di kolom Peran terkait layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Bagian-bagian

- Izin peran terkait layanan untuk AWS Panorama
- Membuat peran terkait layanan untuk AWS Panorama
- Mengedit peran terkait layanan untuk AWS Panorama
- Menghapus peran terkait layanan untuk AWS Panorama
- Wilayah yang Didukung untuk AWS Panorama peran terkait layanan

Izin peran terkait layanan untuk AWS Panorama

AWS Panorama menggunakan peran terkait layanan bernama AWSServiceRoleForAWSPanorama—Memungkinkan AWS Panorama mengelola sumber daya di AWS IoT, AWS Secrets Manager, dan AWS Panorama..

Peran AWSService RoleFor AWSPanorama terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `panorama.amazonaws.com`

Kebijakan izin peran memungkinkan AWS Panorama untuk menyelesaikan tindakan berikut:

- Pantau AWS Panorama sumber daya
- Mengelola AWS IoT sumber daya untuk AWS Panorama Appliance
- Akses AWS Secrets Manager rahasia untuk mendapatkan kredensi kamera

Untuk daftar lengkap izin, lihat AWSPanorama ServiceLinkedRolePolicy kebijakan di 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 tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran terkait layanan untuk AWS Panorama

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda mendaftarkan alat di Konsol Manajemen AWS, AWS CLI, atau AWS API, AWS Panorama buat peran terkait 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 Panorama buat peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk AWS Panorama

AWS Panorama tidak memungkinkan Anda untuk mengedit peran AWSService RoleFor AWSPanorama terkait layanan. 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 selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk AWS Panorama

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, kami merekomendasikan 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 menghapus AWS Panorama sumber daya yang digunakan oleh AWSService RoleForAWSPanorama, gunakan prosedur di bagian berikut dari panduan ini.

- Hapus versi dan aplikasi
- Deregister alat

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

Untuk menghapus peran AWSService RoleFor AWSPanorama terkait layanan, gunakan konsol IAM, the AWS CLI, atau API. AWS Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang Didukung untuk AWS Panorama peran terkait layanan

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

Pencegahan "confused deputy" lintas layanan

Masalah "confused deputy" adalah masalah keamanan saat entitas yang tidak memiliki izin untuk melakukan suatu tindakan dapat memaksa entitas yang memilik hak akses lebih tinggi untuk

melakukan tindakan tersebut. Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses. Untuk mencegah hal ini, AWS sediakan alat yang membantu Anda melindungi data Anda untuk semua layanan dengan prinsip layanan yang telah diberikan akses ke sumber daya di akun Anda.

Sebaiknya gunakan kunci konteks kondisi `aws:SourceAccount`global `aws:SourceArn`dan dalam kebijakan sumber daya untuk membatasi izin yang AWS Panorama memberikan layanan lain ke sumber daya. Jika Anda menggunakan kedua kunci konteks kondisi global, `aws:SourceAccount` nilai dan akun dalam `aws:SourceArn` nilai harus menggunakan ID akun yang sama saat digunakan dalam pernyataan kebijakan yang sama.

Nilai `aws:SourceArn` harus ARN dari suatu AWS Panorama perangkat.

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

Untuk petunjuk tentang mengamankan peran layanan yang AWS Panorama digunakan untuk memberikan izin ke AWS Panorama Appliance, 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 Konsol Manajemen AWS memberitahu Anda bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk bantuan. Administrator Anda adalah orang yang memberikan nama pengguna dan kata sandi 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.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

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 memerlukan bantuan, hubungi AWS administrator 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 orang-orang 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 (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mengetahui apakah AWS Panorama mendukung fitur-fitur ini, lihat. Cara AWS Panorama bekerja dengan IAM
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM 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. Untuk informasi selengkapnya tentang tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS, lihat Dokumentasi AWS Keamanan.

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.
- Mempertimbangkan 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 oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses AWS Panorama melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Menerapkan AWS Panorama Appliance di pusat data Anda

AWS Panorama Appliance membutuhkan akses internet untuk berkomunikasi dengan AWS layanan. 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. Berhati-hatilah 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 dari 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, lihat. Menghubungkan 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 memperbarui perangkat lunak dalam gambar alat. AWS Panorama secara teratur merilis pembaruan perangkat lunak, yang dapat Anda terapkan dengan 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 spesifik 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
Pemberitahuan akhir dukungan	Pemberitahuan akhir dukungan: Pada 31 Mei 2026, AWS akan mengakhiri dukungan untuk AWS Panorama. Setelah 31 Mei 2026, Anda tidak akan lagi dapat mengakses AWS Panorama konsol atau AWS Panorama sumber daya. Untuk informasi lebih lanjut, lihat AWS Panorama akhir dukungan.	Mei 20, 2025
Kebijakan terkelola yang diperbarui	AWS Identity and Access Management kebijakan terkelola untuk AWS Panorama telah diperbarui. Untuk detailnya, lihat kebijakan terkelola AWS.	Desember 10, 2024
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	28 Desember 2023

	pribadi, Anda harus mengizinkan akses ke titik akhir dan port tambahan sebelum menerapkan pembaruan. Untuk informasi selengkapnya, lihat log perubahan.	
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 peningkatan penanganan kesalahan. Untuk informasi selengkapnya, lihat log perubahan.	31 Maret 2023
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 selengkapnya, 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 PrivateLink dukungan	AWS Panorama mendukung 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.	5 Mei 2022

Lenovo ThinkEdge® SE7 0	Alat baru untuk AWS Panorama tersedia dari Lenovo. Lenovo ThinkEdge® SE7 0, 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 menambahk an 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 Management kebijakan terkelola untuk AWS Panorama telah diperbaru i. 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 pengatura n NTP selama pengaturan alat dengan pengaturan jaringan lainnya. Untuk informasi selengkapnya, lihat Menyiapka n.	Januari 13, 2022
Wilayah tambahan	AWS Panorama sekarang tersedia di Wilayah Asia Pasifik (Singapura) dan Asia Pasifik (Sydney).	Januari 13, 2022
Pembaruan perangkat lunak alat	Versi 4.3.4 menambahkan dukungan untuk `precision Mode` pengaturan untuk model dan memperbarui perilaku logging. Untuk informasi selengkapnya, lihat log perubahan.	November 8, 2021

Kebijakan terkelola yang diperbarui	AWS Identity and Access Management kebijakan terkelola untuk AWS Panorama telah diperbaru i. Untuk detailnya, lihat kebijakan terkelola AWS.	20 Oktober 2021
Ketersediaan umum	AWS Panorama sekarang 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 Panorama tersedia melalui undangan di Wilayah AS Timur (Virginia N.) dan AS Barat (Oregon).	1 Desember 2020