aws

Panduan Pengembang WebRTC Amazon Kinesis Video Streams

Kinesis Video Streams

Kinesis Video Streams: Panduan Pengembang WebRTC Amazon Kinesis Video Streams

Table of Contents

Apa itu Amazon Kinesis Video Streams dengan WebRTC?

WebRTC adalah spesifikasi teknologi terbuka untuk memungkinkan komunikasi real-time (RTC) di seluruh browser dan aplikasi seluler melalui simple. APIs Ini menggunakan teknik peering untuk pertukaran data real-time antara rekan-rekan yang terhubung dan menyediakan streaming media latensi rendah yang diperlukan untuk human-to-human interaksi. Spesifikasi WebRTC mencakup seperangkat protokol IETF termasuk Interactive Connectivity Establishment, Traversal Using Relay around NAT (TURN), dan Session Traversal Utilities for NAT (STUN) untuk peer-to-peer membangun konektivitas, selain spesifikasi protokol untuk media real-time dan streaming data yang andal dan aman.

Amazon Kinesis Video Streams menyediakan implementasi WebRTC yang sesuai standar sebagai kemampuan yang dikelola sepenuhnya. Anda dapat menggunakan Amazon Kinesis Video Streams dengan WebRTC untuk media streaming langsung dengan aman atau melakukan interaksi audio atau video dua arah antara perangkat IoT kamera apa pun dan pemutar seluler atau web yang sesuai dengan WebRTC. Sebagai kemampuan yang dikelola sepenuhnya, Anda tidak perlu membangun, mengoperasikan, atau menskalakan infrastruktur cloud terkait WebRTC apa pun, seperti server pensinyalan atau relai media untuk mengalirkan media secara aman di seluruh aplikasi dan perangkat.

Menggunakan Kinesis Video Streams dengan WebRTC, Anda dapat dengan mudah membangun peer-to-peer aplikasi untuk streaming media langsung, atau interaktivitas audio atau video real-time antara perangkat IoT kamera, browser web, dan perangkat seluler untuk berbagai kasus penggunaan. Aplikasi semacam itu dapat membantu orang tua mengawasi kamar bayi mereka, memungkinkan pemilik rumah menggunakan bel pintu video untuk memeriksa siapa yang ada di pintu, memungkinkan pemilik penyedot debu robot yang diaktifkan kamera untuk mengontrol robot dari jarak jauh dengan melihat aliran kamera langsung di ponsel, dan sebagainya.

Jika Anda pengguna pertama kali Kinesis Video Streams dengan WebRTC, kami sarankan Anda membaca bagian berikut:

- Cara kerjanya
- Amazon Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan
- Amazon Kinesis Video Streams dengan WebRTC JavaScript SDK untuk aplikasi web
- Amazon Kinesis Video Streams WebRTC SDK untuk Android
- Amazon Kinesis Video Streams WebRTC SDK untuk iOS
- Pesawat kontrol APIs

- Pesawat data REST APIs
- Bidang data Websocket APIs

Ketersediaan wilayah

> **Note**
>
> Amazon Kinesis Video Streams dengan WebRTC belum didukung di. AWS GovCloud (US) Region

Amazon Kinesis Video Streams dengan WebRTC tersedia di wilayah berikut:

Nama wilayah	AWS Kode Wilayah
AS Timur (Ohio)	us-east-2
US East (N. Virginia)	us-east-1
AS Barat (Oregon)	as-barat-2
Afrika (Cape Town)	af-selatan-1
Asia Pasifik (Hong Kong)	ap-timur-1
Asia Pasifik (Mumbai)	ap-south-1
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pasifik (Malaysia)	ap-southeast-5
Asia Pasifik (Tokyo)	ap-northeast-1
Canada (Central)	ca-sentral-1

Nama wilayah	AWS Kode Wilayah
Europe (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2
Europe (Paris)	eu-west-3
Eropa (Spanyol)	eu-south-2
Amerika Selatan (Sao Paulo)	sa-east-1
Timur Tengah (Bahrain)	me-south-1

Cara kerjanya

Amazon Kinesis Video Streams dengan WebRTC memungkinkan komunikasi video real-time antara browser web, perangkat seluler, dan aplikasi lain yang mendukung WebRTC. Ini menyediakan infrastruktur yang aman dan terukur untuk membangun aplikasi streaming video, menangani tugas-tugas seperti pensinyalan, streaming media, dan integrasi dengan layanan lain AWS . Bagian ini menjelaskan arsitektur, komponen, dan alur kerja layanan yang mendasari, membantu menjelaskan prinsip dan mekanisme desainnya.

Topik

Konsep utama

Berikut ini adalah istilah dan konsep kunci khusus untuk Amazon Kinesis Video Streams dengan WebRTC.

Saluran pensinyalan

Sumber daya yang memungkinkan aplikasi menemukan, mengatur, mengontrol, dan mengakhiri peer-to-peer koneksi dengan bertukar pesan sinyal. Pesan sinyal adalah metadata yang dipertukarkan dua aplikasi satu sama lain untuk membangun konektivitas. peer-to-peer Metadata ini mencakup informasi media lokal, seperti codec media dan parameter codec, dan kemungkinan jalur kandidat jaringan untuk kedua aplikasi untuk terhubung satu sama lain untuk streaming langsung.

Aplikasi streaming dapat mempertahankan konektivitas persisten dengan saluran pensinyalan dan menunggu aplikasi lain terhubung dengannya. Atau, mereka dapat terhubung ke saluran pensinyalan hanya ketika mereka perlu melakukan streaming langsung media. Saluran pensinyalan memungkinkan aplikasi untuk terhubung satu sama lain dalam sebuah one-to-few model, menggunakan konsep satu master yang terhubung ke beberapa pemirsa. Aplikasi yang memulai koneksi memikul tanggung jawab master menggunakan `ConnectAsMaster` API dan menunggu pemirsa. Hingga 10 aplikasi kemudian dapat terhubung ke saluran pensinyalan itu dengan memikul tanggung jawab pemirsa dengan menjalankan API. `ConnectAsViewer` Setelah terhubung ke saluran pensinyalan, aplikasi master dan penampil dapat saling mengirim pesan sinyal untuk membangun peer-to-peer konektivitas untuk streaming media langsung.

Sebaya

Perangkat atau aplikasi apa pun (misalnya, aplikasi seluler atau web, kamera web, kamera keamanan rumah, monitor bayi, dll.) yang dikonfigurasi untuk streaming dua arah real-time melalui Kinesis Video Streams dengan WebRTC.

Tuan

Rekan yang memulai koneksi dan terhubung ke saluran pensinyalan dengan kemampuan untuk menemukan dan bertukar media dengan pemirsa terhubung saluran sinyal mana pun.

> **Important**
>
> Saat ini, saluran pensinyalan hanya dapat memiliki satu master.

Pemirsa

Rekan yang terhubung ke saluran pensinyalan dengan kemampuan untuk menemukan dan bertukar media hanya dengan master saluran pensinyalan. Penampil tidak dapat menemukan

atau berinteraksi dengan pemirsa lain melalui saluran pensinyalan tertentu. Saluran pensinyalan dapat memiliki hingga 10 pemirsa yang terhubung.

Konsep teknologi

Saat Anda memulai Kinesis Video Streams dengan WebRTC, Anda juga dapat memperoleh manfaat dari mempelajari beberapa protokol yang saling terkait APIs dan yang terdiri dari teknologi WebRTC.

Utilitas Traversal Sesi untuk NAT (STUN)

Protokol yang digunakan untuk menemukan alamat publik Anda dan menentukan batasan apa pun di router Anda yang akan mencegah koneksi langsung dengan rekan.

Traversal Menggunakan Relay di sekitar NAT (TURN)

Server yang digunakan untuk melewati pembatasan NAT Simetris dengan membuka koneksi dengan server TURN dan menyampaikan semua informasi melalui server itu.

Protokol Deskripsi Sesi (SDP)

Standar untuk menggambarkan konten multimedia dari koneksi seperti resolusi, format, codec, enkripsi, dll. Sehingga kedua rekan dapat saling memahami setelah data ditransfer.

Penawaran SDP

Pesan SDP yang dikirim oleh agen yang menghasilkan deskripsi sesi untuk membuat atau memodifikasi sesi. Ini menggambarkan aspek komunikasi media yang diinginkan.

SDP Jawaban

Pesan SDP yang dikirim oleh penjawab sebagai tanggapan atas tawaran yang diterima dari penawaran. Jawabannya menunjukkan aspek-aspek yang diterima. Misalnya, jika semua aliran audio dan video dalam penawaran diterima.

Pembentukan Konektivitas Interaktif (ICE)

Kerangka kerja yang memungkinkan browser web Anda terhubung dengan rekan-rekan.

Kandidat ICE

Metode yang dapat digunakan rekan pengirim untuk berkomunikasi.

Bagaimana STUN, TURN, dan ICE bekerja sama

Mari kita ambil skenario dua rekan, A dan B, yang keduanya menggunakan peer to peer webRTC streaming media dua arah (misalnya, aplikasi obrolan video). Apa yang terjadi ketika A ingin memanggil B?

Untuk terhubung ke aplikasi B, aplikasi A harus menghasilkan penawaran SDP. Penawaran SDP berisi informasi tentang sesi A yang ingin dibuat oleh aplikasi, termasuk codec apa yang akan digunakan, apakah ini sesi audio atau video, dll. Ini juga berisi daftar kandidat ICE, yang merupakan pasangan IP dan port yang dapat coba digunakan oleh aplikasi B untuk terhubung ke A.

Untuk membangun daftar kandidat ICE, aplikasi A membuat serangkaian permintaan ke server STUN. Server mengembalikan alamat IP publik dan pasangan port yang berasal dari permintaan. Aplikasi A menambahkan setiap pasangan ke daftar kandidat ICE, dengan kata lain, ia mengumpulkan kandidat ICE. Setelah aplikasi A selesai mengumpulkan kandidat ICE, ia dapat mengembalikan SDP.

Selanjutnya, aplikasi A harus meneruskan SDP ke aplikasi B melalui saluran pensinyalan di mana aplikasi ini berkomunikasi. Protokol transport untuk pertukaran ini tidak ditentukan dalam standar WebRTC. Ini dapat dilakukan melalui HTTPS, aman WebSocket, atau protokol komunikasi lainnya.

Sekarang, aplikasi B harus menghasilkan jawaban SDP. Aplikasi B mengikuti langkah yang sama A yang digunakan pada langkah sebelumnya: mengumpulkan kandidat ICE, dll. Aplikasi B kemudian perlu mengembalikan jawaban SDP ini ke aplikasi A.

Setelah A dan B bertukar SDPs, mereka kemudian melakukan serangkaian pemeriksaan konektivitas. Algoritma ICE di setiap aplikasi mengambil IP/port pasangan kandidat dari daftar yang diterimanya di SDP pihak lain, dan mengirimkannya permintaan STUN. Jika respons kembali dari aplikasi lain, aplikasi asal menganggap cek berhasil dan menandai IP/port pasangan itu sebagai kandidat ICE yang valid.

Setelah pemeriksaan konektivitas selesai pada semua IP/port pasangan, aplikasi bernegosiasi dan memutuskan untuk menggunakan salah satu pasangan yang tersisa dan valid. Ketika pasangan dipilih, media mulai mengalir di antara aplikasi.

Jika salah satu aplikasi tidak dapat menemukan IP/port pasangan yang melewati pemeriksaan konektivitas, mereka akan membuat permintaan STUN ke server TURN untuk mendapatkan alamat relay media. Alamat relay adalah alamat IP publik dan port yang meneruskan paket yang diterima ke dan dari aplikasi untuk mengatur alamat relay. Alamat relay ini kemudian ditambahkan ke daftar kandidat dan dipertukarkan melalui saluran pensinyalan.

Komponen-komponen

Kinesis Video Streams dengan WebRTC mencakup komponen-komponen berikut:

- Bidang kontrol

  Komponen bidang kontrol bertanggung jawab untuk membuat dan memelihara Kinesis Video Streams dengan saluran pensinyalan WebRTC. Untuk informasi selengkapnya, lihat Referensi API Amazon Kinesis Video Streams.

- Pensinyalan

  Komponen pensinyalan mengelola titik akhir pensinyalan WebRTC yang memungkinkan aplikasi terhubung dengan aman satu sama lain untuk streaming media langsung. peer-to-peer Komponen pensinyalan termasuk Amazon Kinesis Video Signaling APIs REST dan satu set Websocket. APIs

- SETRUM

  Komponen ini mengelola titik akhir STUN yang memungkinkan aplikasi menemukan alamat IP publik mereka ketika mereka berada di belakang NAT atau firewall.

- PUTAR

  Komponen ini mengelola titik akhir TURN yang mengaktifkan relai media melalui cloud saat aplikasi tidak dapat melakukan streaming media peer-to-peer.

- Kinesis Video Streams WebRTC SDKs

  Ini adalah pustaka perangkat lunak yang dapat Anda unduh, instal, dan konfigurasikan di perangkat dan klien aplikasi Anda untuk mengaktifkan perangkat IoT kamera Anda dengan kemampuan WebRTC untuk terlibat dalam streaming media latensi rendah. peer-to-peer Ini SDKs juga memungkinkan klien aplikasi Android, iOS, dan web untuk mengintegrasikan Kinesis Video Streams dengan kemampuan pensinyalan WebRTC, TURN, dan STUN dengan pemutar seluler atau web yang sesuai dengan WebRTC.

  - Amazon Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan
  - Amazon Kinesis Video Streams dengan WebRTC JavaScript SDK untuk aplikasi web
  - Amazon Kinesis Video Streams WebRTC SDK untuk Android
  - Amazon Kinesis Video Streams WebRTC SDK untuk iOS

Persyaratan sistem

Bagian ini mencakup persyaratan sistem dasar untuk menggunakan Amazon Kinesis Video Streams dengan WebRTC, termasuk persyaratan jaringan dan lingkungan. Ini juga mencakup informasi tentang koneksi debugging.

Persyaratan jaringan

Persyaratan jaringan umum untuk titik akhir layanan saluran pensinyalan untuk Kinesis Video Streams dengan WebRTC adalah:

- Panggilan HTTPS ke titik akhir yang dihosting di `https://*.kinesisvideo.{region}.amazonaws.com`
- WebSocket integrasi dengan titik akhir `wss://*.kinesisvideo.{region}.amazonaws.com`
- STUNserver di `stun:stun.kinesisvideo.{aws-region}.amazonaws.com:443`
- TURNserver di `turn:_._.kinesisvideo.{aws-region}.amazonaws.com:443` dan `turns:_._.kinesisvideo.{aws-region}.amazonaws.com:443`

> **Note**
>
> IPv6 alamat saat ini tidak didukung untuk STUN dan TURN server.

Protokol yang digunakan antara rekan-rekan sebagai bagian dari RTCPeer Koneksi dapat berupa berbasis TCP atau UDP.

Sebagian besar aplikasi mencoba untuk membangun peer-to-peer koneksi langsung dengan menentukan alamat IP untuk setiap rekan, serta port dan protokol yang akan dipertukarkan sebagai kandidat ICE. Kandidat ini digunakan untuk mencoba terhubung satu sama lain menggunakan kandidat ini. Mereka akan mencoba setiap pasangan sampai koneksi dapat dibuat.

Lingkungan jaringan

Jika pesan dari penampil telah dikirim ke master dan log seperti `No valid ICE candidate` direkam, maka tidak ada rute koneksi yang valid telah ditemukan. Ini bisa terjadi jika ada firewall yang mencegah koneksi langsung, atau jika jaringan tidak dapat dijangkau.

Lakukan hal berikut untuk memecahkan masalah konektivitas:

- Jika Anda tidak menggunakan TURN di sisi master, pastikan untuk mengaktifkanTURN.

  TURNdiaktifkan secara default di C SDK. Di JavaScript SDK, pilih `STUN/TURN` atau `TURN only` di NAT Traversal.
- Untuk jaringan terbatas, seperti jaringan perusahaan, coba jaringan lain yang tersedia (kabel atau nirkabel).

  Jika Anda terhubung ke VPN, putuskan sambungan darinya.

> **Note**
>
> Anda dapat mengabaikan 403 `Forbidden IP` kesalahan yang dikembalikan oleh Kinesis Video Streams TURN server. Server menolak pasangan kandidat ICE yang berisi `localhost` IPs, seperti `192.168.*` atau`10.0.0.*`.
> Ini dapat menyebabkan beberapa, tetapi tidak semua, pasangan kandidat ICE gagal.

Debugging koneksi yang sedang berlangsung

Ada beberapa area yang dapat menyebabkan masalah dengan koneksi WebRTC yang mapan dan berkelanjutan, tetapi jaringan adalah yang paling umum.

Anda dapat mengonfirmasi log level VERBOSE dari SDK dengan menyetelnya `export AWS_KVS_LOG_LEVEL=1` sebagai variabel lingkungan.

> **Note**
>
> Jika tidak ada pasangan kandidat yang ditemukan dalam batas waktu yang ditentukan, agen ICE mengembalikan status kesalahan. `0x5a00000d`

Untuk informasi tambahan tentang tingkat log, lihat GitHub.

```
export AWS_KVS_LOG_LEVEL=1
./kvsWebrtcClientMasterGstSample TestChannel
```

Anda akan melihat log seperti berikut ini. Dari log ini, Anda dapat mengonfirmasi kandidat Interactive Connectivity Establishment (ICE) (alamat IP dan port) dan pasangan kandidat yang dipilih.

```
2023-02-13 05:57:16 DEBUG   iceAgentReadyStateSetup(): Selected pair w1UdV9fE+_/
CuBel1pl, local candidate type: srflx. Round trip time 7 ms. Local candidate priority:
 1694498815, ice candidate pair priority: 7240977859836116990
2023-02-13 05:57:16 INFO    onConnectionStateChange(): New connection state 3
2023-02-13 05:57:16 DEBUG   rtcPeerConnectionGetMetrics(): ICE local candidate Stats
 requested at 16762678365731494
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate IP
 Address: XXX.XXX.XXX.XXX
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 type: srflx
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 port: 38563
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 priority: 1694498815
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 transport protocol: udp
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 relay protocol: N/A
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate Ice
 server source: stun.kinesisvideo.ap-northeast-1.amazonaws.com
2023-02-13 05:57:16 DEBUG   rtcPeerConnectionGetMetrics(): ICE remote candidate Stats
 requested at 16762678365732111
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate IP
 Address: XXX.XXX.XXX.XXX
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 type: srflx
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 port: 45867
2023-02-13 05:57:16 VERBOSE signalingClientGetCurrentState(): Signaling Client Get
 Current State
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 priority: 1685921535
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 transport protocol: udp
```

Amazon Kinesis Video Streams dengan kuota layanan WebRTC

Kinesis Video Streams dengan WebRTC memiliki kuota layanan berikut:

> **Important**
>
> Kuota layanan berikut adalah soft [s], yang dapat ditingkatkan, atau hard [h], yang tidak dapat ditingkatkan. Anda akan melihat [s] dan [h] di samping kuota layanan individual pada tabel di bawah ini.

> **Note**
>
> TPS adalah singkatan dari transaksi per detik.

Topik

Kuota layanan API bidang kontrol

Bagian berikut menjelaskan kuota layanan untuk bidang APIs kontrol.

API	Tingkat permintaan API maksimum per Akun AWS	Jumlah maksimum saluran per Akun AWS per Wilayah AWS	Permintaan API maksimum per saluran
CreateSignalingChannel	50 TPS [s]	• Untuk AS Timur (Virginia N.) (us-east-1) dan AS Barat (Oregon) (us-west-2) saja - 10.000	N/A

API	Tingkat permintaan API maksimum per Akun AWS	Jumlah maksimum saluran per Akun AWS per Wilayah AWS	Permintaan API maksimum per saluran
		• Semua Wilayah lain yang didukung - 5.000	
DeleteSignalingChannel	50 TPS [h]	N/A	5 TPS [h]
DescribeMediaStorageConfiguration	50 TPS [h]	N/A	5 TPS [h]
DescribeSignalingChannel	300 TPS [h]	N/A	5 TPS [h]
GetSignalingChannelEndpoint	300 TPS [h]	N/A	N/A
ListSignalingChannels	50 TPS [h]	N/A	N/A
ListTagsForResource	50 TPS [h]	N/A	5 TPS [h]
TagResource	50 TPS [h]	N/A	5 TPS [h]
UntagResource	50 TPS [h]	N/A	5 TPS [h]
UpdateMediaStorageConfiguration	10 TPS [h]	N/A	5 TPS [h]
UpdateSignalingChannel	50 TPS [h]	N/A	5 TPS [h]

Mensinyalkan kuota layanan API

Bagian berikut menjelaskan kuota layanan untuk komponen pensinyalan di Kinesis Video Streams dengan WebRTC. Untuk informasi selengkapnya, lihat Cara kerjanya.

API	Masa tenggang pesan GO_AWAY maksimum sebelum mengakhiri koneksi	Tingkat permintaan API maksimum per saluran	Jumlah maksimum koneksi bersamaan per saluran	Durasi koneksi maksimum	Periode batas waktu koneksi idle maksimum
ConnectAsMaster	1 menit (h)	3 TPS (h)	1 (h)	1 jam (h)	10 menit (h)
ConnectAsViewer	1 menit (h)	3 TPS (h)	10 (s)	1 jam (h)	10 menit (h)

API	Tingkat permintaan API maksimum	Ukuran payload pesan maksimum
SendAlexaOffertoMaster	• 5 TPS per saluran pensinyalan (h) • 100 TPS Wilayah AWS per Akun AWS Per	N/A
Kirim ICECandidate	20 TPS per WebSocket koneksi (h)	10k (h)
Kirim SDPAnswer	5 TPS per WebSocket koneksi (h)	10k (h)
Kirim SDPOffer	5 TPS per WebSocket koneksi (h)	10k (h)

TURN kuota layanan

Bagian berikut menjelaskan kuota layanan untuk Traversal Using Relay around NAT (TURN) komponen di Kinesis Video Streams dengan WebRTC. Untuk informasi selengkapnya, lihat Cara kerjanya.

API atau parameter	Nilai
GetIceServerConfig	• 5 TPS per saluran pensinyalan (h) • 100 TPS Wilayah AWS per Akun AWS Per
Tingkat Bit	5Mbps (h)
Siklus Hidup Kredensi	5 menit (h)
Jumlah alokasi	50 per saluran pensinyalan (h)

WebRTC kuota layanan konsumsi

Bagian berikut menjelaskan kuota layanan untuk komponen perekaman media di Amazon Kinesis Video Streams WebRTC. Untuk informasi selengkapnya, lihat Gunakan Amazon Kinesis Video Streams dengan WebRTC untuk menelan dan menyimpan media.

JoinStorageSession

- API:
  - Per Akun AWS - 50 TPS (h)
  - Per saluran - 2 TPS (h)
- Kuota sesi streaming:
  - Kecepatan bit - 1 Mbps
  - Durasi sesi - 1 jam (h)
  - Batas waktu idle - 3 menit (h)

JoinStorageSessionAsViewer

- API:
  - Per Akun AWS - 50 TPS (h)
  - Per saluran - 2 TPS (h)
- Kuota sesi streaming:
  - Klien bersamaan maksimum dalam satu sesi - 3 hitungan (h)
  - Durasi sesi - 1 jam (h)

- Batas waktu idle - 1 menit (h)

Pemecahan masalah

Anda hanya dapat menghubungkan satu master dan satu atau lebih pemirsa ke satu saluran pensinyalan.

Tidak mungkin menghubungkan beberapa master ke satu saluran pensinyalan.

Akses Kinesis Video Streams dengan WebRTC

Anda dapat bekerja dengan Kinesis Video Streams dengan WebRTC dengan salah satu cara berikut:

Konsol Manajemen AWS

Memulai dengan Konsol Manajemen AWS

Konsol adalah antarmuka berbasis browser untuk mengakses dan menggunakan AWS layanan, termasuk Kinesis Video Streams dengan WebRTC.

AWS SDKs

AWS menyediakan kit pengembangan perangkat lunak (SDKs) yang terdiri dari pustaka dan kode sampel untuk berbagai bahasa dan platform pemrograman (misalnya, Java, Python, Ruby, .NET, iOS, Android, dan lainnya). SDKs Menyediakan cara mudah untuk membuat akses terprogram ke Kinesis Video Streams dengan WebRTC. Untuk informasi tentang AWS SDKs, termasuk cara mengunduh dan menginstalnya, lihat Alat untuk Amazon Web Services.

Kinesis Video Streams dengan WebRTC HTTPS API

Anda dapat mengakses Kinesis Video Streams dengan AWS WebRTC dan secara terprogram menggunakan Kinesis Video Streams dengan WebRTC, yang memungkinkan Anda mengeluarkan permintaan API langsung ke layanan APIs. Untuk informasi selengkapnya, lihat referensi API Amazon Kinesis Video Streams.

Mulai menggunakan

Bagian ini menjelaskan cara melakukan tugas-tugas berikut di Amazon Kinesis Video Streams dengan WebRTC:

- Siapkan Akun AWS dan buat administrator.
- Buat Kinesis Video Streams dengan saluran pensinyalan WebRTC.
- Siapkan Kinesis Video Streams dengan WebRTC pada perangkat keras Ingenic T31.

Jika Anda baru mengenal Kinesis Video Streams dengan WebRTC, kami sarankan Anda membaca terlebih dahulu. Cara kerjanya

Mengatur sebuah Akun AWS

Sebelum Anda menggunakan Kinesis Video Streams dengan WebRTC untuk pertama kalinya, selesaikan tugas-tugas berikut:

Topik

- Mendaftar untuk Akun AWS
- Buat pengguna dengan akses administratif
- Buat kunci AWS akun

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai

praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke Konsol Manajemen AWSsebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Buat kunci AWS akun

Anda memerlukan kunci AWS akun untuk mengakses Kinesis Video Streams dengan WebRTC secara terprogram.

Untuk membuat kunci AWS akun, lakukan hal berikut:

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Pengguna di bilah navigasi, dan pilih pengguna Administrator.
3. Pilih tab Security credentials, dan pilih Create access key.
4. Rekam ID kunci Akses. Pilih Tampilkan di bawah Kunci akses Rahasia, lalu rekam kunci akses Rahasia.

Buat saluran pensinyalan

Kinesis Video Streams dengan saluran pensinyalan WebRTC memfasilitasi pertukaran pesan pensinyalan yang diperlukan untuk membangun dan memelihara koneksi antara klien WebRTC. peer-to-peer Ini menangani negosiasi penawaran dan jawaban Session Description Protocol (SDP)

untuk parameter sesi, serta pertukaran kandidat Interactive Connectivity Establishment (ICE) untuk informasi jaringan.

Untuk membuat saluran pensinyalan, panggil CreateSignalingChannelAPI. Halaman ini akan menunjukkan cara menjalankan API itu menggunakan Konsol Manajemen AWS, AWS CLI, dan salah satunya. AWS SDKs

> **Important**
>
> Catat saluran ARN, Anda akan membutuhkannya nanti.

Konsol Manajemen AWS

Lakukan hal-hal berikut:

1. Buka konsol Saluran Sinyal Kinesis Video Streams di rumah/#/SignalingChannels. https://console.aws.amazon.com/kinesisvideo/
2. Pilih Buat saluran pensinyalan.
3. Pada halaman Buat saluran pensinyalan baru, ketikkan nama untuk saluran pensinyalan.

   Biarkan nilai default Time-to-live (Ttl) sebagai 60 detik.

   Pilih Buat saluran pensinyalan.
4. Setelah saluran pensinyalan dibuat, tinjau detail di halaman detail saluran.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS Command Line Interface.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensil dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan perintah Create-Signaling-Channel berikut menggunakan: AWS CLI

```
aws kinesisvideo create-signaling-channel \
```

```
  --channel-name "YourChannelName" \
  --region "us-west-2"
```

Responsnya akan terlihat seperti berikut:

```
{
    "ChannelARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123"
}
```

AWS SDK

Cuplikan kode ini menunjukkan cara membuat Kinesis Video Streams dengan saluran pensinyalan WebRTC menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap di GitHub.

Buat klien Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil `CreateSignalingChannel` API.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `CreateSignalingChannel` API.

```
const createSignalingChannelResponse = await kinesisVideoClient
    .createSignalingChannel({
        ChannelName: 'YourChannelName',
    })
    .promise();
```

Cetak responsnya.

```
console.log(createSignalingChannelResponse.ChannelARN);
```

Halaman web langsung dengan contoh kode ini tersedia untuk digunakan di GitHub. Masukkan wilayah, AWS kredensil, dan nama saluran pensinyalan Anda.

Pilih Buat Saluran.

Integrasikan dengan Ingenic T31

> **Note**
>
> AWS tidak mendukung chipset ini dengan cara apa pun, kami juga tidak menjamin integrasi akan berfungsi. Panduan ini didasarkan pada pengujian oleh tim Amazon Kinesis Video Streams dan disediakan untuk membantu pelanggan dalam menyiapkan perangkat mereka.

Ikuti prosedur ini untuk menyiapkan Amazon Kinesis Video Streams dengan WebRTC pada perangkat keras Ingenic T31.

Unduh kodenya

1. Siapkan direktori build.

   Untuk tutorial ini, buat direktori yang disebut `ingenic` di dalam `Downloads` folder.

   ```
   mkdir ~/Downloads/ingenic
   cd ~/Downloads/ingenic
   ```

2. Kloning repo berikut ke direktori baru Anda: https://github.com/aws-samples/ amazon-kinesis-video-streams -media-interface.

   ```
   git clone https://github.com/aws-samples/amazon-kinesis-video-streams-media-interface.git
   ```

Siapkan lingkungan build

1. Dapatkan gambar docker pra-bangun yang berisi rantai alat dan letakkan di `ingenic` folder.

   > **Important**
   >
   > Rantai alat mungkin berbeda berdasarkan jenis papan dan CPU. Periksa dengan vendor untuk rantai alat yang benar.

2. Muat gambar `Ingenic-T31-app-build.tar.gz` docker.

```
docker load --input ./Ingenic-T31-app-build.tar.gz
```

Keluaran yang diharapkan

```
580272b5675c: Loading layer [==================================================>]
 1.666GB/1.666GB
Loaded image ID:
 sha256:76d41ef9b2f53ad3f2a33f00ae110df3d1b491378a4005e19ea989ce97e99bc1
```

Catat`Image Id`. Anda akan membutuhkan ini di langkah berikutnya.

3. Jalankan perintah berikut untuk me-mount direktori saat ini (`~/Downloads/ingenic`) pada mesin host Anda ke `/ingenicMappedFolder` direktori di dalam wadah Docker.

Perintah ini juga menjalankan wadah dalam mode interaktif dengan sesi terminal. Anda akan dijatuhkan ke dalam shell Bash di dalam wadah, yang memungkinkan Anda untuk menjalankan perintah secara langsung.

```
docker run \
   -v `pwd`:/ingenicMappedFolder \
   -it 76d41ef9b2f53ad3f2a33f00ae110df3d1b491378a4005e19ea989ce97e99bc1 \
   /bin/bash
```

4. Tinjau isi `/ingenicMappedFolder` folder untuk mengonfirmasi bahwa volume bind-mount berhasil.

Prompt:

```
ls /ingenicMappedFolder
```

Tanggapan:

```
AWS-Solution-Remote-Diagnostic-Media-Interface Ingenic-T31-app-build.tar.gz
```

5. Verifikasi bahwa wadah docker memiliki variabel lingkungan yang disiapkan:

Prompt:

```
env
```

Tanggapan:

```
CC=mips-linux-gnu-gcc
CXX=mips-linux-gnu-g++
PATH=/mips-gcc540-glibc222-64bit-r3.3.0/bin:/usr/local/sbin:/usr/local/bin:/usr/
sbin:/usr/bin:/sbin:/bin
```

6. Verifikasi bahwa wadah docker berisi rantai alat dan Ingenic SDK. Itu harus ditempatkan di/`ingenic-sdk`.

   Prompt:

   ```
   ls /
   ```

   Tanggapan:

   ```
   bin   ingenic-sdk          libx32                              proc  sys
   boot  ingenicMappedFolder  media                               root  tmp
   dev   lib                  mips-gcc540-glibc222-64bit-r3.3.0  run   usr
   etc   lib32                mnt                                 sbin  var
   home  lib64                opt                                 srv
   ```

Bangun Amazon Kinesis Video Streams dengan aplikasi WebRTC

1. Ketik berikut ini:

   ```
   cd /ingenicMappedFolder/AWS-Solution-Remote-Diagnostic-Media-Interface
   cp -r /ingenic-sdk/* \
       /ingenicMappedFolder/amazon-kinesis-video-streams-media-interface/3rdparty/T31/
   export LDFLAGS=-Wl,--dynamic-linker=/lib/ld.so.1
   mkdir build
   cmake -B ./build -DBOARD=T31 -DCMAKE_BUILD_TYPE=Release -DBUILD_WEBRTC_SAMPLES=ON \
       -DBUILD_KVS_SAMPLES=ON -DBUILD_SAVE_FRAME_SAMPLES=ON
   cmake --build ./build --config Release
   ```

`kvswebrtcmaster-static`Aplikasi ini terletak di`/ingenicMappedFolder/AWS-Solution-Remote-Diagnostic-Media-Interface/build/samples/webrtc/`.

2. Jalankan perintah berikut untuk keluar dari wadah docker:

```
exit
```

3. Pada mesin host Anda, verifikasi bahwa `kvswebrtcmaster-static` aplikasi ada.

```
ls ~/Downloads/ingenic/AWS-Solution-Remote-Diagnostic-Media-Interface/build/
samples/webrtc/kvswebrtcmaster-static
```

Unggah aplikasi ke perangkat

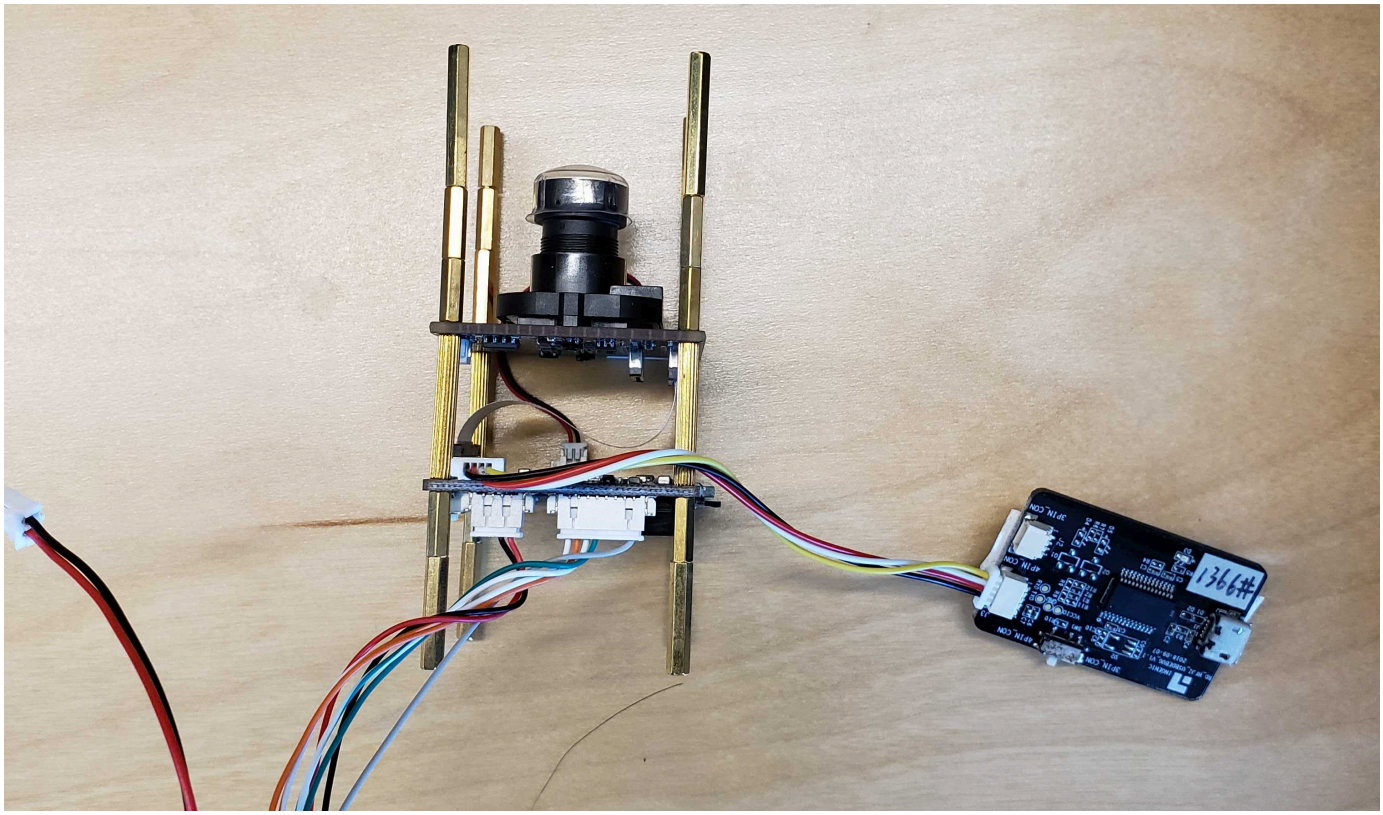
1. Pada mesin host Anda, salin biner statis ke Kartu micro SD.

```
cp ~/Downloads/ingenic/AWS-Solution-Remote-Diagnostic-Media-Interface/build/
samples/webrtc/kvswebrtcmaster-static /Volumes/IngenicSDCard
```

2. Unduh file.pem ini dan letakkan di kartu micro SD sebagai file. `cert.pem`
3. Masukkan kartu micro SD ke dalam slot kartu micro SD.

Connect ke perangkat

1. Pasang alat port serial ke papan, seperti yang ditunjukkan di bawah ini.



2. Colokkan ethernet dan kabel daya ke perangkat. LED merah harus menyala.
3. Hubungkan mesin host Anda ke slot micro-usb pada alat port serial.
4. Periksa perangkat yang terhubung:

```
ls /dev/tty.*
```

> **Note**
>
> Jika Anda memiliki beberapa perangkat TTY, tentukan perangkat TTY mana yang merupakan papan Ingenic. Putuskan sambungan perangkat dari mesin host Anda dan jalankan `ls` lagi. Bandingkan output perintah dan temukan perbedaannya.

5. Connect ke sana menggunakan `screen` atau alat port serial lainnya (misalnya, Tera Term). Atur baud rate ke 115200.

```
screen /dev/tty.usbserial-XXXXXXX 115200
```

6. Tentukan tindakan yang sesuai, berdasarkan keadaan dewan Anda:

- Jika sesi shell berakhir dengan a#, boot terputus. Gunakan `boot` perintah untuk memulai OS Linux, lalu lanjutkan dengan sisa langkah ini.
- Jika sesi shell meminta Anda untuk masuk, ketik kata sandi Anda.
- Jika layar kosong, tekan Enter.
- Jika layar menunjukkan kesalahan`Cannot exec '/dev/tty.usbserial-XXXXXXX': No such file or directory`, periksa kembali semua koneksi fisik antara papan dan mesin host Anda.

Pasang kartu micro SD di papan tulis

1. Buat direktori:

   > **Note**
   >
   > Untuk contoh ini, kita menggunakan`sdcard`.

   ```
   mkdir /tmp/sdcard
   ```

2. Ketik berikut ini untuk memasang kartu micro SD ke folder:

   ```
   mount /dev/mmcblk0p1 /tmp/sdcard
   ```

3. Tinjau isi folder:

   ```
   ls /tmp/sdcard
   ```

Jalankan aplikasi

1. Arahkan ke direktori yang dipasang:

   ```
   cd /tmp/sdcard
   ```

2. Ekspor kredensi Anda dan informasi lainnya dari papan:

   ```
   export AWS_ACCESS_KEY_ID=ID
   ```

```
export AWS_SECRET_ACCESS_KEY=key
export AWS_DEFAULT_REGION=us-west-2
export AWS_KVS_CACERT_PATH=`pwd`/cert.pem
```

3. Jalankan aplikasi:

```
./kvswebrtcmaster-static channel-name
```

Lihat media

Untuk melihat media, sambungkan ke saluran pensinyalan sebagai penampil. Lihat bagian berikut untuk contoh:

- JavaScript
- Konsol Manajemen AWS
- Android
- iOS

Pemecahan masalah

Bagian ini berisi pertanyaan dan masalah umum yang kami temui.

Ketika saya menghubungkan papan ke mesin host saya dan menjalankannya**`ls /dev/tty.*`**, perangkat tidak muncul.

Verifikasi koneksi dan semua kabel diamankan, dan perangkat dihidupkan. Periksa apakah ada indikator LED pada USB-to-TTY perangkat Anda menyala. Jika Anda masih mengalami masalah, hubungi vendor untuk mendiagnosis lebih lanjut masalah yang terhubung ke papan.

Aplikasi gagal terhubung ke pensinyalan

Jika menerima salah satu kesalahan berikut:

```
SSL error: certificate is not yet valid
```

atau

```
2024-09-19 08:56:34.920 WARN    lwsHttpCallbackRoutine(): Received client http read
 response:  { "message": "Signature expired: 20240919T085634Z is now earlier than
 20240919T155135Z (20240919T155635Z - 5 min.)" }
```

Ini berarti waktunya tidak tepat di papan Ingenic Anda. Verifikasi ini dengan menjalankan perintah tanggal. Atur waktu sesuai dengan instruksi vendor, lalu jalankan aplikasi lagi.

Aplikasi gagal untuk memulai, meskipun file ada di sana

Jika kesalahannya terlihat seperti:

```
-sh ./kvswebrtcmaster-static: not found
```

Ini berarti bahwa sistem operasi tidak dapat menemukan ketergantungan yang tercantum dalam header ELF aplikasi. Misalnya, `libc` pada perangkat.

Biasanya, ini menunjukkan bahwa toolchain yang digunakan tidak kompatibel dengan board, atau bahwa flag linker (`LDFLAGS`) tidak menunjuk dengan benar ke jalur pustaka yang diperlukan, yang mengarah ke dependensi yang belum terselesaikan selama runtime.

Pastikan rantai alat yang benar (`uclibc`vs `glibc` dan versinya) cocok dengan perangkat lunak di papan tulis dan atur `LDFLAGS` sesuai the section called “Membangun dengan aplikasi WebRTC” dengan.

Bagaimana cara melepas kartu micro SD?

Untuk melepas kartu micro SD, gunakan perintah:

```
umount /tmp/sdcard
```

Streaming media langsung (SDKs)

Di Amazon Kinesis Video Streams WebRTC, peer adalah perangkat yang dikonfigurasi untuk streaming dua arah real-time melalui saluran pensinyalan. Amazon Kinesis Video Streams SDKs dengan easy-to-use WebRTC adalah pustaka perangkat lunak yang dapat Anda unduh dan instal pada perangkat dan klien aplikasi yang ingin Anda konfigurasikan sebagai rekan melalui saluran pensinyalan tertentu.

Amazon Kinesis Video Streams dengan WebRTC mencakup hal-hal berikut: SDKs

- Amazon Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan
- Amazon Kinesis Video Streams dengan WebRTC JavaScript SDK untuk aplikasi web
- Amazon Kinesis Video Streams WebRTC SDK untuk Android
- Amazon Kinesis Video Streams WebRTC SDK untuk iOS

Setiap SDK menyertakan sampel dan step-by-step instruksi terkait yang dapat membantu Anda membangun dan menjalankan aplikasi tersebut. Anda dapat menggunakan sampel ini untuk latensi rendah, live, streaming audio dan video dua arah dan pertukaran data antara kombinasi Web/Android/iOS aplikasi atau perangkat yang disematkan. Dengan kata lain, Anda dapat melakukan streaming audio dan video langsung dari perangkat kamera yang disematkan ke aplikasi Android atau web atau di antara dua aplikasi Android.

Amazon Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan

step-by-stepPetunjuk berikut menjelaskan cara mengunduh, membuat, dan menjalankan Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan dan sampel yang sesuai.

Codec berikut didukung:

- Audio:
  - G.711 A-Law
  - G.711 U-Hukum
  - Opus

- Video:
  - H.264
  - H.265
  - VP8

Unduh SDK

Untuk mengunduh Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan, jalankan perintah berikut:

```
$ git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-c.git
```

Membangun SDK

> **Important**
>
> Sebelum Anda menyelesaikan langkah-langkah ini di macOS dan tergantung pada versi macOS yang Anda miliki, Anda harus menjalankan `xcode-select --install` untuk mengunduh paket dengan alat dan header baris perintah. Kemudian buka `/Library/Developer/CommandLineTools/Packages/macOS_SDK_headers_for_macOS_10.14.pkg` dan ikuti penginstal untuk menginstal alat baris perintah dan header. Anda hanya perlu melakukan ini sekali dan sebelum memanggil`cmake`. Jika Anda sudah menginstal alat baris perintah dan header, Anda tidak perlu menjalankan perintah ini lagi.

Selesaikan langkah-langkah berikut:

1. Instal cmake:
   - Di macOS, jalankan `brew install cmake pkg-config srtp`
   - di Ubuntu, jalankan `sudo apt-get install pkg-config cmake libcap2 libcap-dev`
2. Dapatkan kunci akses dan kunci rahasia Akun AWS yang ingin Anda gunakan untuk demo ini.
3. Jalankan perintah berikut untuk membuat `build` direktori di WebRTC C SDK yang diunduh, dan jalankan darinya: `cmake`

```
$ mkdir -p amazon-kinesis-video-streams-webrtc-sdk-c/build; cd amazon-kinesis-
video-streams-webrtc-sdk-c/build; cmake ..
```

4. Sekarang Anda berada di `build` direktori yang baru saja Anda buat dengan langkah di atas, jalankan `make` untuk membangun WebRTC C SDK dan sampel yang disediakan.

> **Note**
>
> Tidak `kvsWebrtcClientMasterGstSample` akan dibangun jika sistem tidak `gstreamer` diinstal. Untuk memastikannya dibangun (di macOS), Anda harus menjalankan: `brew install gstreamer gst-plugins-base gst-plugins-good`

Jalankan sampel SDK

Setelah Anda menyelesaikan prosedur di atas, Anda berakhir dengan contoh aplikasi berikut di `build` direktori Anda:

- `kvsWebrtcClientMaster`- Aplikasi ini mengirimkan sampel frame H264/Opus (jalur:/samples/h264SampleFrames and /samples/opusSampleFrames) melalui saluran pensinyalan. Ini juga menerima audio yang masuk, jika diaktifkan di browser. Ketika diperiksa di browser, itu mencetak metadata dari paket audio yang diterima di terminal Anda.
- `kvsWebrtcClientViewer`- Aplikasi ini menerima sampel bingkai H264/Opus dan mencetaknya.
- `kvsWebrtcClientMasterGstSample`- Aplikasi ini mengirimkan sampel frame H264/Opus dari pipa. GStreamer

Untuk menjalankan salah satu sampel ini, selesaikan langkah-langkah berikut:

1. Siapkan lingkungan Anda dengan Akun AWS kredensil Anda:

```
export AWS_ACCESS_KEY_ID=YourAccessKey
export AWS_SECRET_ACCESS_KEY=YourSecretKey
export AWS_DEFAULT_REGION=YourAWSRegion
```

   Jika Anda menggunakan AWS kredensil sementara, ekspor juga token sesi Anda:

```
export AWS_SESSION_TOKEN=YourSessionToken
```

Jika Anda memiliki jalur sertifikat CA kustom untuk disetel, Anda dapat mengaturnya menggunakan:

```
export AWS_KVS_CACERT_PATH=../certs/cert.pem
```

> **Note**
>
> Secara default, sertifikat SSL CA diatur ke.. /certs/cert.pem yang menunjuk ke file dalam repositori ini di. GitHub

2. Jalankan salah satu aplikasi sampel dengan meneruskan nama yang ingin Anda berikan ke saluran pensinyalan Anda. Aplikasi membuat saluran pensinyalan menggunakan nama yang Anda berikan. Misalnya, untuk membuat saluran pensinyalan yang dipanggil `myChannel` dan untuk mulai mengirim sampel frame H264/Opus melalui saluran ini, jalankan perintah berikut:

```
./kvsWebrtcClientMaster myChannel
```

Ketika aplikasi baris perintah mencetak`Connection established`, Anda dapat melanjutkan ke langkah berikutnya.

3. Sekarang saluran pensinyalan Anda dibuat dan master yang terhubung sedang mengalirkan media ke sana, Anda dapat melihat aliran ini. Misalnya, Anda dapat melihat streaming langsung ini di aplikasi web. Untuk melakukannya, buka WebRTC SDK Test Page menggunakan langkah-langkah Gunakan aplikasi sampel dan atur nilai berikut menggunakan kredensi yang sama dan saluran pensinyalan AWS yang sama yang Anda tentukan untuk master di atas:

   - ID kunci akses
   - Kunci akses rahasia
   - Nama saluran pensinyalan
   - ID Klien (opsional)

   Pilih Mulai penampil untuk memulai streaming video langsung dari sampel bingkai H264/Opus.

Video tutorial

Video ini menunjukkan cara menghubungkan kamera Anda dan memulai dengan Amazon Kinesis Video Streams untuk WebRTC.

Amazon Kinesis Video Streams dengan WebRTC JavaScript SDK untuk aplikasi web

Anda dapat menemukan Kinesis Video Streams dengan JavaScript WebRTC SDK untuk aplikasi web dan sampel yang sesuai di. GitHub

Topik

- Instal SDK
- WebRTC SDK dokumentasi JavaScript
- Gunakan aplikasi sampel
- Edit aplikasi sampel

Instal SDK

Apakah dan bagaimana Anda menginstal Kinesis Video Streams dengan JavaScript WebRTC SDK tergantung pada apakah kode dijalankan dalam modul atau skrip browser. `Node.js`

NodeJS module

Cara yang lebih disukai untuk menginstal Kinesis Video Streams dengan JavaScript WebRTC SDK untuk Node.js adalah dengan menggunakan npm, pengelola paket Node.js.

Paket ini di-host di https://www.npmjs.com/package/amazon-kinesis-video-streams-webrtc.

Untuk menginstal SDK ini di `Node.js` project Anda, gunakan terminal untuk menavigasi ke direktori yang sama dengan project Anda: `package.json`

Ketik berikut ini:

```
npm install amazon-kinesis-video-streams-webrtc
```

Anda dapat mengimpor kelas SDK seperti modul Node.js biasa:

```
// JavaScript
const SignalingClient = require('amazon-kinesis-video-streams-
webrtc').SignalingClient;
// TypeScript
import { SignalingClient } from 'amazon-kinesis-video-streams-webrtc';
```

Browser

Anda tidak perlu menginstal SDK untuk menggunakannya dalam skrip browser. Anda dapat memuat paket SDK yang dihosting langsung AWS dengan skrip di halaman HTML Anda.

Untuk menggunakan SDK di browser, tambahkan elemen skrip berikut ke halaman HTML Anda:

```
<script src="https://unpkg.com/amazon-kinesis-video-streams-webrtc/dist/kvs-
webrtc.min.js"></script>
```

Setelah SDK dimuat di halaman Anda, SDK tersedia dari variabel global `KVSWebRTC` (atau`window.KVSWebRTC`).

Misalnya, `window.KVSWebRTC.SignalingClient`.

WebRTC SDK dokumentasi JavaScript

Dokumentasi untuk metode SDK ada di GitHub readme, di bawah Dokumentasi.

Di bagian Penggunaan, ada informasi tambahan untuk mengintegrasikan SDK ini bersama dengan AWS SDK JavaScript untuk membangun aplikasi penampil berbasis web.

Lihat `examples` direktori untuk contoh aplikasi lengkap, termasuk peran master dan penampil.

Gunakan aplikasi sampel

Kinesis Video Streams dengan WebRTC juga menghosting aplikasi sampel yang dapat Anda gunakan untuk membuat saluran pensinyalan baru atau terhubung ke saluran yang ada dan menggunakannya sebagai master atau penampil.

Kinesis Video Streams dengan aplikasi sampel WebRTC terletak di. GitHub

Kode untuk aplikasi sampel ada di `examples` direktori.

Topik

Streaming peer-to-peer dari aplikasi sampel ke Konsol Manajemen AWS

1. Buka Kinesis Video Streams dengan aplikasi sampel WebRTC dan lengkapi yang berikut ini:

   - Wilayah AWS. Misalnya,`us-west-2`.
   - Kunci AWS akses dan kunci rahasia untuk pengguna atau peran IAM Anda. Biarkan token sesi kosong jika Anda menggunakan AWS kredensi jangka panjang.
   - Nama saluran pensinyalan yang ingin Anda sambungkan.

     Jika Anda ingin terhubung ke saluran pensinyalan baru, pilih Buat Saluran untuk membuat saluran pensinyalan dengan nilai yang disediakan di kotak.

     > **Note**
     >
     > Nama saluran pensinyalan Anda harus unik untuk akun dan wilayah saat ini. Anda dapat menggunakan huruf, angka, garis bawah (_), dan tanda hubung (-), tetapi bukan spasi.

   - Apakah Anda ingin mengirim audio, video, atau keduanya.
   - WebRTC Tertelan dan Penyimpanan. Perluas node dan pilih salah satu dari berikut ini:
     - Pilih Secara otomatis menentukan mode konsumsi.
     - Pastikan Secara otomatis menentukan mode konsumsi tidak dipilih dan atur penggantian manual ke OFF.

     > **Note**
     >
     > Secara otomatis menentukan mode konsumsi memiliki aplikasi memanggil DescribeMediaStorageConfigurationAPI untuk menentukan mode mana yang akan dijalankan (atau konsumsi Peer-to-peer WebRTC). Panggilan API tambahan ini menambahkan sedikit waktu startup.

Jika Anda tahu sebelumnya mode mana saluran pensinyalan ini berjalan, gunakan penggantian manual untuk melewati panggilan API ini.

- Generasi kandidat ICE. BiarkanSTUN/TURNdipilih dan biarkan `Trickle ICE` diaktifkan.

2. Pilih Mulai Master untuk terhubung ke saluran pensinyalan.

   Izinkan akses ke and/or mikrofon kamera Anda, jika diperlukan.

3. Buka konsol Kinesis Video Streams di. Konsol Manajemen AWS

   Pastikan wilayah yang benar dipilih.

4. Di navigasi kiri, pilih saluran pensinyalan.

   Pilih nama saluran pensinyalan di atas. Gunakan bilah pencarian, jika perlu.

5. Perluas bagian Penampil pemutaran media.

6. Pilih tombol putar pada pemutar video. Ini bergabung dengan sesi WebRTC sebagai. `viewer`

   Media yang sedang dikirim pada halaman demo harus ditampilkan di Konsol Manajemen AWS.

Streaming peer-to-peer dari aplikasi sampel ke aplikasi sampel

1. Buka Kinesis Video Streams dengan aplikasi sampel WebRTC dan lengkapi informasi berikut:

   - Wilayah AWS. Misalnya,`us-west-2`.
   - Kunci AWS akses dan kunci rahasia untuk pengguna atau peran IAM Anda. Biarkan token sesi kosong jika Anda menggunakan AWS kredensi jangka panjang.
   - Nama saluran pensinyalan yang ingin Anda sambungkan.

     Jika Anda ingin terhubung ke saluran pensinyalan baru, pilih Buat Saluran untuk membuat saluran pensinyalan dengan nilai yang disediakan di kotak.

     > **Note**
     >
     > Nama saluran pensinyalan Anda harus unik untuk akun dan wilayah saat ini. Anda dapat menggunakan huruf, angka, garis bawah (_), dan tanda hubung (-), tetapi bukan spasi.

   - Apakah Anda ingin mengirim audio, video, atau keduanya.

- WebRTC Tertelan dan Penyimpanan. Perluas node dan pilih salah satu dari berikut ini:
  - Pilih Secara otomatis menentukan mode konsumsi.
  - Pastikan Secara otomatis menentukan mode konsumsi tidak dipilih dan atur penggantian manual ke OFF.

> **Note**
>
> Secara otomatis menentukan mode konsumsi memiliki aplikasi memanggil DescribeMediaStorageConfigurationAPI untuk menentukan mode mana yang akan dijalankan (atau konsumsi Peer-to-peer WebRTC). Panggilan API tambahan ini menambahkan sedikit waktu startup.
> Jika Anda tahu sebelumnya mode mana saluran pensinyalan ini berjalan, gunakan penggantian manual untuk melewati panggilan API ini.

- Generasi kandidat ICE. BiarkanSTUN/TURNdipilih dan biarkan `Trickle ICE` diaktifkan.

2. Pilih Mulai Master untuk terhubung ke saluran pensinyalan sebagai `master` peran.

   Izinkan akses ke and/or mikrofon kamera Anda, jika diperlukan.

3. Buka tab browser lain dan buka Kinesis Video Streams dengan aplikasi contoh WebRTC. Semua informasi dari proses sebelumnya harus dimuat.
4. Gulir ke bawah dan pilih Mulai Penampil untuk terhubung ke saluran pensinyalan sebagai `viewer` peran.

   Anda harus melihat media dipertukarkan antara `master` dan`viewer`.

Streaming peer-to-peer dengan WebRTC Ingestion dari halaman sampel ke halaman sampel

1. Ikuti the section called “Menelan media dari browser” untuk menghubungkan peserta utama dan pastikan itu terhubung ke sesi penyimpanan.
2. Ikuti the section called “Tambahkan pemirsa ke sesi konsumsi” untuk menambahkan peserta pemirsa.

   Peserta pemirsa akan terhubung dan menerima media dari sesi penyimpanan. Mereka dapat mengirim audio opsional kembali ke sesi penyimpanan.

Sesi penyimpanan menangani pencampuran media yang diterima dari peserta master dan pemirsa dan mengirimkannya ke tujuan yang sesuai.

3. Anda dapat melihat dan menggunakan media yang dicerna melalui pemutaran Kinesis Video Streams.

Edit aplikasi sampel

Untuk mengedit SDK dan contoh aplikasi untuk tujuan pengembangan, ikuti petunjuk di bawah ini.

Prasyarat

NodeJS versi 16+

> **Note**
>
> Kami merekomendasikan mengunduh versi dukungan jangka panjang (LTS) terbaru dari https://nodejs. org/en/download.

Edit aplikasi sampel

1. Unduh Kinesis Video Streams dengan JavaScript WebRTC SDK di.

   Ketik berikut ini di terminal:

   ```
   git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-js.git
   ```

2. Arahkan ke direktori dengan file package.json. File ini terletak di direktori root repositori.

   Ketik berikut ini di terminal:

   ```
   cd amazon-kinesis-video-streams-webrtc-sdk-js
   ```

3. Instal dependensi.

   Ketik perintah CLI npm berikut di terminal:

   ```
   npm install
   ```

4. Mulai server web untuk mulai melayani halaman web.

Ketik perintah CLI npm berikut di terminal:

```
npm run develop
```

5. Di browser Anda, kunjungi http://localhost:3001/.

   Anda dapat melakukan pengeditan ke halaman web dengan mengedit file di `examples` direktori.

Amazon Kinesis Video Streams WebRTC SDK untuk Android

step-by-stepPetunjuk berikut menjelaskan cara mengunduh, membuat, dan menjalankan Kinesis Video Streams dengan WebRTC SDK for Android dan sampelnya yang sesuai.

> **Note**
>
> Amazon Kinesis Video Streams IPv6 tidak mendukung alamat di Android. Lihat informasi selengkapnya tentang menonaktifkan IPv6 di perangkat Android Anda.

Unduh SDK

Untuk mengunduh SDK WebRTC di Android, jalankan perintah berikut:

```
$ git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-android.git
```

Membangun SDK

Untuk membangun SDK WebRTC di Android, selesaikan langkah-langkah berikut:

1. Impor Android WebRTC SDK ke lingkungan pengembangan terintegrasi (IDE) Android Studio dengan membuka dengan Open as Project. **`amazon-kinesis-video-streams-webrtc-sdk-android/build.gradle`**
2. Jika Anda membuka proyek untuk pertama kalinya, secara otomatis akan disinkronkan. Jika tidak - memulai sinkronisasi. Ketika Anda melihat kesalahan build, pilih untuk menginstal apa pun yang diperlukan SDKs dengan memilih Instal paket SDK yang hilang, lalu pilih Terima dan selesaikan penginstalan.

3. Konfigurasikan setelan Amazon Cognito (kumpulan pengguna dan kumpulan identitas). Untuk langkah-langkah detail, lihatKonfigurasikan Amazon Cognito untuk SDK. Ini menghasilkan pengaturan autentikasi dan otorisasi yang diperlukan untuk membangun Android WebRTC SDK.
4. Di IDE Android Anda, buka `awsconfiguration.json` (dari`src/main/res/raw/`). File tersebut terlihat seperti berikut:

```
{
  "Version": "1.0",
  "CredentialsProvider": {
    "CognitoIdentity": {
      "Default": {
        "PoolId": "REPLACE_ME",
        "Region": "REPLACE_ME"
      }
    }
  },
  "IdentityManager": {
    "Default": {}
  },
  "CognitoUserPool": {
    "Default": {
      "AppClientSecret": "REPLACE_ME",
      "AppClientId": "REPLACE_ME",
      "PoolId": "REPLACE_ME",
      "Region": "REPLACE_ME"
    }
  }
}
```

   Perbarui `awsconfiguration.json` dengan nilai yang dihasilkan dengan menjalankan langkah-langkah diKonfigurasikan Amazon Cognito untuk SDK.
5. Pastikan perangkat Android Anda terhubung ke komputer tempat Anda menjalankan IDE Android. Di Android IDE, pilih perangkat yang terhubung lalu buat dan jalankan WebRTC Android SDK.

   Langkah ini menginstal aplikasi yang dipanggil `AWSKinesisVideoWebRTCDemoApp` di perangkat Android Anda. Dengan menggunakan aplikasi ini, Anda dapat memverifikasi streaming WebRTC langsung antara audio/video klien perangkat seluler, web, dan IoT.

Jalankan aplikasi sampel

Selesaikan langkah-langkah berikut:

1. Di perangkat Android Anda, buka AWSKinesisVideoWebRTCDemoAplikasi dan masuk menggunakan akun baru (dengan membuatnya terlebih dahulu) atau akun Amazon Cognito yang sudah ada.
2. Di AWSKinesisVideoWebRTCDemoApp, navigasikan ke halaman Konfigurasi Saluran dan buat saluran pensinyalan baru atau pilih yang sudah ada.

   > **Note**
   >
   > Saat ini, dengan menggunakan contoh aplikasi di SDK ini, Anda hanya dapat menjalankan satu saluran pensinyalan di AWSKinesis VideoWeb RTCDemo App.

3. Opsional: pilih ID Klien unik jika Anda ingin terhubung ke saluran ini sebagai penampil. ID klien hanya diperlukan jika beberapa pemirsa terhubung ke saluran. Ini membantu master saluran mengidentifikasi masing-masing pemirsa.
4. Pilih Wilayah AWS dan apakah Anda ingin mengirim data audio atau video, atau keduanya.
5. Untuk memverifikasi peer-to-peer streaming, lakukan salah satu hal berikut:

   > **Note**
   >
   > Pastikan Anda menentukan nama saluran pensinyalan, AWS wilayah, ID penampil, dan ID AWS akun yang sama pada semua klien yang Anda gunakan dalam demo ini.

   - Peer-to-peer streaming antara dua perangkat Android: master dan viewer
     - Menggunakan prosedur di atas, unduh, buat, dan jalankan Android WebRTC SDK di dua perangkat Android.
     - Buka AWSKinesisVideoWebRTCDemoAplikasi di satu perangkat Android dalam mode master (pilih MULAI MASTER) untuk memulai sesi baru (saluran pensinyalan).

       > **Note**
       >
       > Saat ini, hanya ada satu master untuk saluran pensinyalan tertentu.

    - Buka AWSKinesisVideoWebRTCDemoAplikasi di perangkat Android kedua Anda dalam mode penampil untuk terhubung ke saluran pensinyalan (sesi) yang dimulai pada langkah di atas (pilih MULAI PENAMPIL).

      Verifikasi bahwa pemirsa dapat melihat audio/video data master.
- Peer-to-peer streaming antara master SDK yang disematkan dan penampil perangkat Android
  - Unduh, buat, dan jalankan mode master Amazon Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan di perangkat kamera.
  - Menggunakan prosedur di atas, unduh, buat, dan jalankan Android WebRTC SDK di perangkat Android. Buka AWSKinesisVideoWebRTCDemoAplikasi di perangkat Android ini dalam mode penampil dan verifikasi bahwa penampil dapat melihat audio/video data master SDK yang disematkan.
- Peer-to-peer streaming antara perangkat Android sebagai master dan browser web sebagai penampil
  - Menggunakan prosedur di atas, unduh, buat, dan jalankan Android WebRTC SDK di perangkat Android. Buka AWSKinesisVideoWebRTCDemoAplikasi di perangkat Android ini dalam mode master (pilih MULAI MASTER) untuk memulai sesi baru (saluran pensinyalan).
  - Unduh, buat, dan jalankan penampil Amazon Kinesis Video Streams dengan WebRTC JavaScript SDK untuk aplikasi web as dan verifikasi bahwa pemirsa dapat melihat audio/video master Android.

Konfigurasikan Amazon Cognito untuk SDK

Prasyarat

- Kami merekomendasikan Android Studio untuk memeriksa, mengedit, dan menjalankan kode aplikasi. Kami merekomendasikan menggunakan versi stabil terbaru.
- Dalam kode contoh, Anda memberikan kredensi Amazon Cognito.

Ikuti prosedur ini untuk menyiapkan kumpulan pengguna Amazon Cognito dan kumpulan identitas.

Siapkan kumpulan pengguna

Untuk mengatur kumpulan pengguna

1. Masuk ke konsol Amazon Cognito dan verifikasi wilayah tersebut benar.

2. Di navigasi di sebelah kiri pilih Kumpulan pengguna.
3. Di bagian User pool, pilih Create user pool.
4. Lengkapi bagian berikut:

   a. Langkah 1: Konfigurasikan pengalaman masuk - Di bagian opsi masuk kumpulan pengguna Cognito, pilih opsi yang sesuai.

      Pilih Selanjutnya.

   b. Langkah 2: Konfigurasikan persyaratan keamanan - Pilih opsi yang sesuai.

      Pilih Selanjutnya.

   c. Langkah 3: Konfigurasikan pengalaman pendaftaran - Pilih opsi yang sesuai.

      Pilih Selanjutnya.

   d. Langkah 4: Konfigurasikan pengiriman pesan - Pilih opsi yang sesuai.

      Di bidang pemilihan peran IAM, pilih peran yang ada atau buat peran baru.

      Pilih Selanjutnya.

   e. Langkah 5: Integrasikan aplikasi Anda - Pilih opsi yang sesuai.

      Di bidang Klien aplikasi awal, pilih Klien rahasia.

      Pilih Selanjutnya.

   f. Langkah 6: Tinjau dan buat - Tinjau pilihan Anda dari bagian sebelumnya, lalu pilih Buat kumpulan pengguna.

5. Pada halaman User pool, pilih pool yang baru saja Anda buat.

   Salin ID kumpulan Pengguna dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CognitoUserPool.Default.PoolId`.

6. Pilih tab Integrasi aplikasi dan pergi ke bagian bawah halaman.
7. Di bagian Daftar klien Aplikasi, pilih nama klien Aplikasi yang baru saja Anda buat.

   Salin ID Klien dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CognitoUserPool.Default.AppClientId`.

8. Tunjukkan rahasia Klien dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CognitoUserPool.Default.AppClientSecret`.

Siapkan kolam identitas

Untuk mengatur kumpulan identitas

1. Masuk ke konsol Amazon Cognito dan verifikasi wilayah tersebut benar.
2. Di navigasi di sebelah kiri pilih Identity pool.
3. Pilih Buat kumpulan identitas.
4. Konfigurasikan kumpulan identitas.

   a. Langkah 1: Konfigurasikan kepercayaan kumpulan identitas - Lengkapi bagian berikut:

      - Akses pengguna - Pilih Akses yang Diautentikasi
      - Sumber identitas yang diautentikasi - Pilih kumpulan pengguna Amazon Cognito

      Pilih Selanjutnya.

   b. Langkah 2: Konfigurasikan izin - Di bagian peran yang diautentikasi, lengkapi bidang berikut:

      - Peran IAM - Pilih Buat peran IAM baru
      - Nama peran IAM - Masukkan nama dan catat untuk langkah selanjutnya.

      Pilih Selanjutnya.

   c. Langkah 3: Hubungkan penyedia identitas - Di bagian Rincian kumpulan pengguna, lengkapi bidang berikut:

      - ID kumpulan pengguna - Pilih kumpulan pengguna yang Anda buat sebelumnya.
      - ID klien aplikasi - Pilih ID klien aplikasi yang Anda buat sebelumnya.

      Pilih Selanjutnya.

   d. Langkah 4: Konfigurasikan properti - Ketik nama di bidang Identity pool name.

      Pilih Selanjutnya.

   e. Langkah 5: Tinjau dan buat - Tinjau pilihan Anda di setiap bagian, lalu pilih Buat kumpulan identitas.

5. Pada halaman Identity pool, pilih kumpulan identitas baru Anda.

Salin ID kumpulan Identitas dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CredentialsProvider.CognitoIdentity.Default.PoolId`.

6. Perbarui izin untuk peran IAM.

   a. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   b. Di navigasi di sebelah kiri, pilih Peran.

   c. Temukan dan pilih peran yang Anda buat di atas.

   > **Note**
   >
   > Gunakan bilah pencarian, jika perlu.

   d. Pilih kebijakan izin terlampir.

   Pilih Edit.

   e. Pilih tab JSON dan ganti kebijakan dengan yang berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:*",
                "kinesisvideo:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

   Pilih Selanjutnya.

   f. Pilih kotak di samping Setel versi baru ini sebagai default jika belum dipilih.

   Pilih Simpan perubahan.

Amazon Kinesis Video Streams WebRTC SDK untuk iOS

step-by-stepPetunjuk berikut menjelaskan cara mengunduh, membuat, dan menjalankan SDK WebRTC Kinesis Video Streams di iOS dan sampelnya yang sesuai.

Unduh SDK

Untuk mengunduh SDK WebRTC di iOS, jalankan perintah berikut:

```
$ git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-ios.git
```

Membangun SDK

Selesaikan langkah-langkah berikut:

1. Impor SDK WebRTC iOS ke lingkungan pengembangan terintegrasi (IDE) XCode di komputer iOS dengan `KinesisVideoWebRTCDemoApp.xcworkspace` membuka (path: - /Swift/ App.xcWorkspace). amazon-kinesis-video-streams webrtc-sdk-ios AWSKinesis VideoWeb RTCDemo
2. Jika Anda membuka proyek untuk pertama kalinya, proyek akan dibangun secara otomatis. Jika tidak, mulailah membangun.

   Anda mungkin melihat kesalahan berikut:

   ```
   error: The sandbox is not in sync with the Podfile.lock. Run 'pod install' or
    update your CocoaPods installation.
   ```

   Jika Anda melihat ini, lakukan hal berikut:

   a. Ubah direktori kerja Anda saat ini menjadi `amazon-kinesis-video-streams-webrtc-sdk-ios/Swift` dan jalankan yang berikut di baris perintah:

      ```
      pod cache clean --all
      pod install
      ```

   b. Ubah direktori kerja Anda saat ini menjadi `amazon-kinesis-video-streams-webrtc-sdk-ios` dan jalankan yang berikut di baris perintah:

      ```
      $ git checkout Swift/Pods/AWSCore/AWSCore/Service/AWSService.m
      ```

c. `Build`lagi.

3. Konfigurasikan setelan Amazon Cognito (kumpulan pengguna dan kumpulan identitas). Untuk langkah-langkah detail, lihatKonfigurasikan Amazon Cognito untuk SDK. Ini menghasilkan pengaturan otentikasi dan otorisasi yang diperlukan untuk membangun SDK WebRTC iOS.

4. Di IDE Anda, buka `awsconfiguration.json` file (dari`/Swift/KVSiOSApp`). File tersebut terlihat seperti berikut:

```
{
    "Version": "1.0",
    "CredentialsProvider": {
        "CognitoIdentity": {
            "Default": {
                "PoolId": "REPLACEME",
                "Region": "REPLACEME"
            }
        }
    },
    "IdentityManager": {
        "Default": {}
    },
    "CognitoUserPool": {
        "Default": {
            "AppClientSecret": "REPLACEME",
            "AppClientId": "REPLACEME",
            "PoolId": "REPLACEME",
            "Region": "REPLACEME"
        }
    }
}
```

Perbarui `awsconfiguration.json` dengan nilai yang dihasilkan dengan menjalankan langkah-langkah diKonfigurasikan Amazon Cognito untuk SDK.

5. Di IDE Anda, buka `Constants.swift` file (dari`/Swift/KVSiOSApp`). File tersebut terlihat seperti berikut:

```
import Foundation
import AWSCognitoIdentityProvider

let CognitoIdentityUserPoolRegion = AWSRegionType.USWest2
let CognitoIdentityUserPoolId = "REPLACEME"
```

```
let CognitoIdentityUserPoolAppClientId = "REPLACEME"
let CognitoIdentityUserPoolAppClientSecret = "REPLACEME"

let AWSCognitoUserPoolsSignInProviderKey = "UserPool"
let CognitoIdentityPoolID = "REPLACEME"

let AWSKinesisVideoEndpoint = "https://kinesisvideo.us-west-2.amazonaws.com"
let AWSKinesisVideoKey = "kinesisvideo"

let VideoProtocols =  ["WSS", "HTTPS"]

let ConnectAsMaster = "connect-as-master"
let ConnectAsViewer = "connect-as-viewer"

let MasterRole = "MASTER"
let ViewerRole = "VIEWER"

let ClientID = "ConsumerViewer"
```

Perbarui `Constants.swift` dengan nilai yang dihasilkan dengan menjalankan langkah-langkah di[Konfigurasikan Amazon Cognito untuk SDK](#).

6. Pastikan perangkat iOS Anda terhubung ke komputer Mac tempat Anda menjalankan XCode. Di XCode, pilih perangkat yang terhubung lalu buat dan jalankan WebRTC iOS SDK.

   Langkah ini menginstal aplikasi yang dipanggil `AWSKinesisVideoWebRTCDemoApp` di perangkat iOS Anda. Dengan menggunakan aplikasi ini, Anda dapat memverifikasi streaming WebRTC langsung antara audio/video klien perangkat seluler, web, dan IoT.

Jalankan aplikasi sampel

Selesaikan langkah-langkah berikut:

1. Di perangkat iOS Anda, buka AWSKinesisVideoWebRTCDemoAplikasi dan masuk menggunakan akun baru (dengan membuatnya terlebih dahulu) atau akun Amazon Cognito yang sudah ada.
2. Di AWSKinesisVideoWebRTCDemoApp, navigasikan ke halaman Konfigurasi Saluran dan buat saluran pensinyalan baru atau pilih yang sudah ada.

> **Note**
>
> Saat ini, dengan menggunakan contoh aplikasi di SDK ini, Anda hanya dapat menjalankan satu saluran pensinyalan di AWSKinesis VideoWeb RTCDemo App.

3. (Opsional) Pilih ID Klien unik jika Anda ingin terhubung ke saluran ini sebagai penampil. ID Klien hanya diperlukan jika beberapa pemirsa terhubung ke saluran. Ini membantu master saluran mengidentifikasi masing-masing pemirsa.
4. Pilih Wilayah AWS dan apakah Anda ingin mengirim data audio atau video, atau keduanya.
5. Untuk memverifikasi peer-to-peer streaming, lakukan salah satu hal berikut:

> **Note**
>
> Pastikan Anda menentukan nama saluran pensinyalan, AWS wilayah, ID penampil, dan ID AWS akun yang sama pada semua klien yang Anda gunakan dalam demo ini.

- Peer-to-peer streaming antara dua perangkat iOS: master dan viewer
  - Menggunakan prosedur di atas, unduh, buat, dan jalankan SDK WebRTC iOS di dua perangkat iOS.
  - Buka AWSKinesisVideoWebRTCDemoAplikasi di satu perangkat iOS dalam mode master (pilih MULAI MASTER) untuk memulai sesi baru (saluran pensinyalan).

    > **Note**
    >
    > Saat ini, hanya ada satu master untuk saluran pensinyalan tertentu.

  - Buka AWSKinesisVideoWebRTCDemoAplikasi di perangkat iOS kedua Anda dalam mode penampil untuk terhubung ke saluran pensinyalan (sesi) yang dimulai pada langkah di atas (pilih MULAI PENAMPIL).

    Verifikasi bahwa pemirsa dapat melihat audio/video data master.
- Peer-to-peer streaming antara master SDK yang disematkan dan penampil perangkat iOS
  - Unduh, buat, dan jalankan mode master Amazon Kinesis Video Streams dengan WebRTC SDK di C untuk perangkat yang disematkan di perangkat kamera.

    - Menggunakan prosedur di atas, unduh, buat, dan jalankan SDK WebRTC iOS di perangkat iOS. Buka AWSKinesisVideoWebRTCDemoAplikasi di perangkat iOS ini dalam mode penampil dan verifikasi bahwa penampil iOS dapat melihat audio/video data master SDK yang disematkan.
- Peer-to-peer streaming antara perangkat iOS sebagai master dan browser web sebagai penampil
    - Menggunakan prosedur di atas, unduh, buat, dan jalankan SDK WebRTC iOS di perangkat iOS. Buka AWSKinesisVideoWebRTCDemoAplikasi di perangkat iOS ini dalam mode master (pilih MULAI MASTER) untuk memulai sesi baru (saluran pensinyalan).
    - Unduh, buat, dan jalankan penampil Amazon Kinesis Video Streams dengan WebRTC JavaScript SDK untuk aplikasi web as dan verifikasi bahwa JavaScript pemirsa dapat melihat audio/video master Android.

Konfigurasikan Amazon Cognito untuk SDK

Prasyarat

- Kami merekomendasikan XCode untuk memeriksa, mengedit, dan menjalankan kode aplikasi. Kami merekomendasikan versi terbaru.
- Dalam kode contoh, Anda memberikan kredensi Amazon Cognito.

Ikuti prosedur ini untuk menyiapkan kumpulan pengguna Amazon Cognito dan kumpulan identitas.

Siapkan kumpulan pengguna

Untuk mengatur kumpulan pengguna

1. Masuk ke konsol Amazon Cognito dan verifikasi wilayah tersebut benar.
2. Di navigasi di sebelah kiri pilih Kumpulan pengguna.
3. Di bagian User pool, pilih Create user pool.
4. Lengkapi bagian berikut:

    a. Langkah 1: Konfigurasikan pengalaman masuk - Di bagian opsi masuk kumpulan pengguna Cognito, pilih opsi yang sesuai.

    Pilih Selanjutnya.

b. Langkah 2: Konfigurasikan persyaratan keamanan - Pilih opsi yang sesuai.

   Pilih Selanjutnya.

c. Langkah 3: Konfigurasikan pengalaman pendaftaran - Pilih opsi yang sesuai.

   Pilih Selanjutnya.

d. Langkah 4: Konfigurasikan pengiriman pesan - Pilih opsi yang sesuai.

   Di bidang pemilihan peran IAM, pilih peran yang ada atau buat peran baru.

   Pilih Selanjutnya.

e. Langkah 5: Integrasikan aplikasi Anda - Pilih opsi yang sesuai.

   Di bidang Klien aplikasi awal, pilih Klien rahasia.

   Pilih Selanjutnya.

f. Langkah 6: Tinjau dan buat - Tinjau pilihan Anda dari bagian sebelumnya, lalu pilih Buat kumpulan pengguna.

5. Pada halaman User pool, pilih pool yang baru saja Anda buat.

   Salin ID kumpulan Pengguna dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CognitoUserPool.Default.PoolId`.

6. Pilih tab Integrasi aplikasi dan pergi ke bagian bawah halaman.
7. Di bagian Daftar klien Aplikasi, pilih nama klien Aplikasi yang baru saja Anda buat.

   Salin ID Klien dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CognitoUserPool.Default.AppClientId`.

8. Tunjukkan rahasia Klien dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CognitoUserPool.Default.AppClientSecret`.

Siapkan kolam identitas

Untuk mengatur kumpulan identitas

1. Masuk ke konsol Amazon Cognito dan verifikasi wilayah tersebut benar.
2. Di navigasi di sebelah kiri pilih Identity pool.
3. Pilih Buat kumpulan identitas.

4. Konfigurasikan kumpulan identitas.

   a. Langkah 1: Konfigurasikan kepercayaan kumpulan identitas - Lengkapi bagian berikut:

      - Akses pengguna - Pilih Akses yang Diautentikasi
      - Sumber identitas yang diautentikasi - Pilih kumpulan pengguna Amazon Cognito

      Pilih Selanjutnya.

   b. Langkah 2: Konfigurasikan izin - Di bagian peran yang diautentikasi, lengkapi bidang berikut:

      - Peran IAM - Pilih Buat peran IAM baru
      - Nama peran IAM - Masukkan nama dan catat untuk langkah selanjutnya.

      Pilih Selanjutnya.

   c. Langkah 3: Hubungkan penyedia identitas - Di bagian Rincian kumpulan pengguna, lengkapi bidang berikut:

      - ID kumpulan pengguna - Pilih kumpulan pengguna yang Anda buat sebelumnya.
      - ID klien aplikasi - Pilih ID klien aplikasi yang Anda buat sebelumnya.

      Pilih Selanjutnya.

   d. Langkah 4: Konfigurasikan properti - Ketik nama di bidang Identity pool name.

      Pilih Selanjutnya.

   e. Langkah 5: Tinjau dan buat - Tinjau pilihan Anda di setiap bagian, lalu pilih Buat kumpulan identitas.

5. Pada halaman Identity pool, pilih kumpulan identitas baru Anda.

   Salin ID kumpulan Identitas dan catat ini untuk nanti. Dalam `awsconfiguration.json` file, ini adalah`CredentialsProvider.CognitoIdentity.Default.PoolId`.

6. Perbarui izin untuk peran IAM.

   a. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   b. Di navigasi di sebelah kiri, pilih Peran.

   c. Temukan dan pilih peran yang Anda buat di atas.

> **Note**
>
> Gunakan bilah pencarian, jika perlu.

d. Pilih kebijakan izin terlampir.

   Pilih Edit.

e. Pilih tab JSON dan ganti kebijakan dengan yang berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:*",
                "kinesisvideo:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

   Pilih Selanjutnya.

f. Pilih kotak di samping Setel versi baru ini sebagai default jika belum dipilih.

   Pilih Simpan perubahan.

Metrik klien untuk C SDK

Aplikasi yang dibangun dengan Amazon Kinesis Video Streams dengan WebRTC terdiri dari berbagai bagian yang bergerak, termasuk jaringan, pensinyalan, pertukaran kandidat, koneksi rekan, dan pertukaran data. Kinesis Video Streams dengan WebRTC di C mendukung berbagai metrik sisi klien yang memungkinkan Anda memantau dan melacak kinerja dan penggunaan komponen ini dalam aplikasi Anda. Metrik yang didukung terbagi dalam dua kategori utama: metrik khusus yang didefinisikan secara khusus untuk implementasi sinyal dan jaringan Kinesis Video Streams, dan

metrik spesifik protokol terkait media dan data yang berasal dari standar W3C. Perhatikan bahwa hanya sebagian dari metrik standar W3C yang saat ini didukung untuk Kinesis Video Streams dengan WebRTC di C.

Topik

Metrik pensinyalan

Metrik pensinyalan dapat digunakan untuk memahami bagaimana klien pensinyalan berperilaku saat aplikasi Anda berjalan. Anda dapat menggunakan `STATUS signalingClientGetMetrics (SIGNALING_CLIENT_HANDLE, PSignalingClientMetrics)` API untuk mendapatkan metrik pensinyalan ini. Berikut adalah contoh pola penggunaan:

```
SIGNALING_CLIENT_HANDLE signalingClientHandle;
SignalingClientMetrics signalingClientMetrics;
STATUS retStatus = signalingClientGetMetrics(signalingClientHandle,
 &signalingClientMetrics);
printf("Signaling client connection duration: %" PRIu64 " ms",
       (signalingClientMetrics.signalingClientStats.connectionDuration /
 HUNDREDS_OF_NANOS_IN_A_MILLISECOND));
```

Definisi `signalingClientStats` dapat ditemukan di Stats.h.

Metrik pensinyalan berikut saat ini didukung:

Metrik	Deskripsi	
cpApiCallLatensi	Hitung latensi untuk panggilan API bidang kontrol. Perhitung an dilakukan dengan menggunakan Exponenti al Moving Average (EMA). Panggilan terkait meliputi: DescribeChannel, createCha nnel, dan deleteChannel. getChannelEndpoint	

Metrik	Deskripsi	
dpApiCallLatensi	Hitung latensi untuk panggilan API bidang data. Perhitungan dilakukan dengan menggunakan Exponential Moving Average (EMA). Panggilan terkait meliputi: getIceConfig.	
signalingClientUptime	Ini menunjukkan waktu keberadaan objek klien. Setiap kali metrik ini dipanggil, nilai uptime terbaru dipancarkan.	
KoneksiDurasi	Jika koneksi dibuat, ini memancarkan durasi koneksi hidup. Lain, nilai 0 dipancarkan. Ini berbeda dengan memberi sinyal uptime klien karena, koneksi datang dan pergi, tetapi signalingClientUptime merupakan indikasi dari objek klien itu sendiri.	
numberOfMessagesDikirim	Nilai ini diperbarui ketika rekan mengirimkan penawaran, jawaban, atau kandidat ICE.	
numberOfMessagesDiterima	Tidak seperti numberOfMessages Terkirim, metrik ini diperbarui untuk semua jenis pesan pensinyalan. Jenis pesan pensinyalan tersedia di`SIGNALING_MESSAGE_TYPE` .	

Metrik	Deskripsi	
iceRefreshCount	Ini bertambah saat getIceCon fig dipanggil. Tingkat di mana ini dipanggil didasarkan pada TTL sebagai bagian dari konfigurasi ICE yang diterima. Setiap kali satu set konfigura si ICE baru diterima, timer diatur untuk menyegarkan waktu berikutnya, mengingat validitas kredensil dalam konfigurasi dikurangi beberapa masa tenggang.	
numberOfErrors	Penghitung digunakan untuk melacak jumlah kesalahan yang dihasilkan dalam klien pensinyalan. Kesalahan yang dihasilkan saat mendapatkan konfigurasi ICE, mendapatk an status pensinyalan, melacak metrik pensinyalan, mengirim pesan pensinyalan, dan menghubungkan klien pensinyalan ke soket web agar pesan dilacak. send/rece ive	
numberOfRuntimeKesalahan	Metrik mencakup kesalahan yang terjadi saat inti klien pensinyalan sedang berjalan. Skenario seperti kegagalan menyambung kembali, kegagalan penerimaan pesan, dan kesalahan penyegaran konfigurasi ICE dilacak di sini.	

Metrik	Deskripsi	
numberOfReconnects	Metrik bertambah pada setiap penyambungan kembali. Ini adalah metrik yang berguna untuk memahami stabilita s koneksi jaringan dalam pengaturan.	

Metrik standar W3C didukung untuk C SDK

Subset dari metrik standar W3C saat ini didukung untuk aplikasi yang dibangun dengan WebRTC C SDK. Ini termasuk dalam kategori berikut:

- Jaringan:
  - Kandidat Es: metrik ini memberikan informasi tentang kandidat lokal dan jarak jauh yang dipilih untuk pertukaran data antara rekan-rekan. Ini termasuk sumber server kandidat, alamat IP, jenis kandidat yang dipilih untuk komunikasi, dan prioritas kandidat. Metrik ini berguna sebagai laporan snapshot.
  - Ice Server: metrik ini untuk mengumpulkan informasi operasional tentang berbagai server ICE yang didukung. Ini berguna ketika mencoba memahami server yang terutama digunakan untuk pemeriksaan komunikasi dan konektivitas. Dalam beberapa kasus, juga berguna untuk memeriksa metrik ini jika pengumpulan kandidat gagal.
  - Pasangan Kandidat Es: metrik ini untuk memahami jumlah bytes/packets yang dipertukarkan antara rekan-rekan dan juga pengukuran terkait waktu.
- Media dan data:
  - RTP Inbound Jarak Jauh: metrik ini mewakili perspektif titik akhir dari aliran data yang dikirim oleh pengirim.
  - Outbound RTP: metrik ini memberikan informasi tentang aliran RTP keluar. Mereka juga bisa sangat berguna saat menganalisis streaming berombak atau streaming berhenti.
  - RTP masuk: metrik ini memberikan informasi tentang media yang masuk.
  - Metrik saluran data: metrik ini dapat membantu Anda menganalisis jumlah pesan dan byte yang dikirim dan diterima melalui saluran data. Metrik dapat ditarik dengan menggunakan ID saluran.

Anda dapat menggunakan `STATUS rtcPeerConnectionGetMetrics (PRtcPeerConnection, PRtcRtpTransceiver, PRtcStats)` API untuk mengumpulkan metrik yang terkait dengan ICE, RTP, dan saluran data. Berikut adalah contoh penggunaan:

```
RtcStats rtcStats;
rtcStats.requestedTypeOfStats = RTC_STATS_TYPE_LOCAL_CANDIDATE;
STATUS retStatus = rtcPeerConnectionGetMetrics (pRtcPeerConnection, NULL, &rtcStats);
printf("Local Candidate address: %s\n",
 rtcStats.rtcStatsObject.localIceCandidateStats.address);
```

Berikut contoh lain yang menunjukkan pola penggunaan untuk mendapatkan statistik terkait transceiver:

```
RtcStats rtcStats;
PRtcRtpTransceiver pVideoRtcRtpTransceiver;
rtcStats.requestedTypeOfStats = RTC_STATS_TYPE_OUTBOUND_RTP;
STATUS retStatus = rtcPeerConnectionGetMetrics (pRtcPeerConnection,
 pVideoRtcRtpTransceiver, &rtcStats);
printf("Number of packets discarded on send: %s\n",
 rtcStats.rtcStatsObject.outboundRtpStreamStats.packetsDiscardedOnSend);
```

Dalam contoh di atas, jika argumen kedua untuk rtcPeerConnection GetMetrics () adalah NULL, data untuk transceiver pertama dalam daftar dikembalikan.

Definisi untuk rtcStatsObject dapat ditemukan di Stats.h. dan definisi untuk RtcStats dapat ditemukan di Include.h.

Penggunaan sampel dari APIs dan metrik yang berbeda dapat ditemukan di direktori sampel di repositori WebRTC C SDK dan di repositori demo Kinesis Video Stream.

Metrik standar W3C berikut saat ini didukung untuk aplikasi yang dibangun dengan WebRTC C SDK.

Topik

- Jaringan
- Media
- Saluran data

Jaringan

Metrik Server ICE:

Metrik	Deskripsi
URL	URL dari STUN/TURN server yang sedang dilacak
Port	Nomor port yang digunakan oleh klien
Protokol	Protokol transportasi diekstrak dari URI ICE Server. Jika nilainya UDP, ICE mencoba TURN over UDP, kalau tidak ICE mencoba TURN over. TCP/TLS. If the URI does not contain transport, ICE tries TURN over UDP and TCP/TLS Dalam kasus STUN server, bidang ini kosong.
Total Permintaan Terkirim	Nilai diperbarui untuk setiap permintaan kandidat srflx dan saat mengirim permintaan yang mengikat dari kandidat giliran.
Total Tanggapan Diterima	Nilai diperbarui setiap kali respons pengikatan STUN diterima.
Total Waktu Perjalanan Pulang Pergi	Nilai diperbarui setiap kali respons yang setara diterima untuk permintaan. Paket permintaan dilacak dalam peta hash dengan checksum sebagai kuncinya.

Statistik Kandidat ICE: Hanya informasi tentang kandidat yang dipilih (lokal dan jarak jauh) yang disertakan.

Metrik	Deskripsi
alamat	Ini menunjukkan alamat IP kandidat lokal dan jarak jauh.
port	Nomor port kandidat
protokol	Protokol digunakan untuk mendapatkan kandidat. Nilai yang valid adalah UDP/TCP.
Jenis KandidateType	Jenis kandidat yang dipilih - host, srflx atau relay.
prioritas	Prioritas kandidat lokal dan jarak jauh yang dipilih.
url	Sumber kandidat lokal yang dipilih. Ini memberikan indikasi apakah kandidat yang dipilih diterima dari server STUN atau server TURN.
RelayProtocol	Jika server TURN digunakan untuk mendapatkan kandidat lokal yang dipilih, bidang ini menunjukkan protokol apa yang digunakan untuk mendapatkannya. Nilai yang valid adalah TCP/UDP.

Statistik Pasangan Calon ICE: Hanya informasi tentang pasangan calon yang dipilih yang disertakan.

Metrik	Deskripsi
localCandidateId	ID kandidat lokal yang dipilih dalam pasangan.
remoteCandidateId	ID kandidat jarak jauh yang dipilih dalam pasangan.
negara	Keadaan pasangan calon yang sedang diperiksa.
dinominasikan	Setel ke TRUE karena statistik ditarik untuk pasangan kandidat yang dipilih.
PaketsSent	Jumlah paket yang dikirim Ini dihitung dalam `.` panggilan dalam `writeFrame` panggilan. Informasi ini juga dapat diekstraksi dari Statistik RTP keluar, tetapi karena pasangan kandidat Ice menyertakan lastPacke tSent stempel waktu, mungkin berguna untuk menghitun g jumlah paket yang dikirim antara dua titik waktu.
PaketDiterima	Ini diperbarui setiap kali incomingDataHandler dipanggil.
olehTessent	Ini dihitung `iceAgentS endPacket()` dalam `writeFrame()` panggilan . Ini berguna saat menghitun g bit rate. Saat ini, ini juga termasuk header dan padding

Metrik	Deskripsi	
	karena lapisan ICE tidak menyadari format paket RTP.	
BytesDiterima	Ini diperbarui setiap kali incomingDataHandler dipanggil. Saat ini, ini juga termasuk header dan padding karena lapisan ICE tidak menyadari format paket RTP.	
lastPacketSentStempel waktu	Ini diperbarui setiap kali paket dikirim. Ini dapat digunakan bersama dengan PacketsSent dan waktu mulai yang direkam dalam aplikasi ke laju transfer paket saat ini.	
lastPacketReceivedStempel waktu	Ini diperbarui saat menerima data di`incomingDataHandler()` . Ini dapat digunakan bersama dengan PacketsReceived untuk menyimpulkan tingkat penerimaan paket saat ini. Waktu mulai harus direkam di lapisan aplikasi di `transceiverOnFrame()` callback.	

Metrik	Deskripsi
firstRequestTimestamp	Terekam ketika permintaan pengikatan STUN pertama berhasil dikirim masuk. `iceAgentSendStunPacket()` Ini dapat digunakan bersama dengan lastRequestTimestamp dan requestsSent untuk menemukan waktu rata-rata antara permintaan pengikatan STUN.
lastRequestTimestamp	Direkam setiap kali permintaan pengikatan STUN berhasil dikirim masuk. `iceAgentSendStunPacket()`
lastResponseTimestamp	Direkam setiap kali respons pengikatan STUN diterima.
totalRoundTripWaktu	Diperbarui saat respons yang mengikat diterima untuk permintaan. Permintaan dan respons dipetakan dalam tabel hash berdasarkan checksum.
currentRoundTripWaktu	Waktu perjalanan pulang pergi terbaru diperbarui ketika tanggapan yang mengikat diterima untuk permintaan pada pasangan kandidat.
Permintaan Diterima	Penghitung yang diperbarui pada setiap permintaan pengikatan STUN yang diterima.

Metrik	Deskripsi
PermintaanSent	Penghitung yang diperbaru i pada setiap permintaa n pengikatan STUN yang dikirim masuk `iceAgentS endStunPacket()` .
TanggapanTerkirim	Penghitung yang diperbarui pada setiap respons pengikata n STUN yang dikirim sebagai tanggapan atas permintaan yang mengikat di `handleStu nPacket()` .
TanggapanDiterima	Penghitung yang diperbarui pada setiap respons pengikata n STUN yang diterima di `handleStunPacket()` .
packetsDiscardedOnKirim	Diperbarui saat pengirima n paket gagal. Dengan kata lain, ini diperbarui ketika `iceUtilsSendData()` gagal. Ini berguna untuk menentukan persentase paket yang dijatuhkan dalam durasi tertentu.

Metrik	Deskripsi
bytesDiscardedOnKirim	Diperbarui saat pengirima n paket gagal. Dengan kata lain, ini diperbarui ketika `iceUtilsSendData()` gagal. Ini berguna saat menentukan persentase paket yang dijatuhkan dalam durasi tertentu. Perhatikan bahwa penghitung juga menyertakan header paket.

Media

Statistik RTP Keluar

Metrik	Deskripsi
voiceActivityFlag	Ini saat ini merupakan bagian dari `RtcEncoderStats` didefinisikan dalam Include.h . Bendera diatur ke TRUE jika paket audio terakhir berisi suara. Bendera saat ini tidak diatur dalam sampel.
PaketsSent	Ini menunjukkan jumlah total paket RTP yang dikirim untuk SSRC yang dipilih. Ini adalah bagian dari https://www.w3.org/TR/webrtc-stats/ #sentrtpstats -dict* dan disertakan sebagai bagian dari statistik keluar. Ini bertambah setiap kali writeFrame () dipanggil.

Metrik	Deskripsi
olehTessent	Jumlah total byte tidak termasuk header RTP dan padding yang dikirim. Ini diperbarui pada setiap panggilan WriteFrame.
EncoderImplementasi	Ini diperbarui oleh lapisan aplikasi sebagai bagian dari RtcEncoderStats objek.
packetsDiscardedOnKirim	Bidang ini diperbarui jika agen ICE gagal mengirim paket RTP terenkripsi karena alasan apa pun dalam panggilan. `iceAgentSendPacket`
bytesDiscardedOnKirim	Bidang ini juga diperbarui jika agen ICE gagal mengirim paket RTP terenkripsi karena alasan apa pun dalam panggilan tersebut. `iceAgentSendPacket`
FrameSent	Ini bertambah hanya jika jenis taktik aliran media adalah MEDIA_STREAM_TRACK _KIND_VIDEO.

Metrik	Deskripsi	
hugeFramesSent	Penghitung ini diperbarui untuk bingkai yang 2,5 kali ukuran rata-rata bingkai. Ukuran frame diperoleh dengan menghitung fps (berdasarkan waktu hitungan frame terakhir yang diketahui dan jumlah frame yang dikodekan dalam interval waktu) dan menggunakan targetBitRate yang RtcEncoderStats ditetapkan oleh aplikasi.	
FrameDisandikan	Penghitung ini diperbarui hanya untuk trek video setelah pengkodean frame berhasil. Ini diperbarui pada setiap panggilan WriteFrame.	
keyFramesEncoded	Penghitung ini diperbarui hanya untuk trek video setelah berhasil menyandikan bingkai kunci. Ini diperbarui pada setiap panggilan WriteFrame.	
framesDiscardedOnKirim	Ini diperbarui ketika pengiriman bingkai gagal karena kegagalan `iceAgentSendPacket` panggilan. Sebuah frame terdiri dari sekelompok paket dan saat ini, framesDiscardedOn Send gagal jika ada paket yang dibuang saat mengirim karena kesalahan.	

Metrik	Deskripsi
FrameWidth	Ini idealnya mewakili lebar bingkai dari bingkai yang dikodekan terakhir. Saat ini, ini diatur ke nilai oleh aplikasi sebagai bagian dari RtcEncoderStats * * dan tidak terlalu penting.
FrameHeight	Ini idealnya mewakili ketinggian bingkai dari bingkai yang dikodekan terakhir. Saat ini, ini diatur ke nilai oleh aplikasi sebagai bagian dari RtcEncoderStats dan tidak terlalu penting.
frameBitDepth	Ini mewakili kedalaman bit per lebar piksel dari bingkai yang dikodekan terakhir. Saat ini, ini diatur oleh aplikasi sebagai bagian dari RtcEncoderStats dan diterjemahkan ke dalam statistik keluar.
NackCount	Nilai ini diperbarui setiap kali NACK diterima pada paket RTP dan upaya ulang untuk mengirim paket dilakukan . Tumpukan mendukung transmisi ulang paket saat menerima NACK.

Metrik	Deskripsi	
FirCount	Nilai diperbarui saat menerima paket FIR (onRtcpPacket-> FIRPacket onRTCP). Ini menunjukkan seberapa sering aliran tertinggal dan harus melewati bingkai untuk mengejar ketinggalan. Paket FIR saat ini tidak diterjema hkan untuk mengekstrak bidang, jadi, meskipun hitungan diatur, tidak ada tindakan yang diambil.	
PliCount	Nilai diperbarui saat menerima paket PLI (-> onRTCP)on RtcpPacket. PLIPacket Ini menunjukkan bahwa sejumlah data video yang dikodekan telah hilang untuk satu atau lebih frame.	
SliCount	Nilai diperbarui saat menerima paket SLI (-> onRTCP)on RtcpPacket. SLIPacket Ini menunjukkan seberapa sering kehilangan paket mempengar uhi satu frame.	
qualityLimitationResolution Perubahan	Saat ini, tumpukan mendukung metrik ini, namun, lebar dan tinggi bingkai tidak dipantau untuk setiap bingkai yang dikodekan.	

Metrik	Deskripsi	
lastPacketSentStempel waktu	Stempel waktu di mana paket terakhir dikirim. Ini diperbaru i pada setiap panggilan WriteFrame.	
headerBytesSent	Jumlah total header RTP dan byte padding yang dikirim untuk SSRC ini tidak termasuk muatan RTP yang sebenarny a.	
bytesDiscardedOnKirim	Ini diperbarui ketika pengirima n bingkai gagal karena kegagalan panggilan iceAgentSend paket. Sebuah frame terdiri dari sekelompo k paket, yang pada giliranny a terdiri dari byte dan saat ini, bytesDiscardedOn Send gagal jika ada paket yang dibuang saat mengirim karena kesalahan.	
retransmittedPacketsSent	Jumlah paket yang ditransmi sikan kembali pada penerimaa n. PLI/SLI/NACK Saat ini, tumpukan hanya menghitun g paket resent NACK karena transmisi ulang berbasis PLI dan SLI tidak didukung.	

Metrik	Deskripsi	
retransmittedBytesSent	Jumlah byte yang ditransmi sikan kembali pada penerimaa n. PLI/SLI/NACK Saat ini, tumpukan hanya menghitung byte resent dari NACK karena transmisi ulang berbasis PLI dan SLI tidak didukung.	
TargetBitrate	Ini diatur dalam tingkat aplikasi.	
totalEncodedBytesTarget	Ini ditingkatkan dengan ukuran frame target dalam byte setiap kali frame dikodekan . Ini diperbarui menggunak an parameter ukuran dalam struktur Frame.	
framesPerSecond	Ini dihitung berdasarkan waktu yang direkam untuk bingkai terenkode terakhir yang diketahui dan jumlah frame yang dikirim dalam satu detik.	
totalEncodeTime	Ini diatur ke nilai arbitrer dalam aplikasi dan diterjemahkan ke statistik keluar secara internal.	
totalPacketSendKeterlambata n	Ini saat ini diatur ke 0 karena iceAgentSend Packet mengirim paket segera.	

Statistik RTP masuk jarak jauh:

Metrik	Deskripsi	
roundTripTime	Nilai diekstraksi dari laporan penerima RTCP saat menerima paket RTCP tipe 201 (laporan penerima) . Laporan tersebut terdiri dari rincian seperti laporan pengirim terakhir dan penundaan sejak laporan pengirim terakhir untuk menghitung waktu pulang pergi. Laporan pengirim dihasilkan kira-kira setiap 200 milidetik yang terdiri dari informasi seperti jumlah paket yang dikirim dan byte yang dikirim yang diekstraksi dari statistik keluar.	
totalRoundTripWaktu	Jumlah waktu perjalanan pulang pergi dihitung	
FraksiHilang	Merupakan fraksi paket RTP yang hilang untuk SSRC sejak ReporFractionLost sebelumny a dikirim. sender/receiver	
LaporanDiterima	Diperbarui setiap kali paket jenis laporan penerima diterima.	
roundTripTimePengukuran	Menunjukkan jumlah total laporan yang diterima untuk SSRC yang berisi waktu pulang pergi yang valid. Namun, saat ini nilai	

Metrik	Deskripsi	
	ini bertambah terlepas sehingga artinya sama dengan ReportsReceived.	

Statistik RTP Masuk:

Metrik	Deskripsi	
PaketDiterima	Penghitung diperbarui ketika paket diterima untuk SSRC tertentu.	
jitter	Metrik ini menunjukkan paket Jitter yang diukur dalam hitungan detik untuk SSRC tertentu.	
jitterBufferDelay	Metrik ini menunjukkan jumlah waktu yang dihabiskan oleh setiap paket dalam buffer jitter.	
jitterBufferEmittedHitungan	Jumlah total sampel audio atau bingkai video yang keluar dari buffer jitter.	
PaketDibuang	Penghitung diperbarui ketika buffer Jitter penuh dan paket tidak dapat didorong ke dalamnya. Ini dapat digunakan untuk menghitung persentase paket yang dibuang dalam durasi tetap.	

Metrik	Deskripsi
BingkaiDijatuhkan	Nilai ini diperbarui saat `onFrameDroppedFunc()` dipanggil.
lastPacketReceivedStempel waktu	Merupakan stempel waktu di mana paket terakhir diterima untuk SSRC ini.
headerBytesReceived	Penghitung diperbarui saat menerima paket RTP.
BytesDiterima	Jumlah byte yang diterima. Ini tidak termasuk byte header. Metrik ini dapat digunakan untuk menghitung bit rate yang masuk.
packetsFailedDecryption	Ini bertambah ketika dekripsi paket SRTP gagal.

Saluran data

Metrik saluran data:

Metrik	Deskripsi
label	Label adalah nama saluran data yang sedang diperiksa.
protokol	Karena tumpukan kami menggunakan SCTP, protokol diatur ke SCTP konstan.
dataChannelIdentifier	Pengidentifikasi genap atau ganjil digunakan untuk mengidentifikasi saluran data

Metrik	Deskripsi	
	secara unik. Ini diperbarui ke nilai ganjil jika SDK adalah penawaran dan nilai genap jika SDK adalah penjawab.	
negara	Status saluran data saat statistik ditanyakan. Saat ini, dua status yang didukung adalah RTC_DATA_CHANNEL_STATE_CONNECTING (saat saluran dibuat) dan RTC_DATA_CHANNEL_STATE_OPEN (Set dalam acara onOpen ()).	
PesanSent	Penghitung diperbarui saat SDK mengirim pesan melalui saluran data.	
olehTessent	Penghitung diperbarui dengan byte dalam pesan yang dikirim. Ini dapat digunakan untuk memahami berapa banyak byte yang tidak dikirim karena kegagalan, yaitu, untuk memahami persentase byte yang dikirim.	
PesanDiterima	Metrik bertambah dalam `onMessage()` callback.	
BytesDiterima	Metrik dihasilkan dalam `onMessage()` callback.	

Gunakan Amazon Kinesis Video Streams dengan WebRTC untuk menelan dan menyimpan media

Amazon Kinesis Video Streams menawarkan kemampuan untuk melakukan streaming video dan audio secara real-time melalui WebRTC ke cloud untuk penyimpanan, pemutaran, dan pemrosesan analitis. Pelanggan dapat menggunakan SDK WebRTC dan APIs cloud kami yang disempurnakan untuk mengaktifkan streaming real-time, serta konsumsi media ke cloud.

Untuk memulai, Anda dapat menginstal Amazon Kinesis Video Streams dengan WebRTC SDK pada kamera atau perangkat AWS IoT keamanan apa pun dengan sensor video APIsdan menggunakan kami untuk mengaktifkan streaming media dengan latensi sub 1 detik, serta konsumsi dan penyimpanan di cloud. Setelah tertelan, Anda dapat mengakses data Anda melalui kami easy-to-use APIs. Amazon Kinesis Video Streams memungkinkan Anda memutar video untuk ditonton langsung dan sesuai permintaan, serta dengan cepat membangun aplikasi yang memanfaatkan visi komputer dan analitik video melalui integrasi dengan Amazon Rekognition Video dan AI. SageMaker

Topik

Operasi API

Gunakan operasi API berikut untuk mengonfigurasi konsumsi Amazon Kinesis Video Streams WebRTC:

- DescribeMappedResourceConfiguration
- DescribeMediaStorageConfiguration
- JoinStorageSession
- JoinStorageSessionAsViewer
- UpdateMediaStorageConfiguration

Apa itu Amazon Kinesis Video Streams dengan konsumsi dan penyimpanan WebRTC?

Amazon Kinesis Video Streams menawarkan kemampuan untuk melakukan streaming video dan audio secara real-time melalui WebRTC ke cloud untuk penyimpanan, pemutaran, dan pemrosesan analitis. Topik ini akan memberikan step-by-step instruksi untuk mengatur dan menggunakan SDK WebRTC dan cloud kami untuk mengaktifkan streaming real-time dan konsumsi media ke APIs cloud. Instruksi ini mencakup panduan untuk menggunakan AWS Command Line Interface dan konsol Kinesis Video Streams.

Sebelum Anda menggunakan Amazon Kinesis Video Streams dengan WebRTC untuk pertama kalinya, lihat. the section called “Mengatur sebuah Akun AWS”

Memahami konsumsi dan penyimpanan WebRTC

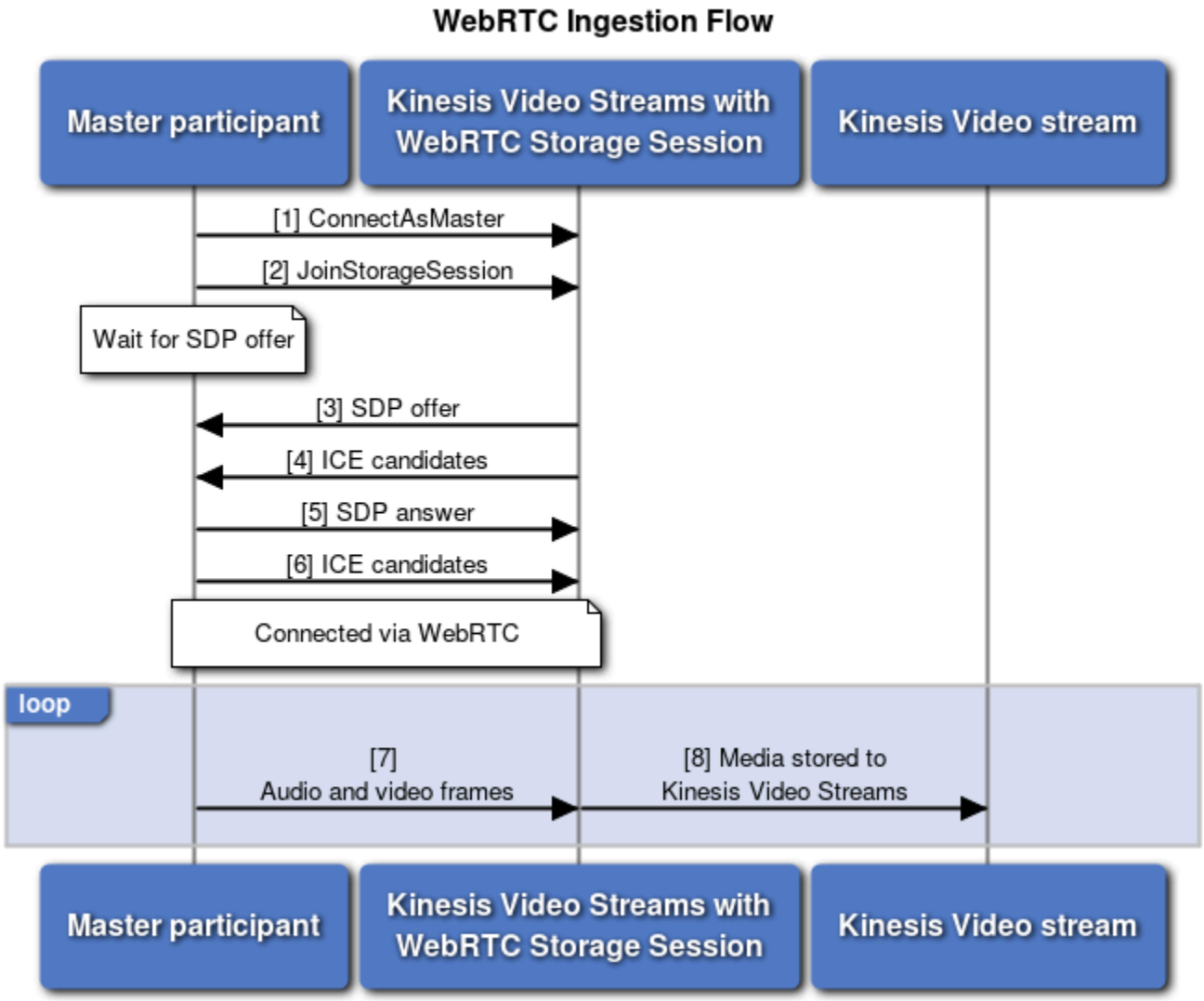
Bagian berikut menjelaskan berbagai opsi konsumsi dan penyimpanan yang tersedia di Kinesis Video Streams dengan WebRTC.

Topik

- Hanya peserta master
- Peserta master dan pemirsa bersama-sama

Hanya peserta master

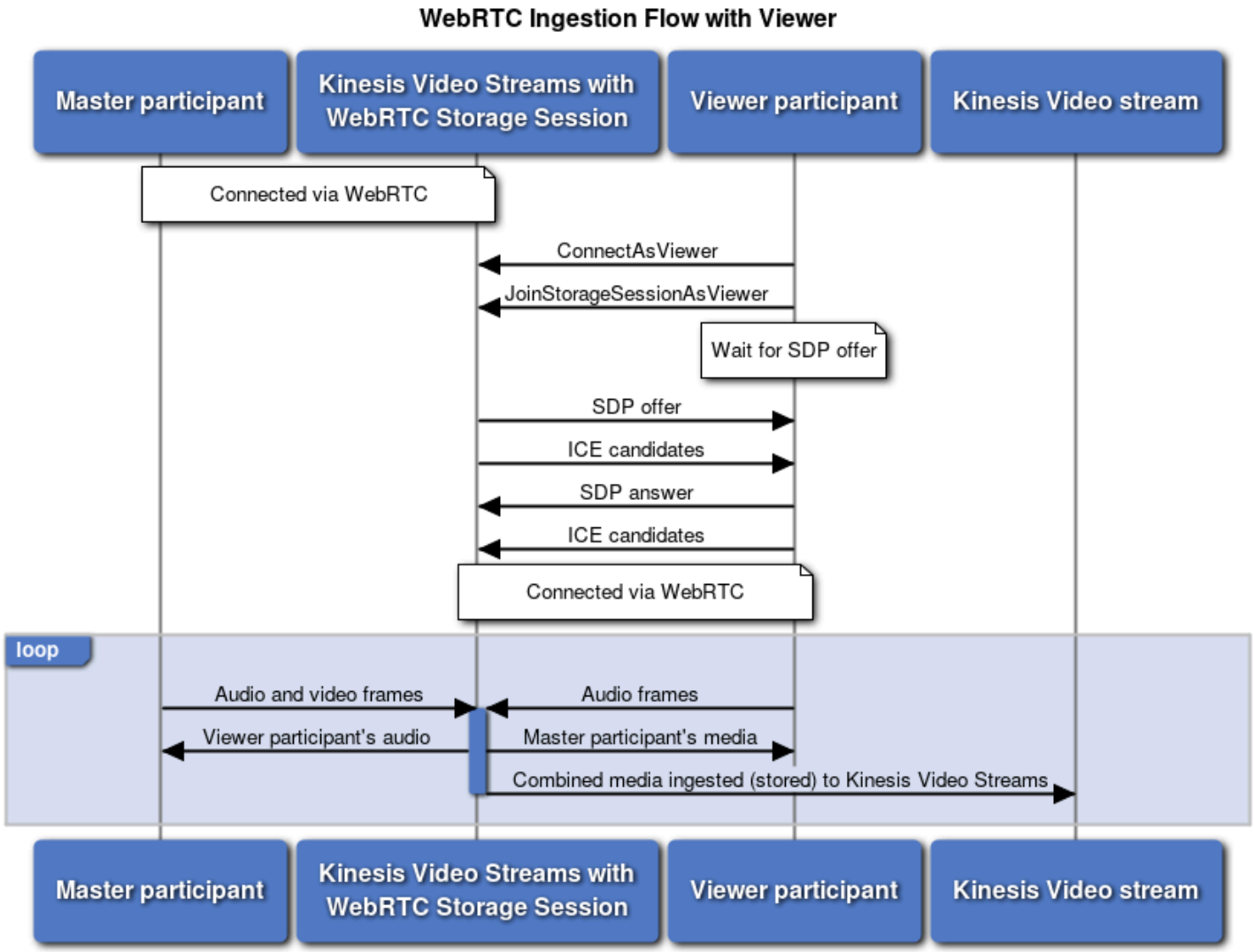
Peserta master pertama-tama terhubung ke Kinesis Video Streams dengan the section called “ConnectAsMaster” WebRTC Signaling via. Selanjutnya, mereka memanggil JoinStorageSessionAPI agar sesi penyimpanan memulai koneksi WebRTC. Setelah koneksi WebRTC dibuat, media akan dicerna ke aliran video Kinesis yang dikonfigurasi.



Peserta master dan pemirsa bersama-sama

Peserta pemirsa pertama terhubung ke Kinesis Video Streams dengan the section called “ConnectAsViewer” WebRTC Signaling via. Selanjutnya, mereka memanggil JoinStorageSessionAsViewerAPI agar sesi penyimpanan memulai koneksi WebRTC. Setelah koneksi WebRTC dibuat, media gabungan dari master dan semua peserta pemirsa akan dicerna ke aliran video Kinesis yang dikonfigurasi, selama peserta master hadir.

Sesi penyimpanan menggabungkan dan meneruskan semua audio peserta pemirsa ke peserta utama. Peserta pemirsa menerima media gabungan dari peserta utama dan audio dari peserta pemirsa lain dari sesi penyimpanan.



Buat koneksi WebRTC dengan sesi penyimpanan

Karena sesi penyimpanan berada dalam jaringan Amazon, sesi penyimpanan hanya akan mengirim `relay` (TURN) kandidat ke peserta. Jika jaringan peserta memungkinkan, `srflx` (STUN) kandidat dapat digunakan untuk terhubung ke sesi penyimpanan. Dengan kata lain, dari sudut pandang peserta, kandidat ICE yang dinominasikan lokal dapat `srflx` atau`relay`, sedangkan kandidat ICE jarak jauh selalu`relay`.

Untuk mengoptimalkan waktu koneksi, jangan mengirim `host` kandidat ke sesi penyimpanan. Sesi penyimpanan juga `Trickle ICE` harus digunakan.

Lihat the section called “Memecahkan masalah koneksi sesi penyimpanan” untuk memecahkan masalah koneksi ke sesi penyimpanan.

Buat saluran pensinyalan

Kinesis Video Streams dengan saluran pensinyalan WebRTC memfasilitasi pertukaran pesan pensinyalan yang diperlukan untuk membangun dan memelihara koneksi antara klien WebRTC. peer-to-peer Ini menangani negosiasi penawaran dan jawaban Session Description Protocol (SDP) untuk parameter sesi, serta pertukaran kandidat Interactive Connectivity Establishment (ICE) untuk informasi jaringan.

Untuk membuat saluran pensinyalan, panggil CreateSignalingChannelAPI. Halaman ini akan menunjukkan kepada Anda cara menjalankan API itu menggunakan Konsol Manajemen AWS, AWS CLI, dan salah satunya. AWS SDKs

> **Important**
>
> Catat saluran ARN, Anda akan membutuhkannya nanti.

Konsol Manajemen AWS

Lakukan hal-hal berikut:

1. Buka konsol Saluran Pensinyalan Kinesis Video Streams di rumah/#/SignalingChannels. https://console.aws.amazon.com/kinesisvideo/
2. Pilih Buat saluran pensinyalan.
3. Pada halaman Buat saluran pensinyalan baru, ketikkan nama untuk saluran pensinyalan.

   Biarkan nilai default Time-to-live (Ttl) sebagai 60 detik.

   Pilih Buat saluran pensinyalan.
4. Setelah saluran pensinyalan dibuat, tinjau detail di halaman detail saluran.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS Command Line Interface.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan perintah Create-Signaling-Channel berikut menggunakan: AWS CLI

```
aws kinesisvideo create-signaling-channel \
  --channel-name "YourChannelName" \
  --region "us-west-2"
```

Responsnya akan terlihat seperti berikut:

```
{
    "ChannelARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123"
}
```

AWS SDK

Cuplikan kode ini menunjukkan cara membuat Kinesis Video Streams dengan saluran pensinyalan WebRTC menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap di GitHub.

Buat klien Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil `CreateSignalingChannel` API.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `CreateSignalingChannel` API.

```
const createSignalingChannelResponse = await kinesisVideoClient
    .createSignalingChannel({
        ChannelName: 'YourChannelName',
    })
    .promise();
```

Cetak responsnya.

```
console.log(createSignalingChannelResponse.ChannelARN);
```

Halaman web langsung dengan contoh kode ini tersedia untuk digunakan di GitHub. Masukkan wilayah, AWS kredensil, dan nama saluran pensinyalan Anda.

Pilih Buat Saluran.

Buat aliran video

Ikuti prosedur ini untuk membuat aliran yang akan dicerna media. Jika Anda telah membuat aliran tujuan, lewati langkah ini.

> **Important**
>
> WebRTC Ingestion memerlukan aliran video Kinesis dengan retensi data lebih besar dari 0. Minimal adalah 1 jam.

Untuk membuat stream, panggil CreateStreamAPI menggunakan Konsol Manajemen AWS, AWS CLI, atau salah satu AWS SDK.

> **Important**
>
> Catat aliran ARN, Anda akan membutuhkannya nanti.

Konsol Manajemen AWS

Lakukan hal-hal berikut:

1. Buka konsol Kinesis Video Streams https://console.aws.amazon.com/kinesisvideo/ di rumah/.
2. Pada halaman Streaming video, pilih Buat aliran video.
3. Pada halaman Create a new video stream, masukkan *YourStreamName* nama stream. Biarkan tombol konfigurasi Default dipilih.

   Ini akan membuat aliran dengan retensi data lebih dari 0.

   Pilih Buat aliran video.

4. Setelah Kinesis Video Streams membuat streaming, tinjau detail *YourStreamName* di halaman.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS Command Line Interface.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan `Create-Stream` perintah berikut menggunakan AWS CLI:

```
aws kinesisvideo create-stream \
  --stream-name "YourStreamName" \
  --data-retention-in-hours 24 \
  --region "us-west-2"
```

Responsnya akan terlihat seperti berikut:

```
{
    "StreamARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:stream/YourStreamName/1234567890123"
}
```

AWS SDK

Cuplikan kode ini menunjukkan cara membuat aliran video Kinesis menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap di GitHub.

Buat klien Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil CreateStreamAPI.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
```

```
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `CreateStream` API.

```
const createStreamResponse = await kinesisVideoClient
    .createStream({
        StreamName: 'YourStreamName',
        DataRetentionInHours: 48,
    })
    .promise();
```

Cetak responsnya.

```
console.log(createStreamResponse.StreamARN);
```

Halaman web langsung dengan contoh kode ini tersedia untuk digunakan di GitHub. Masukkan wilayah, AWS kredensil, dan nama saluran pensinyalan Anda.

Perluas node WebRTC Ingestion dan Storage, ketik nama stream Anda, lalu pilih Create Stream. Sebuah pop-up menanyakan jumlah jam yang Anda inginkan untuk menyimpan data streaming. Masukkan nilai yang lebih besar dari 0, lalu pilih Create Stream.

Izin pemberian

Anda harus memberikan izin streaming ke peran IAM Anda untuk menyerap aliran di Amazon Kinesis Video Streams dengan WebRTC.

> **Note**
>
> Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

Peran Master dan Viewer juga harus memiliki`DescribeStream`,`GetDataEndpoint`, dan `PutMedia` izin untuk menyerap media ke Kinesis Video Streams.

Lihat contoh kebijakan IAM di bawah ini untuk peserta Master:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:DescribeSignalingChannel",
        "kinesisvideo:DescribeMediaStorageConfiguration",
        "kinesisvideo:GetSignalingChannelEndpoint",
        "kinesisvideo:GetIceServerConfig",
        "kinesisvideo:ConnectAsMaster",
        "kinesisvideo:JoinStorageSession"
      ],
      "Resource": "arn:aws:kinesisvideo:us-west-2:123456789012:channel/
SignalingChannelName/1234567890123"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:GetDataEndpoint",
        "kinesisvideo:DescribeStream",
        "kinesisvideo:PutMedia"
      ],
      "Resource": "arn:aws:kinesisvideo:us-west-2:123456789012:stream/
VideoStreamName/1234567890123"
    }
  ]
}
```

Konfigurasikan tujuan

Setelah Anda membuat sumber daya Kinesis Video Streams, Anda perlu memberi tahu saluran pensinyalan aliran mana yang akan disimpan.

Jika Anda ingin menghapus saluran pensinyalan atau streaming, Anda harus memutuskan tautannya terlebih dahulu. Lihat the section called “Putuskan tautan saluran dan streaming pensinyalan”.

Tautkan saluran dan aliran pensinyalan

Gunakan UpdateMediaStorageConfigurationAPI dan masukkan sumber daya Kinesis Video Streams yang ingin Anda tautkan. ARNs

> **Important**
>
> Setelah `StorageStatus` diaktifkan, koneksi langsung peer-to-peer (master-viewer) tidak lagi terjadi. Peer terhubung langsung ke sesi penyimpanan. Anda harus memanggil `JoinStorageSession` API untuk memicu pengiriman penawaran SDP dan membuat koneksi antara peer dan sesi penyimpanan.

Konsol Manajemen AWS

> **Note**
>
> Operasi ini saat ini tidak didukung di Kinesis Konsol Manajemen AWS Video Streams.

Buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Ikuti instruksi di AWS CLI tab.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi lebih lanjut, lihat AWS Command Line Interfacedokumentasi.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan `Update-Media-Storage-Configuration` perintah di AWS CLI:

```
aws kinesisvideo update-media-storage-configuration \
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
```

```
  --media-storage-configuration \
    StreamARN="arn:aws:kinesisvideo:us-
west-2:123456789012:stream/YourStreamName/1234567890123",Status="ENABLED" \
  --region "us-west-2"
```

AWS SDK

Cuplikan kode ini menunjukkan cara mengonfigurasi saluran pensinyalan untuk menyerap media ke aliran video Kinesis yang ditentukan menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap di GitHub.

Buat klien Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil UpdateMediaStorageConfigurationAPI.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `UpdateMediaStorageConfiguration` API.

```
await kinesisVideoClient
    .updateMediaStorageConfiguration({
        ChannelARN: 'YourChannelARN',
        MediaStorageConfiguration: {
            Status: 'ENABLED',
            StreamARN: 'YourStreamARN',
        },
    })
    .promise();
```

Halaman web langsung dengan contoh kode ini tersedia untuk digunakan di GitHub. Masukkan wilayah, AWS kredensil, dan nama saluran pensinyalan Anda.

Perluas node WebRTC Ingestion and Storage, ketik nama aliran Anda, lalu pilih Perbarui Konfigurasi Penyimpanan Media. Saluran akan dikonfigurasi untuk menyerap media ke aliran yang ditentukan.

Putuskan tautan saluran dan streaming pensinyalan

> **Important**
>
> Anda tidak dapat menghapus saluran pensinyalan atau streaming sampai mereka tidak terhubung satu sama lain.

Jika Anda tidak ingin media saluran pensinyalan tertelan ke streaming, gunakan UpdateMediaStorageConfigurationAPI untuk memutuskan tautan sumber daya Kinesis Video Streams. Setelah saluran tidak terhubung, peer-to-peer koneksi langsung dapat dilanjutkan.

Konsol Manajemen AWS

> **Note**
>
> Operasi ini saat ini tidak didukung di Kinesis Konsol Manajemen AWS Video Streams.

Buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Ikuti instruksi di AWS CLI tab.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi lebih lanjut, lihat AWS Command Line Interfacedokumentasi.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan `Update-Media-Storage-Configuration` perintah di AWS CLI:

```
aws kinesisvideo update-media-storage-configuration \
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
```

```
  --media-storage-configuration \
    StreamARN="null",Status="DISABLED" \
  --region "us-west-2"
```

AWS SDK

Cuplikan kode ini menunjukkan cara mengonfigurasi saluran pensinyalan untuk menyerap media ke aliran video Kinesis yang ditentukan menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap di GitHub.

Buat klien Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil UpdateMediaStorageConfigurationAPI.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `UpdateMediaStorageConfiguration` API.

```
await kinesisVideoClient
    .updateMediaStorageConfiguration({
        ChannelARN: 'YourChannelARN',
        MediaStorageConfiguration: {
            Status: 'DISABLED',
            StreamARN: 'null',
        },
    })
    .promise();
```

Halaman web langsung dengan contoh kode ini tersedia untuk digunakan di GitHub. Masukkan wilayah, AWS kredensyal, dan nama saluran pensinyalan Anda.

Perluas simpul WebRTC Ingestion dan Storage, verifikasi bahwa bidang Nama Aliran kosong, lalu pilih Perbarui Konfigurasi Penyimpanan Media. Saluran tidak akan lagi dikonfigurasi untuk menyerap media ke aliran yang ditentukan.

Menelan media

Batasan berikut sudah ada:

- Durasi sesi: satu jam, maksimal
- Saluran pensinyalan: maksimum 100 per akun dengan konfigurasi penyimpanan diaktifkan

Topik

Menelan media dari browser

> **Important**
>
> Chrome saat ini adalah satu-satunya browser yang didukung.

1. Buka Amazon Kinesis Video Streams dengan WebRTC SDK di halaman contoh. JavaScript
2. Lengkapi informasi berikut:
   - KVS Endpoint - Di bidang Region, pilih wilayah Anda.

     Misalnya, `us-west-2`.
   - AWS Kredensialnya

     Lengkapi bidang berikut:
     - ID Kunci Akses
     - Kunci Akses Rahasia
     - Token Sesi - Aplikasi sampel mendukung kredensyal sementara dan jangka panjang. Biarkan bidang ini kosong jika Anda menggunakan kredensi IAM jangka panjang. Lihat Kredensi keamanan sementara di IAM untuk informasi selengkapnya.

- Saluran Sinyal - Di bidang Nama Saluran, ketik nama saluran pensinyalan yang Anda konfigurasikan sebelumnya. Untuk informasi selengkapnya, lihat the section called "Konfigurasikan tujuan".
- Trek - Pilih Kirim Video dan Kirim Audio.
- WebRTC Ingestion and Storage - Perluas node dan pilih Secara otomatis menentukan mode konsumsi. Opsi ini membuat aplikasi sampel memanggil DescribeMediaStorageConfigurationAPI untuk menentukan mode mana yang akan dijalankan.

3. Pilih Mulai Master.

   Jika saluran pensinyalan dikonfigurasi untuk konsumsi menggunakan DescribeMediaStorageConfigurationAPI, aplikasi sampel akan secara otomatis memanggil JoinStorageSessionAPI segera setelah tersambung ke saluran Signaling untuk memulai alur kerja penyerapan WebRTC.

Menelan media dari WebRTC C SDK

Ikuti the section called "C SDK untuk perangkat yang disematkan" prosedur untuk membangun aplikasi sampel.

1. Siapkan lingkungan Anda dengan Akun AWS kredensi Anda:

   ```
   export AWS_ACCESS_KEY_ID=YourAccessKey
   export AWS_SECRET_ACCESS_KEY=YourSecretKey
   export AWS_DEFAULT_REGION=YourAWSRegion
   ```

   Jika Anda menggunakan AWS kredensil sementara, ekspor juga token sesi Anda:

   ```
   export AWS_SESSION_TOKEN=YourSessionToken
   ```

2. Jalankan sampel:

   Sampel master

   Arahkan ke `build` folder dan gunakan "1" sebagai argumen kedua. Jenis:

   ```
   ./samples/kvsWebrtcClientMaster channel-name 1
   ```

   GStreamer sampel master

Arahkan ke `build` folder dan gunakan audio-video-storage "" sebagai argumen kedua. Jenis:

```
./samples/kvsWebrtcClientMasterGstSample channel-name audio-video-storage testsrc
```

Ini memulai konsumsi WebRTC.

> **Note**
>
> Saluran pensinyalan yang disediakan harus dikonfigurasi untuk penyimpanan. Gunakan DescribeMediaStorageConfigurationAPI untuk mengonfirmasi.

Tambahkan pemirsa ke sesi konsumsi

Setelah saluran pensinyalan berada dalam mode WebRTC Ingestion, peserta pemirsa tidak lagi terhubung langsung ke peserta utama. Peserta penampil terhubung langsung ke sesi penyimpanan. Peserta pemirsa menerima media yang dikirim oleh peserta utama, dan peserta pemirsa dapat mengirim kembali audio opsional ke peserta utama. Setiap audio yang dikirim kembali oleh pemirsa akan dikirim ke semua rekan lain yang terhubung ke sesi penyimpanan dan dicerna ke Kinesis Video Stream, selama peserta utama terhubung ke sesi penyimpanan.

Batasan berikut sudah ada:

- Jumlah maksimum pemirsa: 3
- Waktu maksimum peserta penampil dapat dihubungkan ke sesi penyimpanan tanpa peserta master hadir: 3 menit

> **Important**
>
> Jika pemirsa terputus dari sesi penyimpanan (menutup koneksi rekan), kuota mereka (batas penampil) tetap dikonsumsi selama 1 menit. Selama periode 1 menit ini, pemirsa dapat memanggil API ini dengan ID Klien yang sama untuk bergabung kembali dengan sesi tanpa menggunakan kuota penampil tambahan. Setelah 1 menit, kuota pemirsa dirilis dan tersedia bagi pemirsa lain untuk bergabung.

Browser

> **Important**
>
> Chrome adalah satu-satunya browser yang didukung.

1. Buka tab lain di Amazon Kinesis Video Streams dengan WebRTC SDK di halaman contoh. JavaScript Semua informasi dari halaman sebelumnya akan diisi secara otomatis. Jika tidak, lengkapi informasi berikut:

   - KVS Endpoint - Di bidang Region, pilih wilayah Anda.

     Misalnya, `us-west-2`.

   - AWS Kredensialnya

     Lengkapi bidang-bidang berikut:

     - ID Kunci Akses
     - Kunci Akses Rahasia
     - Token Sesi - Aplikasi sampel mendukung kredensyal sementara dan jangka panjang. Biarkan bidang ini kosong jika Anda menggunakan kredensi IAM jangka panjang. Lihat Kredensi keamanan sementara di IAM untuk informasi selengkapnya.

   - Saluran Sinyal - Di bidang Nama Saluran, ketik nama saluran pensinyalan yang Anda konfigurasikan sebelumnya. Untuk informasi selengkapnya, lihat the section called “Konfigurasikan tujuan”.
   - Trek - Pilih Kirim Audio. Perhatikan bahwa jika Kirim Video dicentang, secara otomatis akan dihapus centang saat memilih Mulai Penampil.
   - WebRTC Ingestion and Storage - Perluas node dan pilih Secara otomatis menentukan mode konsumsi. Opsi ini membuat aplikasi sampel memanggil DescribeMediaStorageConfigurationAPI untuk menentukan mode mana yang akan dijalankan.

2. Pilih Mulai Penampil.

   Aplikasi secara otomatis memanggil JoinStorageSessionAsViewerAPI segera setelah terhubung ke saluran pensinyalan untuk memicu penawaran SDP yang dikirim ke pemirsa dari sesi.

> **Note**
>
> Dengan peer-to-peer WebRTC, peserta pemirsa adalah rekan pengendali dan peserta utama adalah rekan yang dikendalikan. Dalam mode konsumsi WebRTC, sesi penyimpanan sekarang menjadi rekan pengendali. Setelah terhubung ke pensinyalan dan pemanggilan JoinStorageSessionAsViewer, penampil harus menanggapi penawaran SDP dan membuat koneksi ke sesi penyimpanan melalui WebRTC.

> **Note**
>
> Sesi penyimpanan hanya akan mengirim TURN kandidat. Saat mencalonkan pasangan kandidat ICE dari sudut pandang peserta, kandidat jarak jauh akan selalu bertipe. `relay`

Pemutaran media yang dicerna

Anda dapat menggunakan data media dengan melihatnya di konsol, atau dengan membuat aplikasi yang membaca data media dari aliran menggunakan Hypertext Live Streaming (HLS).

Lihat media di konsol

Di tab browser lain, buka file Konsol Manajemen AWS. Di Dasbor Kinesis Video Streams, pilih Streaming video.

Pilih nama aliran Anda dalam daftar aliran. Gunakan bilah pencarian, jika perlu.

Perluas bagian Pemutaran media. Jika video masih diunggah, itu akan ditampilkan. Jika unggahan telah selesai, pilih panah kiri ganda.

Konsumsi data media menggunakan HLS

Anda dapat membuat aplikasi klien yang mengkonsumsi data dari aliran video Kinesis menggunakan HLS. Untuk informasi tentang membuat aplikasi yang menggunakan data media menggunakan HLS, lihat Pemutaran Kinesis Video Streams.

Melihat media menggunakan aplikasi penampil media sampel

Amazon Kinesis Video Streams membuat contoh implementasi halaman web dari penampil media yang mampu memutar Kinesis Video Streams di HLS atau DASH. Anda dapat melihat kode sumber GitHubdan mencoba menggunakan halaman web yang dihosting.

1. Buka Penampil Media Streams Video Amazon Kinesis.
2. Lengkapi bidang-bidang berikut:
   - Wilayah - Pilih us-west-2.
   - AWS Kunci Akses
   - AWS Kunci Rahasia
   - Nama aliran
   - Mode Pemutaran - Pilih Langsung.
3. Pilih Mulai Pemutaran.

Connect ke sesi penyimpanan

Ikuti prosedur ini untuk membuat sesi penyimpanan dan memulai proses koneksi WebRTC. Peserta utama harus menelepon JoinStorageSession. Peserta pemirsa harus menelepon JoinStorageSessionAsViewer.

Ini akan membuat sesi penyimpanan mengirim penawaran SDP dan kandidat ICE melalui pensinyalan ke peserta utama yang terhubungthe section called “ConnectAsMaster”, atau peserta pemirsa tertentu yang terhubung melalui. the section called “ConnectAsViewer”

1. Dapatkan ARN dari saluran pensinyalan, karena ini adalah input yang diperlukan untuk langkah selanjutnya. Jika Anda sudah tahu ARN, lanjutkan dengan langkah berikutnya.

   Konsol Manajemen AWS

   1. Buka konsol Saluran Pensinyalan Kinesis Video Streams.
   2. Pilih nama saluran pensinyalan Anda.
   3. Di bawah tab Info saluran Pensinyalan, cari ARN untuk saluran pensinyalan Anda.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS Command Line Interface.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan perintah Describe-Signaling-Channel berikut menggunakan: AWS CLI

```
aws kinesisvideo describe-signaling-channel \
  --channel-name "YourChannelName" \
```

Responsnya akan terlihat seperti berikut:

```
{
    "ChannelInfo": {
        "ChannelName": "YourChannelName",
        "ChannelARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123",
        "ChannelType": "SINGLE_MASTER",
        "ChannelStatus": "ACTIVE",
        "CreationTime": "2024-07-07T23:28:24.941000-07:00",
        "SingleMasterConfiguration": {
        "MessageTtlSeconds": 60
    },
    "Version": "Ws0fZvFGXzEpuZ2CE1s9"
    }
}
```

Anda akan menemukan saluran pensinyalan ARN di `ChannelInfo` objek.

AWS SDK

Cuplikan kode ini menunjukkan cara membuat Kinesis Video Streams dengan saluran pensinyalan WebRTC menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama.

Anda dapat melihat contoh kode lengkap untuk JoinStorageSessionatau JoinStorageSessionAsViewer.

Buat klien Kinesis Video Streams. Ini digunakan untuk memanggil DescribeSignalingChannel API.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `DescribeSignalingChannel` API.

```
const describeSignalingChannelResponse = await kinesisVideoClient
    .describeSignalingChannel({
        ChannelName: 'YourChannelName',
    })
    .promise();
```

Simpan responsnya.

```
const channelARN =
    describeSignalingChannelResponse.ChannelInfo.ChannelARN;
```

2. Dapatkan titik akhir WEBRTC. Permintaan ke JoinStorageSessionatau JoinStorageSessionAsVieweruntuk saluran pensinyalan tertentu harus dilakukan ke titik akhir yang ditentukan.

   Konsol Manajemen AWS

   > **Note**
   >
   > Operasi ini saat ini tidak didukung di Kinesis Konsol Manajemen AWS Video Streams.

   Buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Ikuti instruksi di AWS CLI tab.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi lebih lanjut, lihat AWS Command Line Interfacedokumentasi.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan `Get-Signaling-Channel-Endpoint` perintah di AWS CLI:

```
aws kinesisvideo get-signaling-channel-endpoint \
  --channel-arn "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123" \
  --single-master-channel-endpoint-configuration
 "Protocols=['WEBRTC'],Role=MASTER" \
  --region "us-west-2"
```

Responsnya terlihat seperti berikut ini:

```
{
    "ResourceEndpointList": [
        {
            "Protocol": "WEBRTC",
            "ResourceEndpoint": "https://w-abcd1234.kinesisvideo.aws-
region.amazonaws.com"
        }
    ]
}
```

AWS SDK

Cuplikan kode ini menunjukkan cara memanggil `GetSignalingChannelEndpoint` API untuk Kinesis Video Streams dengan saluran pensinyalan WebRTC menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap untuk JoinStorageSessionatau JoinStorageSessionAsViewer.

Buat klien Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil DescribeSignalingChannel API.

Jika Anda membuat klien Kinesis Video Streams sebelumnya `DescribeSignalingChannel` untuk menelepon, Anda dapat menggunakan kembali klien yang sama.

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

Gunakan klien untuk memanggil `GetSignalingChannelEndpoint` API.

```
const getSignalingChannelEndpointResponse = await kinesisVideoClient
    .getSignalingChannelEndpoint({
        ChannelARN: channelARN,
        SingleMasterChannelEndpointConfiguration: {
            Protocols: ['WEBRTC'],
            Role: 'MASTER',
        },
    })
    .promise();
```

Simpan responsnya:

```
const webrtcEndpoint =
 getSignalingChannelEndpointResponse.ResourceEndpointList[0].ResourceEndpoint;
```

3. Gunakan saluran ARN dan titik akhir WEBRTC untuk melakukan panggilan API. Jika peserta terhubung dengan benar ke Kinesis Video Streams dengan WebRTC `ConnectAsMaster` Signaling `ConnectAsViewer` WebSocket APIs melalui or, penawaran SDP dan aliran pesan kandidat ICE akan dikirim sepanjang sesi penyimpanan ke peserta. WebSocket

Konsol Manajemen AWS

> **Note**
>
> Operasi ini saat ini tidak didukung di Kinesis Konsol Manajemen AWS Video Streams.

Buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Ikuti instruksi di AWS CLI tab.

AWS CLI

Verifikasi bahwa Anda telah AWS CLI menginstal dan mengkonfigurasi. Untuk informasi lebih lanjut, lihat AWS Command Line Interfacedokumentasi.

Untuk petunjuk penginstalan, lihat Panduan AWS Command Line Interface Pengguna. Setelah instalasi, konfigurasikan AWS CLI dengan kredensyal dan wilayah.

Atau, buka AWS CloudShell terminal, yang telah AWS CLI diinstal dan dikonfigurasi. Lihat Panduan AWS CloudShell Pengguna untuk informasi selengkapnya.

Jalankan `Join-Storage-Session` perintah untuk peserta master dalam AWS CLI menggunakan saluran ARN dan titik akhir WEBRTC dari langkah sebelumnya:

```
aws kinesis-video-webrtc-storage join-storage-session \
  --endpoint-url https://w-abcd1234.kinesisvideo.us-west-2.amazonaws.com \
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
  --region "us-west-2"
```

Setelah berhasil dijalankan, respons kosong dikembalikan dan tidak ada yang dicetak.

Dalam kasus peserta penampil, jalankan `Join-Storage-Session-As-Viewer` perintah berikut di AWS CLI menggunakan saluran ARN dan titik akhir WEBRTC dari sebelumnya:

```
aws kinesis-video-webrtc-storage join-storage-session-as-viewer \
  --endpoint-url https://w-abcd1234.kinesisvideo.us-west-2.amazonaws.com \
```

```
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
  --client-id "ExampleViewerClientID"
  --region "us-west-2"
```

Setelah berhasil dijalankan, respons kosong dikembalikan dan tidak ada yang dicetak.

AWS SDK

Cuplikan kode ini menunjukkan cara memanggil `JoinStorageSessionAsViewer` API `JoinStorageSession` atau untuk Kinesis Video Streams dengan saluran pensinyalan WebRTC menggunakan SDK untuk v2. AWS JavaScript Sintaks akan berbeda dari yang lain AWS SDKs, tetapi aliran umum akan sama. Lihat contoh kode lengkap untuk JoinStorageSessionatau JoinStorageSessionAsViewer.

Buat klien penyimpanan WebRTC Kinesis Video Streams. Ini adalah klien yang digunakan untuk memanggil `JoinStorageSession` atau `JoinStorageSessionAsViewer` dan berbeda dari klien Kinesis Video Streams yang dibuat pada langkah sebelumnya.

```
const webrtcStorageClientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2',
    endpoint: webrtcEndpoint
};
const kinesisVideoWebRTCStorageClient = new
 AWS.KinesisVideoWebRTCStorage(clientConfig);
```

Gunakan klien untuk memanggil `JoinStorageSession` API bagi peserta master.

```
await kinesisVideoWebRTCStorageClient
    .joinStorageSession({
        channelArn: channelARN,
    })
    .promise();
```

Dalam kasus peserta penampil, gunakan klien untuk memanggil `JoinStorageSessionAsViewer` API.

```
await kinesisVideoWebRTCStorageClient
    .joinStorageSessionAsViewer({
```

```
        channelArn: channelARN,
        clientId: "ExampleViewerClientID",
    })
    .promise();
```

Halaman web langsung dengan contoh kode ini tersedia di GitHub. Masukkan wilayah, AWS kredensyal, dan nama saluran pensinyalan Anda. Perluas simpul WebRTC Ingestion and Storage, dan hapus centang Secara otomatis menentukan mode konsumsi.

Alihkan penggantian manual ke ON dan pilih tombol Tampilkan untuk memanggil tombol JoinStorageSession API and/or Show secara manual untuk memanggil JoinStorageSessionAsViewer API secara manual.

Ketika Anda memilih Start Master atau Start Viewer, setelah aplikasi terhubung ke signaling via `ConnectAsMaster` or`ConnectAsViewer`, sebuah tombol akan muncul untuk memiliki sesi penyimpanan memulai koneksi WebRTC dengan peer alih-alih aplikasi memanggil API segera setelah menghubungkan ke signaling.

Memecahkan masalah yang berhubungan dengan sesi penyimpanan

Bagian ini memberikan panduan tentang masalah pemecahan masalah yang terkait dengan pengaturan dan konfigurasi penyimpanan untuk merekam aliran video.

Topik

- Mengontrol dan mengendalikan rekan-rekan
- Tinjau codec yang didukung
- Jika saluran tidak dipetakan ke aliran, juga akan membuang 400 InvalidArgumentException

Mengontrol dan mengendalikan rekan-rekan

Di WebRTC, peer pengendali memulai koneksi ke peer yang dikontrol dengan mengirimkan penawaran SDP. Untuk peer-to-peer sesi, peserta pemirsa memulai koneksi dengan mengirimkan penawaran kepada peserta utama melalui Signaling. Saat menghubungkan ke sesi penyimpanan untuk konsumsi WebRTC, sesi penyimpanan adalah rekan pengendali. Untuk peserta master,

mereka masih tetap menjadi peserta yang terkontrol. Namun, peserta pemirsa beralih dari kontrol ke kontrol.

Setelah menelepon JoinStorageSessionatau JoinStorageSessionAsViewer, semua peserta harus menanggapi dengan jawaban SDP dan bertukar kandidat ICE dengan sesi penyimpanan.

Untuk diagram urutan, lihat Memahami konsumsi dan penyimpanan WebRTC.

> **Important**
>
> Pesan sinyal yang diterima dari sesi penyimpanan tidak memiliki senderClientId bidang di JSON, yang berbeda dari peer-to-peer master, yang selalu menerima senderClientId bidang dengan penawaran SDP dan kandidat ICE.

Tinjau codec yang didukung

Saat mengirim jawaban SDP dan bertukar kandidat ICE dengan sesi penyimpanan, kami sarankan untuk menyertakan a `correlationId` dalam pesan. `correlationId`Menyertakan a dalam pesan memungkinkan sesi penyimpanan untuk mengembalikan `statusResponse` pesan. Pesan-pesan ini akan berisi pesan masukan, memungkinkan Anda untuk melacak pesan mana yang `statusResponse` dimiliki. `correlationId` Ini memungkinkan Anda menerima umpan balik langsung tentang mengapa jawaban SDP Anda ditolak.

Untuk informasi selengkapnya tentang `correlationId` dan `statusResponse`, lihat the section called “Penerimaan pesan asinkron”.

Salah satu alasan umum sesi penyimpanan mungkin menolak jawaban SDP adalah karena sesi penyimpanan tidak dapat menerima codec yang ditentukan dalam jawabannya. Sampel `statusResponse` mungkin terlihat seperti berikut:

```
{
  "correlationId": "1700186220273",
  "errorType": "InvalidArgumentException",
  "statusCode": "400",
  "success": false
}
```

Saat Anda meninjau konten jawaban SDP, tinjau baris yang dimulai `a=rtpmap` dan verifikasi bahwa codec cocok dengan codec yang didukung sesi penyimpanan. Di bawah ini adalah cuplikan dari contoh jawaban SDP yang berisi audio dan video opus. VP8

```
...
a=rtpmap:111 opus/48000/2
...
a=rtpmap:120 VP8/90000
...
```

Lihat JoinStorageSessiondaftar codec yang didukung.

Jika saluran tidak dipetakan ke aliran, juga akan membuang 400 InvalidArgumentException

Tinjau arah transceiver

Dalam jawaban SDP, tinjau arah transceiver video dan audio. Garis-garis di SDP terlihat seperti ini:

```
a=sendrecv
a=recvonly
```

Untuk peserta master, persyaratan berikut ada:

- Video H.264: kirim saja
- Opus audio: sendonly atau sendrecv

Untuk peserta pemirsa, persyaratan berikut diberlakukan:

- Video H.264: recvonly
- Opus audio: recvonly atau sendrecv

Jika persyaratan layanan tidak terpenuhi, StatusResponse dengan 400 IllegalArgumentException akan dikembalikan jika CorrelationID diberikan saat jawaban dikirim.

Saat menggunakan aplikasi sampel konsumsi WebRTC yang disediakan KVS:

- Peserta utama: Pastikan pengaturan “kirim video” dan “kirim audio” keduanya diaktifkan

- Peserta pemirsa: Pastikan pengaturan “kirim video” dinonaktifkan

Tinjau konversi kandidat ICE

Saat menerima kandidat ICE dari KVS Signaling SDK, aplikasi mungkin perlu mengubah string menjadi objek kandidat ICE.

Kandidat ICE yang diterima dari sesi penyimpanan tidak mengandung properti SDPMID, dan tidak datang dengan. senderClientId

Tinjau logika aplikasi Anda untuk menambahkan kandidat ICE yang diterima dari jarak jauh ke objek RTCPeer Connection aplikasi Anda.

Tinjau logika antrian kandidat ICE

Semua kandidat ICE yang diterima dari sesi penyimpanan perlu ditambahkan ke RTCPeer Koneksi melalui addIceCandidate API.

Meskipun sesi penyimpanan mengirimkan penawaran SDP sebelum kandidat ICE, karena sifat tidak sinkron dari pengiriman pesan Signaling, aplikasi dapat menerima kandidat ICE sebelum menerima penawaran.

Dalam hal ini, Anda perlu menerapkan buffer sementara untuk menahan kandidat ICE.

Logika yang diterapkan dalam aplikasi sampel KVS adalah sebagai berikut:

1. Sampel memelihara peta senderClientId ke RTCPeer Koneksinya.
2. Sampel mempertahankan peta lain dari daftar Kandidat Es yang tertunda. senderClientId
3. Ketika kandidat ICE diterima, periksa PeerConnection peta. Jika PeerConnection peta belum memiliki koneksi, tambahkan ke daftar yang tertunda. Jika tidak, tambahkan kandidat ICE ke PeerConnection.
4. Ketika penawaran diterima, itu membuat RTCPeer Koneksi dan menambahkannya ke PeerConnection peta. Kemudian, tambahkan semua kandidat ICE dari antrian yang tertunda ke ini PeerConnection.

IPv6Menggunakan/Dual-Stack endpoint dengan Amazon Kinesis Video WebRTC

Anda dapat mengonfigurasi WebRTC Video Amazon Kinesis untuk digunakan IPv6 untuk operasi bidang kontrol dan bidang data. Ini memungkinkan aplikasi Anda untuk berkomunikasi dengan layanan WebRTC Kinesis Video menggunakan alamat melalui titik akhir dual-stack. IPv6

> **Note**
>
> IPv6 dukungan memerlukan versi SDK dan pengaturan konfigurasi tertentu. Pastikan versi Kinesis Video WebRTC SDK dan Amazon Web Services SDK mendukung titik akhir dual-stack. IPv6 Titik akhir dual-stack mendukung keduanya IPv4 dan IPv6 lalu lintas dan tersedia untuk beberapa layanan di beberapa Wilayah.

Amazon Kinesis Video WebRTC mendukung IPv6 melalui titik akhir tumpukan ganda untuk aplikasi master dan penampil. Anda dapat mengonfigurasi aplikasi Anda untuk menggunakan titik akhir IPv6 / Dual-Stack untuk panggilan API bidang kontrol dan operasi bidang data.

Konfigurasikan SDK Amazon Web Services untuk IPv6 -Dual-Stack Endpoint

Jika Anda menggunakan Amazon Web Services SDK untuk memanggil APIs bidang kontrol WebRTC Video Kinesis dalam penyiapan produksi, Anda dapat mengaktifkannya dengan mengonfigurasi titik akhir tumpukan ganda. IPv6 Amazon Web Services SDK menyediakan beberapa metode standar untuk mengaktifkan titik akhir dual-stack.

> **Important**
>
> Ketika titik akhir dual-stack diaktifkan, SDK mencoba menggunakan titik akhir dual-stack untuk membuat permintaan jaringan. Jika titik akhir tumpukan ganda tidak ada untuk layanan atau Wilayah, permintaan gagal.

Menggunakan variabel lingkungan

Tetapkan variabel lingkungan berikut untuk mengaktifkan titik akhir IPv6 dual-stack:

```
export AWS_USE_DUALSTACK_ENDPOINT=true
```

Menggunakan file konfigurasi Amazon Web Services

Tambahkan setelan berikut ke file konfigurasi Amazon Web Services Anda (~/.aws/config):

```
[default]
use_dualstack_endpoint = true
```

Gunakan properti sistem JVM (hanya Java dan Kotlin SDKs)

Untuk aplikasi Java dan Kotlin, tetapkan properti sistem JVM berikut:

```
-Daws.useDualstackEndpoint=true
```

Atau secara terprogram dalam kode Java Anda:

```
System.setProperty("aws.useDualstackEndpoint", "true");
```

Dukungan SDK

Amazon Web Services berikut SDKs mendukung konfigurasi titik akhir dual-stack:

SDK	Didukung	Metode konfigurasi
AWS CLI v2	Ya	Variabel lingkungan, file konfigurasi
SDK untuk C++	Ya	Variabel lingkungan, file konfigurasi
SDK for Go V2 (1.x)	Ya	Variabel lingkungan, file konfigurasi

SDK	Didukung	Metode konfigurasi
SDK for Go 1.x (V1)	Ya	Variabel lingkungan, file konfigurasi
SDK untuk Java 2.x	Ya	Variabel lingkungan, file konfigurasi, properti JVM
SDK for Java 1.x	Tidak	Tidak didukung
SDK untuk 3.x JavaScript	Ya	Variabel lingkungan, file konfigurasi
SDK untuk 2.x JavaScript	Ya	Variabel lingkungan, file konfigurasi
SDK untuk Kotlin	Ya	Variabel lingkungan, file konfigurasi, properti JVM
SDK for .NET 4.x	Ya	Variabel lingkungan, file konfigurasi
SDK for .NET 3.x	Ya	Variabel lingkungan, file konfigurasi
SDK for PHP 3.x	Ya	Variabel lingkungan, file konfigurasi
SDK untuk Python (Boto3)	Ya	Variabel lingkungan, file konfigurasi
SDK for Ruby 3.x	Ya	Variabel lingkungan, file konfigurasi
SDK for Rust	Ya	Variabel lingkungan, file konfigurasi
SDK para Swift	Ya	Variabel lingkungan, file konfigurasi

SDK	Didukung	Metode konfigurasi
Alat untuk PowerShell V5	Ya	Variabel lingkungan, file konfigurasi
Alat untuk PowerShell V4	Ya	Variabel lingkungan, file konfigurasi

Setelah mengonfigurasi titik akhir tumpukan ganda, Amazon Web Services SDK secara otomatis menggunakan IPv6 titik akhir saat memanggil bidang kontrol WebRTC Video Kinesis. APIs

Konfigurasikan SDK WebRTC Video Kinesis untuk/Dual-Stack Endpoint IPv6

Kinesis Video WebRTC SDK menyediakan opsi konfigurasi dual-stack untuk operasi bidang kontrol dan bidang data. Pengaturan ini berfungsi dengan konfigurasi titik akhir tumpukan ganda Amazon Web Services SDK.

Konfigurasikan WebRTC C SDK

Untuk menggunakan titik akhir AWS KVS dual-stack dan mencoba mengumpulkan kandidat IPv6 ICE, tetapkan variabel lingkungan berikut:

```
export KVS_DUALSTACK_ENDPOINTS=ON
```

Dalam mode dual-stack, ICE gathering akan mencoba memasukkan IPv6 kandidat, tetapi kompatibilitas pada akhirnya tergantung pada konfigurasi jaringan lokal dan kemampuan rekan-rekan penerima.

Untuk menonaktifkan mode dual-stack, hapus setel variabel lingkungan:

```
unset KVS_DUALSTACK_ENDPOINTS
```

Resolusi titik akhir bidang data

Untuk operasi bidang data, Kinesis Video WebRTC SDK menggunakan GetSignalingChannelEndpoint API untuk mengambil titik akhir bidang data /Dual-stack yang sesuai. IPv6 SDK secara otomatis meminta IPv6 /Dual-stack endpoint ketika /Dual-stack dikonfigurasi. IPv6

> **Important**
>
> GetSignalingChannelEndpoint API telah diperbarui untuk mendukung IPv6 titik akhir. Pastikan Anda menggunakan versi SDK yang kompatibel yang mendukung fungsi ini.

Konfigurasikan AWS CLI untuk IPv6 /Dual-Stack

Jika Anda menggunakan operasi WebRTC AWS CLI untuk Kinesis Video (biasanya untuk proof-of-concept pekerjaan), Anda dapat mengaktifkan dengan mengonfigurasi titik akhir dual-stack. IPv6

Gunakan variabel lingkungan

```
export AWS_USE_DUALSTACK_ENDPOINT=true
```

Menggunakan file konfigurasi Amazon Web Services

Tambahkan yang berikut ini ke file AWS CLI konfigurasi Anda (~/.aws/config):

```
[default]
use_dualstack_endpoint = true
```

Setelah mengonfigurasi titik akhir dual-stack, akan AWS CLI menggunakan titik akhir IPv6 dual-stack untuk semua panggilan Amazon Web Services, termasuk operasi WebRTC Kinesis Video.

Pertimbangan-pertimbangan

Penyedia kredensi IoT

Jika Anda menggunakan kredensi IoT untuk otentikasi:

- Dukungan titik akhir kredensi IoT IPv6
- Konfigurasikan titik akhir tumpukan ganda menggunakan metode konfigurasi SDK Amazon Web Services standar yang dijelaskan sebelumnya
- Alur kredensyal IoT terpisah dari konfigurasi khusus WebRTC Video Kinesis IPv6

Persyaratan jaringan

- Pastikan infrastruktur jaringan Anda mendukung IPv6 konektivitas
- Verifikasi bahwa grup keamanan dan jaringan Anda ACLs mengizinkan IPv6 lalu lintas
- Uji konektivitas ke IPv6 titik akhir Amazon Web Services dari lingkungan penerapan
- Titik akhir tumpukan ganda tersedia untuk beberapa layanan di beberapa wilayah—Verifikasi ketersediaan untuk Wilayah target Anda

Kompatibilitas SDK

- Pastikan Anda menggunakan versi Amazon Web Services SDK yang didukung (lihat tabel kompatibilitas)
- Amazon Web Services SDK for Java 1.x tidak mendukung konfigurasi titik akhir dual-stack
- Untuk SDK for Go 1.x (V1), Anda harus mengaktifkan pemuatan dari file konfigurasi untuk menggunakan pengaturan file konfigurasi bersama

Pengujian dan validasi

Sebelum Anda menerapkan aplikasi IPv6 WebRTC Kinesis Video yang diaktifkan ke produksi:

- Uji operasi bidang kontrol (pembuatan saluran, penghapusan, daftar)
- Verifikasi operasi pesawat data (STUN, TURN dan WebRTC Signaling)
- Verifikasi pembentukan sesi peer-to-peer streaming yang berhasil
- Validasi kinerja dan konektivitas di lingkungan jaringan Anda
- Jalankan tes kenari untuk memastikan fungsionalitas yang konsisten IPv6
- Uji perilaku failover saat titik akhir tumpukan ganda tidak tersedia

Pelanggan yang terkena dampak peningkatan untuk menyertakan IPv6

Saat Anda mengaktifkan IPv6 WebRTC Video Amazon Kinesis, ada beberapa area di mana Anda mungkin perlu memperbarui konfigurasi dan kebijakan yang ada untuk memastikan fungsionalitas lanjutan. Bagian ini menguraikan area utama yang memerlukan perhatian saat beralih ke IPv6 titik akhir yang diaktifkan.

Kebijakan IAM dan pemfilteran alamat IP

Jika Anda menggunakan pemfilteran alamat IP sumber dalam kebijakan pengguna IAM, kebijakan peran, atau kebijakan berbasis sumber daya, Anda perlu memperbarui kebijakan ini untuk menyertakan rentang alamat. IPv6

> **Important**
>
> Kebijakan IAM yang ada yang menggunakan blok IPv4 CIDR dalam IpAddress atau NotIpAddress kondisi tidak akan berfungsi secara otomatis dengan IPv6 alamat. Anda harus secara eksplisit menambahkan IPv6 rentang untuk mempertahankan kontrol akses.

Contoh pembaruan kebijakan IAM untuk IPv6:

```
{
  "Version": "2012-10-17"       ,
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "kinesisvideo:*",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": [
            "192.0.2.0/24",
            "203.0.113.0/24",
            "2001:db8::/32"
          ]
        }
```

```
      }
    }
  ]
}
```

Pertimbangan utama untuk pembaruan kebijakan IAM:

- Tambahkan blok IPv6 CIDR di samping rentang yang ada IPv4
- Gunakan kunci SourceIp kondisi aws: untuk keduanya IPv4 dan IPv6 alamat
- Menguji kebijakan di lingkungan non-produksi sebelum menerapkan
- Pertimbangkan untuk menggunakan aws: RequestedRegion sebagai kondisi tambahan untuk keamanan yang ditingkatkan

Grup keamanan jaringan dan daftar kontrol akses

Jika Anda menjalankan aplikasi WebRTC Kinesis Video di instans Amazon atau layanan Amazon Web Services EC2 lainnya, Anda perlu memperbarui grup keamanan dan jaringan untuk mengizinkan lalu lintas. ACLs IPv6

- Grup keamanan - Tambahkan aturan masuk dan keluar untuk blok IPv6 CIDR (:: /0 untuk semua IPv6 lalu lintas, atau rentang tertentu) IPv6
- Jaringan ACLs - Perbarui jaringan tingkat subnet ACLs untuk memungkinkan IPv6 lalu lintas pada port yang diperlukan
- Tabel rute - Pastikan tabel rute VPC Anda menyertakan rute IPv6 lalu lintas untuk mencapai gateway internet atau gateway NAT

Pencatatan log dan pemantauan

IPv6 alamat memiliki format yang berbeda dari IPv4 alamat, yang dapat memengaruhi sistem pencatatan, pemantauan, dan analitik Anda.

AWS CloudTrail log

AWS CloudTrail log akan berisi IPv6 alamat di IPAddress bidang sumber saat permintaan dibuat IPv6. Perbarui alat dan skrip penguraian log Anda untuk menangani format IPv6 alamat.

Contoh IPv6 alamat di AWS CloudTrail log:

```
{
  "sourceIPAddress": "2001:db8::1",
  "eventName": "CreateSignalingChannel",
  "eventSource": "kinesisvideo.amazonaws.com"
}
```

Log aplikasi

Jika aplikasi Anda mencatat alamat IP klien atau melakukan analitik berbasis IP, pastikan infrastruktur logging Anda dapat menangani IPv6 alamat:

- Perbarui ekspresi reguler penguraian log agar sesuai dengan format IPv6
- Ubah skema database jika Anda menyimpan alamat IP dengan bidang panjang tetap
- Perbarui kueri analitik dan dasbor untuk bekerja dengan alamat IPv6
- Pertimbangkan untuk menggunakan pustaka normalisasi alamat IP untuk penanganan yang konsisten

Pemantauan dan peringatan

Perbarui sistem pemantauan dan peringatan Anda untuk memperhitungkan IPv6 lalu lintas:

- CloudWatch Metrik dan alarm Amazon yang memfilter berdasarkan alamat IP
- Metrik khusus yang melacak pola berbasis IP
- Alat pemantauan keamanan yang menganalisis pola lalu lintas
- Layanan geolokasi yang memetakan alamat IP ke lokasi

Integrasi pihak ketiga

Tinjau dan perbarui layanan dan alat pihak ketiga yang terintegrasi dengan aplikasi WebRTC Kinesis Video Anda:

- Jaringan pengiriman konten (CDNs) - Pastikan konfigurasi CDN mendukung IPv6 jika Anda menggunakan CDNs untuk distribusi video
- Load balancer - Konfigurasikan Application Load Balancers atau Network Load Balancer untuk menangani lalu lintas IPv6
- Layanan DNS - Perbarui catatan DNS untuk menyertakan catatan AAAA untuk alamat IPv6

- Firewall dan peralatan keamanan - Konfigurasikan peralatan keamanan jaringan untuk memungkinkan IPv6 lalu lintas
- Alat pemantauan — Verifikasi bahwa alat pemantauan dan analitik pihak ketiga mendukung format IPv6 alamat

Pembaruan kode aplikasi

Tinjau kode aplikasi Anda untuk asumsi IPv4 spesifik yang mungkin perlu diperbarui:

- Validasi alamat IP - Perbarui validasi input untuk menerima IPv6 format alamat
- Skema database — Pastikan bidang alamat IP dapat menyimpan IPv6 alamat (biasanya membutuhkan ukuran bidang yang lebih besar)
- File konfigurasi - Perbarui IPv4 alamat hardcode atau blok CIDR
- Pustaka klien — Verifikasi bahwa klien HTTP dan pustaka jaringan mendukung IPv6
- Penanganan kesalahan - Perbarui penanganan kesalahan untuk memperhitungkan kesalahan jaringan IPv6 -spesifik

Pengujian dan validasi

Sebelum mengaktifkan IPv6 produksi, uji aplikasi dan infrastruktur Anda secara menyeluruh:

- Pengujian konektivitas — Verifikasi bahwa semua komponen dapat berkomunikasi IPv6
- Pengujian kinerja - Bandingkan IPv6 dan IPv4 kinerja untuk mengidentifikasi masalah apa pun
- Pengujian keamanan - Validasi bahwa kontrol keamanan bekerja dengan benar dengan lalu lintas IPv6
- Pengujian failover - Uji perilaku saat IPv6 konektivitas tidak tersedia
- Analisis log — Verifikasi bahwa sistem pencatatan dan pemantauan menangani IPv6 alamat dengan benar
- Pengujian integrasi - Uji semua integrasi pihak ketiga dengan diaktifkan IPv6

Strategi migrasi

Pertimbangkan untuk menerapkan pendekatan bertahap untuk IPv6 adopsi:

1. Tahap penilaian - Inventarisasi semua sistem dan identifikasi IPv6 kesiapan

2. Tahap persiapan - Perbarui kebijakan, grup keamanan, dan kode aplikasi
3. Tahap pengujian - Aktifkan IPv6 dalam lingkungan pengembangan dan pementasan
4. Fase percontohan - Aktifkan IPv6 subset lalu lintas produksi
5. Penyebaran penuh - Secara bertahap meningkatkan IPv6 lalu lintas hingga sepenuhnya digunakan
6. Fase pemantauan - Terus memantau masalah dan mengoptimalkan kinerja

Pemecahan masalah

Masalah umum

- Kegagalan koneksi - Verifikasi konektivitas IPv6 jaringan dan resolusi DNS
- Kesalahan SDK — Pastikan Anda menggunakan versi SDK yang kompatibel yang mendukung titik akhir dual-stack
- Masalah otentikasi — Konfirmasikan bahwa kebijakan dan kredensional IAM berfungsi dengan titik akhir IPv6
- Titik akhir tidak tersedia — Jika titik akhir tumpukan ganda tidak ada untuk layanan atau Wilayah, permintaan gagal

Langkah-langkah verifikasi

- Periksa apakah AWS_USE _dualStack_EndPoint=true disetel atau use_dualstack_endpoint = true ada di file konfigurasi Anda
- Verifikasi bahwa flag konfigurasi Kinesis Video WebRTC SDK disetel dengan benar IPv6
- Uji konektivitas jaringan ke titik IPv6 akhir Amazon Web Services
- Tinjau log aplikasi untuk pesan kesalahan IPv6 -spesifik
- Konfirmasikan bahwa Wilayah Anda mendukung titik akhir dual-stack untuk Kinesis Video WebRTC

Validasi konfigurasi

Anda dapat memverifikasi konfigurasi titik akhir dual-stack Anda dengan memeriksa:

- Variabel lingkungan: echo $ AWS_USE _DUALSTACK_ENDPOINT

- File konfigurasi Amazon Web Services: cat ~/.aws/config | grep use_dualstack_endpoint
- Properti JVM (Java): Periksa properti sistem di log aplikasi Anda

Untuk dukungan dan pemecahan masalah tambahan, lihat AWS dokumentasi atau kontak. AWS

Multiviewer

Amazon Kinesis Video Streams Multiviewer adalah solusi WebRTC berbasis cloud yang memungkinkan beberapa pemirsa untuk secara bersamaan bergabung dengan sesi streaming video real-time dari satu perangkat kamera. Fitur ini mengatasi bandwidth dan kendala komputasi perangkat edge dengan memproses aliran video di cloud daripada mengharuskan kamera mengirim aliran terpisah ke setiap pemirsa.

Topik

Ikhtisar

Koneksi peer-to-peer WebRTC tradisional memerlukan perangkat kamera untuk mengirim video secara terpisah ke setiap pemirsa, yang dapat dengan cepat membanjiri perangkat dengan bandwidth terbatas atau kapasitas komputasi. Multiviewer menyelesaikan ini dengan menggunakan “mixer” berbasis cloud yang:

- Menerima satu aliran video dari perangkat kamera
- Memproses dan meneruskan streaming ke beberapa pemirsa
- Menangani pencampuran audio untuk percakapan multi-peserta
- Mempertahankan komputasi perangkat edge dan kapasitas bandwidth

Multiviewer sangat berharga untuk kasus penggunaan termasuk:

- Keamanan Rumah Pintar: Beberapa anggota rumah tangga dapat melihat umpan kamera secara bersamaan tanpa menurunkan kinerja
- Keamanan Perusahaan: Tim keamanan dapat memantau feed secara bersamaan

- Pemantauan Otomotif: Manajer dan pengontrol armada dapat melihat kamera kendaraan secara bersamaan
- Robotika & Drone: Beberapa operator dapat memantau sistem otonom
- Pendidikan/Proctoring: Beberapa proctor dapat mengamati sesi tes
- Telehealth: Tim kesehatan dapat berpartisipasi dalam konsultasi jarak jauh

Persyaratan dan Sumber Daya

Untuk menggunakan Multiviewer, Anda memerlukan sumber daya berikut:

- Kinesis Video Streams Stream: Tujuan untuk menelan dan menyimpan konten video dan audio
- WebRTC Signaling Channel: Mengaktifkan koneksi ke perangkat menggunakan KVS WebRTC SDK
- Konfigurasi Penyimpanan Media: Menautkan Stream dan Saluran menggunakan UpdateMediaStorageConfiguration API

> **Important**
>
> Multiviewer saat ini hanya tersedia bila digabungkan dengan WebRTC Ingestion. Baik master maupun penampil dapat memulai sesi konsumsi WebRTC, dan trek video dan audio disimpan secara bersamaan dalam Stream Streaming Video Kinesis sambil didistribusikan ke beberapa pemirsa.

Persyaratan perangkat:

- Perangkat kamera harus mendukung SDK WebRTC KVS
- Aplikasi penampil harus menggunakan KVS WebRTC SDK
- Semua peserta terhubung ke saluran pensinyalan yang sama

Persyaratan Lacak:

- Peserta master: Baik trek audio dan video diperlukan untuk konsumsi WebRTC
- Peserta pemirsa: Dapat mengirim trek audio opsional atau tidak ada trek sama sekali. Pemirsa tidak dapat mengirim trek video

Menyiapkan Multiviewer

Ikuti langkah-langkah berikut untuk mengonfigurasi Multiviewer untuk aplikasi Anda:

1. Buat Sumber Daya yang Diperlukan

   Buat Stream Streaming Video Kinesis dan Saluran Pensinyalan WebRTC menggunakan konsol, CLI, atau. SDKs Lihat the section called “Buat aliran video” dan the section called “Buat saluran pensinyalan” untuk instruksi terperinci.
2. Sumber Daya Tautan

   Gunakan UpdateMediaStorageConfiguration API untuk menautkan Stream dan Saluran Anda. Konfigurasi ini memungkinkan fungsionalitas Multiviewer. Lihat the section called “Konfigurasikan tujuan” untuk detail implementasi.
3. Konfigurasikan Aplikasi Kamera

   Implementasikan aplikasi kamera menggunakan KVS WebRTC SDK untuk memanggil API. JoinStorageSession Ini memulai sesi konsumsi yang dapat diikuti oleh pemirsa lain.
4. Konfigurasikan Aplikasi Penampil

   Terapkan aplikasi penampil menggunakan KVS WebRTC SDK untuk memanggil API. JoinStorageSessionAsViewer Beberapa pemirsa dapat bergabung dengan sesi yang sama secara bersamaan.

Integrasi dengan Ingestion

Multiviewer dibangun di atas kemampuan WebRTC Ingestion, yang diluncurkan pada tahun 2023. Integrasi ini memberikan beberapa manfaat:

- Perekaman Otomatis: Semua sesi multiviewer direkam secara otomatis ke Kinesis Video Streams untuk pemutaran dan analisis nanti
- Cloud Processing: Pemrosesan video terjadi di cloud, mengurangi beban komputasi pada perangkat edge
- Arsitektur yang Dapat Diskalakan: Pendekatan berbasis cloud dapat menangani beberapa sesi multiviewer bersamaan
- Pengalaman yang Konsisten: Pemirsa menerima streaming berkualitas tinggi yang sama terlepas dari kondisi jaringan mereka

Alur kerja untuk sesi multiviewer tipikal dengan konsumsi adalah:

1. Panggilan perangkat kamera JoinStorageSession untuk mulai menelan video ke cloud
2. Aliran video diproses dan disimpan dalam Stream Streams Video Kinesis yang dikonfigurasi
3. Beberapa perangkat penampil memanggil JoinStorageSessionAsViewer untuk bergabung dengan sesi
4. Cloud mixer mendistribusikan aliran video ke semua pemirsa yang terhubung
5. Audio dari semua peserta dicampur dan didistribusikan dengan tepat

Operasi API

Multiviewer menggunakan operasi API yang sama dengan WebRTC Ingestion. Kuncinya APIs meliputi:

- UpdateMediaStorageConfiguration- Menautkan saluran pensinyalan ke aliran untuk konsumsi
- JoinStorageSession- Memulai sesi konsumsi dari perangkat kamera
- JoinStorageSessionAsViewer- Memungkinkan pemirsa untuk bergabung dengan sesi konsumsi aktif
- DescribeMediaStorageConfiguration- Mengambil konfigurasi penyimpanan media saat ini

Untuk contoh penggunaan API yang mendetail, lihatthe section called “Buat koneksi WebRTC dengan sesi penyimpanan”.

Praktik Terbaik

Ikuti praktik terbaik ini saat menerapkan Multiviewer:

- Optimalkan untuk Perangkat Edge: Gunakan Multiviewer secara khusus saat Anda membutuhkan lebih dari 2-3 pemirsa bersamaan, karena di sinilah batasan perangkat edge biasanya menjadi jelas
- Monitor Session Health: Menerapkan pemantauan untuk melacak kualitas sesi dan koneksi penampil
- Menangani Kegagalan Koneksi: Menerapkan logika coba lagi untuk koneksi kamera dan penampil
- Manajemen Audio: Pertimbangkan untuk mematikan pemirsa secara default untuk mencegah umpan balik audio dalam sesi besar

- Pembersihan Sumber Daya: Pastikan pembersihan koneksi WebRTC yang tepat saat pemirsa meninggalkan sesi

Keamanan

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda akan 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 menggambarkan hal ini sebagai keamanan dari cloud dan keamanan di 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. Efektivitas keamanan kami diuji dan diverifikasi secara rutin oleh auditor pihak ketiga sebagai bagian dari program kepatuhan AWS. Untuk mempelajari tentang program kepatuhan yang berlaku untuk Kinesis Video Streams AWS , lihat Layanan dalam Lingkup berdasarkan Program Kepatuhan.
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor-faktor lain termasuk sensitivitas data, kebutuhan organisasi, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon Kinesis Video Streams dengan WebRTC. Topik berikut menunjukkan cara mengonfigurasi Amazon Kinesis Video Streams dengan WebRTC untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga akan mempelajari cara menggunakan AWS layanan lain yang dapat membantu Anda memantau dan mengamankan Amazon Kinesis Video Streams Anda dengan sumber daya WebRTC.

Topik

Kontrol akses ke Amazon Kinesis Video Streams dengan sumber daya WebRTC AWS Identity and Access Management

Dengan menggunakan AWS Identity and Access Management (IAM) dengan Amazon Kinesis Video Streams dengan WebRTC, Anda dapat mengontrol apakah pengguna di organisasi Anda dapat melakukan tugas menggunakan Kinesis Video Streams tertentu dengan operasi WebRTC API dan apakah mereka dapat menggunakan sumber daya tertentu. AWS

Untuk informasi selengkapnya tentang IAM, lihat berikut ini:

- AWS Identity and Access Management (IAM)
- Memulai dengan IAM
- Panduan Pengguna IAM

Daftar Isi

Sintaksis kebijakan

kebijakan IAM adalah dokumen JSON yang terdiri dari satu atau beberapa pernyataan. Setiap pernyataan memiliki struktur sebagai berikut:

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
```

```
  ]
}
```

Ada berbagai elemen yang membentuk pernyataan:

- Efek: Efek bisa berupa `Allow` atau `Deny`. Secara default, para pengguna IAM tidak memiliki izin untuk menggunakan sumber daya dan tindakan API, jadi semua permintaan akan ditolak. izin eksplisit akan menggantikan izin default. penolakan eksplisit akan menggantikan izin apa pun.
- Tindakan: Tindakan adalah tindakan API tertentu yang Anda izinkan atau tolak.
- Sumber Daya: Sumber daya yang dipengaruhi oleh tindakan. Untuk menentukan sumber daya dalam sebuah pernyataan kebijakan IAM, gunakan Amazon Resource Name (ARN).
- Syarat: Syarat bersifat opsional. Syarat-syarat ini dapat digunakan untuk mengendalikan kapan kebijakan Anda berlaku.

Saat Anda membuat dan mengelola kebijakan IAM, Anda mungkin ingin menggunakan IAM Policy Generator dan IAM Policy Simulator.

Tindakan API

Dalam pernyataan kebijakan IAM, Anda dapat menentukan tindakan API apa pun dari layanan apa pun yang mendukung IAM. Untuk Kinesis Video Streams dengan WebRTC, gunakan awalan berikut dengan nama aksi API:. `kinesisvideo:` Misalnya:`kinesisvideo:CreateSignalingChannel`,`kinesisvideo:ListSignalingChannels`, dan`kinesisvideo:DescribeSignalingChannel`.

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan tindakan-tindakan tersebut menggunakan koma seperti berikut:

```
"Action": ["kinesisvideo:action1", "kinesisvideo:action2"]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard. Misalnya, Anda dapat menentukan semua tindakan yang namanya dimulai dengan kata “Dapatkan” sebagai berikut:

```
"Action": "kinesisvideo:Get*"
```

Untuk menentukan semua operasi Kinesis Video Streams, gunakan kartu liar asterisk (*) sebagai berikut:

```
"Action": "kinesisvideo:*"
```

Untuk daftar lengkap tindakan API Kinesis Video Streams, lihat referensi API Kinesis Video Streams.

Nama Sumber Daya Amazon (ARNs)

Setiap pernyataan kebijakan IAM berlaku untuk sumber daya yang Anda tentukan dengan menggunakan ARNs.

Gunakan format sumber daya ARN berikut untuk Kinesis Video Streams:

```
arn:aws:kinesisvideo:region:account-id:channel/channel-name/code
```

Contoh:

```
"Resource": arn:aws:kinesisvideo::*:111122223333:channel/my-channel/0123456789012
```

Anda bisa mendapatkan ARN dari saluran menggunakan. DescribeSignalingChannel

Berikan akses akun IAM lainnya ke aliran video Kinesis

Anda mungkin perlu memberikan izin ke akun IAM lain untuk melakukan operasi pada Kinesis Video Streams dengan saluran pensinyalan WebRTC. Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

Contoh kebijakan

Contoh kebijakan berikut menunjukkan bagaimana Anda dapat mengontrol akses pengguna ke Kinesis Video Streams Anda dengan saluran WebRTC.

Example 1: Izinkan pengguna mendapatkan data dari saluran pensinyalan apa pun

Kebijakan ini memungkinkan pengguna atau grup untuk melakukan`DescribeSignalingChannel`, `GetSignalingChannelEndpointListSignalingChannels`, dan `ListTagsForResource` operasi pada saluran pensinyalan apa pun.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:Describe*",
                "kinesisvideo:Get*",
                "kinesisvideo:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Example 2: Izinkan pengguna membuat saluran pensinyalan

Kebijakan ini memungkinkan pengguna atau grup untuk melakukan `CreateSignalingChannel` operasi.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:CreateSignalingChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

Example 3: Izinkan pengguna akses penuh ke semua Kinesis Video Streams dan Kinesis Video Streams dengan sumber daya WebRTC

Kebijakan ini memungkinkan pengguna atau grup untuk melakukan operasi Kinesis Video Streams pada sumber daya apa pun. Kebijakan ini sesuai untuk administrator.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kinesisvideo:*",
            "Resource": "*"
        }
    ]
}
```

Example 4: Izinkan pengguna mendapatkan data dari saluran pensinyalan tertentu

Kebijakan ini memungkinkan pengguna atau grup untuk mendapatkan data dari saluran pensinyalan tertentu.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kinesisvideo:DescribeSignalingChannel",
            "Resource": "arn:aws:kinesisvideo:us-west-2:123456789012:channel/
channel_name/0123456789012"
        }
    ]
}
```

Validasi kepatuhan untuk Amazon Kinesis Video Streams dengan WebRTC

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.

Ketahanan di Amazon Kinesis Video Streams dengan WebRTC

Infrastruktur AWS global dibangun di sekitar Wilayah AWS dan Availability Zones. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Anda dapat menggunakan Availability Zones untuk merancang dan mengoperasikan aplikasi dan database yang secara otomatis gagal di antara Availability Zone tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur biasa yang terdiri dari satu atau beberapa pusat data.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur AWS Global.

Keamanan infrastruktur di Kinesis Video Streams dengan WebRTC

Sebagai layanan terkelola, Kinesis Video Streams (termasuk kemampuan WebRTC) dilindungi AWS oleh prosedur keamanan jaringan global yang dijelaskan dalam whitepaper Amazon Web Services: Overview of Security Processes.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Kinesis Video Streams melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2 atau versi yang lebih baru. Kami merekomendasikan TLS 1.3 atau versi yang lebih baru. Klien juga harus

mendukung cipher suite dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan principal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Praktik terbaik keamanan untuk Amazon Kinesis Video Streams dengan WebRTC

Amazon Kinesis Video Streams (termasuk kemampuan WebRTC) menyediakan sejumlah fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau tidak memadai untuk lingkungan Anda, perlakukan itu sebagai pertimbangan yang bermanfaat, bukan sebagai resep.

Untuk praktik terbaik keamanan untuk perangkat jarak jauh, lihat Praktik Terbaik Keamanan untuk Agen Perangkat.

Terapkan akses hak akses paling rendah

Saat memberikan izin, Anda memutuskan siapa yang mendapatkan izin apa untuk sumber daya Kinesis Video Streams mana. Anda memungkinkan tindakan tertentu yang ingin Anda lakukan di sumber daya tersebut. Oleh karena itu, Anda harus memberikan hanya izin yang diperlukan untuk melaksanakan tugas. Menerapkan akses hak istimewa yang terkecil adalah hal mendasar dalam mengurangi risiko keamanan dan dampak yang dapat diakibatkan oleh kesalahan atau niat jahat.

Misalnya, produser yang mengirimkan data ke Kinesis Video Streams `GetStreamingEndpoint` hanya `PutMedia` membutuhkan,, dan. `DescribeStream` Jangan berikan izin aplikasi produsen untuk semua tindakan (`*`), atau untuk tindakan lain seperti`GetMedia`.

Untuk informasi selengkapnya, lihat Menerapkan izin hak istimewa paling sedikit.

Gunakan IAM role

Aplikasi produser dan klien harus memiliki kredensi yang valid untuk mengakses aliran video Kinesis. Anda tidak boleh menyimpan AWS kredensi secara langsung di aplikasi klien atau di bucket Amazon

S3. Ini adalah kredensial jangka panjang yang tidak dirotasi secara otomatis dan dapat menimbulkan dampak bisnis yang signifikan jika dibobol.

Sebagai gantinya, Anda harus menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi produser dan klien Anda untuk mengakses aliran video Kinesis. Ketika Anda menggunakan peran, Anda tidak perlu menggunakan kredensi jangka panjang untuk mengakses sumber daya lain.

Untuk informasi selengkapnya, lihat topik berikut di panduan pengguna IAM:

- Peran IAM
- Skenario umum untuk peran: pengguna, aplikasi, dan layanan

Gunakan CloudTrail untuk memantau panggilan API

Kinesis Video Streams dengan WebRTC terintegrasi dengan, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau layanan AWS di Kinesis Video Streams dengan WebRTC AWS CloudTrail.

Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Kinesis Video Streams dengan WebRTC, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Lihat informasi yang lebih lengkap di the section called “Log panggilan API dengan CloudTrail”.

Enkripsi WebRTC

Enkripsi ujung ke ujung adalah fitur wajib Amazon Kinesis Video Streams dengan WebRTC, dan Kinesis Video Streams menerapkannya pada semua komponen, termasuk pensinyalan dan media atau streaming data. Terlepas dari apakah komunikasi tersebut peer-to-peer atau disampaikan melalui Kinesis Video Streams TURN end point, semua komunikasi WebRTC dienkripsi dengan aman melalui protokol enkripsi standar.

Pesan pensinyalan dipertukarkan menggunakan Websockets aman (WSS), aliran data dienkripsi menggunakan Datagram Transport Layer Security (DTLS), dan aliran media dienkripsi menggunakan Secure Real-time Transport Protocol (SRTP).

Pantau metrik dan panggilan API

Pemantauan merupakan bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon Kinesis Video Streams dengan AWS WebRTC dan solusi Anda. Anda harus mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multi-titik jika terjadi. Namun, sebelum Anda mulai memantau Kinesis Video Streams dengan WebRTC, Anda harus membuat rencana pemantauan yang mencakup jawaban atas pertanyaan-pertanyaan berikut:

- Apa saja sasaran pemantauan Anda?
- Sumber daya apa yang akan Anda pantau?
- Seberapa sering Anda akan memantau sumber daya ini?
- Alat pemantauan apa yang akan Anda gunakan?
- Siapa yang akan melakukan tugas pemantauan?
- Siapa yang harus diberi tahu saat terjadi kesalahan?

Setelah Anda menentukan sasaran pemantauan dan membuat rencana pemantauan, langkah selanjutnya adalah menetapkan garis dasar untuk Kinesis Video Streams normal dengan kinerja WebRTC di lingkungan Anda. Anda harus mengukur Kinesis Video Streams dengan kinerja WebRTC pada berbagai waktu dan dalam kondisi beban yang berbeda. Saat Anda memantau Kinesis Video Streams dengan WebRTC, Anda harus menyimpan riwayat data pemantauan yang telah Anda kumpulkan. Anda dapat membandingkan Kinesis Video Streams saat ini dengan kinerja WebRTC dengan data historis ini untuk membantu Anda mengidentifikasi pola kinerja normal dan anomali kinerja, dan merancang metode untuk mengatasi masalah yang mungkin timbul.

Topik

Pantau Kinesis Video Streams dengan WebRTC

Anda dapat memantau Amazon Kinesis Video Streams dengan CloudWatch WebRTC menggunakan Amazon, yang mengumpulkan dan memproses data mentah dari Amazon Kinesis Video Streams dengan WebRTC menjadi metrik hampir real-time yang dapat dibaca. Statistik ini dicatat untuk jangka

waktu 15 bulan, sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang performa aplikasi atau layanan web Anda.

Amazon Kinesis Video Streams menyediakan metrik berikut:

Topik

Metrik pensinyalan

Bagian ini memberikan informasi tentang cara memantau dan memecahkan masalah terkait pensinyalan menggunakan Log. CloudWatch

Nama metrik	Deskripsi	Unit	Dimensi	
Kegagalan	'0' dipancarkan jika Operasi yang disebutkan dalam dimensi mengembalikan 200 respons kode status. '1' sebaliknya.	Hitungan	Operasi, SignalingChannelName	
Latensi	Mengukur waktu yang dibutuhkan layanan untuk menerima permintaan dan mengembalikan respons.	Milidetik	Operasi, SignalingChannelName	
MessagesTransferred.Hitung	Jumlah total pesan yang dikirim dan diterima untuk saluran.	Hitungan	SignalingChannelName	

`Operation`Dimensi berlaku untuk yang berikut APIs:

- ConnectAsMaster
- ConnectAsViewer

- SendSdpOffer
- SendSdpAnswer
- SendCandidate
- SendAlexaOfferToMaster
- GetIceServerConfig
- Putuskan sambungan

Turn metrik

Bagian ini memberikan informasi tentang cara memantau dan memecahkan masalah terkait Turn menggunakan Log. CloudWatch

Nama metrik	Deskripsi	Unit	Dimensi
TURNConnectedMenit	'1' dipancarkan untuk setiap alokasi TURN yang digunakan untuk mengalirkan data dalam satu menit.	Hitungan	SignalingChannelName

Metrik konsumsi WebRTC

Bagian ini memberikan informasi tentang cara memantau dan memecahkan masalah terkait konsumsi WebRTC menggunakan Log. CloudWatch

Nama metrik	Deskripsi	Unit	Dimensi
Kegagalan	'0' dipancarkan jika Operasi yang disebutkan dalam dimensi mengembalikan 200 respons kode status. '1' sebaliknya.	Hitungan	Operasi, SignalingChannelName
Latensi	Mengukur waktu yang dibutuhkan layanan untuk menerima permintaan dan mengembalikan respons.	Milidetik	Operasi, SignalingChannelName

Nama metrik	Deskripsi	Unit	Dimensi
TotalBitrate	Total bitrate yang dikirim. Jika Peran ditentukan, ini mewakili total bitrate yang dikirim oleh peserta MASTER atau total bitrate agregat yang dikirim oleh peserta VIEWER.	bps	Operasi, SignalingChannelName, Peran
TotalPacketCount	Total jumlah paket yang dikirim. Jika Peran ditentukan, ini mewakili total bitrate yang dikirim oleh peserta MASTER atau total bitrate agregat yang dikirim oleh peserta VIEWER.	Hitungan	Operasi, SignalingChannelName, Peran
RTCRecordingMenit Web	Jumlah Web RTCRecording Minute terjadi untuk saluran.	Hitungan	Operasi, SignalingChannelName, Peran
RTCViewerMenit Web	Jumlah Web RTCViewer Minute terjadi untuk saluran.	Hitungan	Operasi, SignalingChannelName, Peran

`Operation`Dimensi berlaku untuk yang berikut APIs:

- JoinStorageSession
- JoinStorageSessionAsViewer

`Role`dimensi:

- TUAN
- PENONTON

Log panggilan API dengan AWS CloudTrail

Amazon Kinesis Video Streams dengan WebRTC terintegrasi dengan, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau layanan AWS di Amazon Kinesis Video

Streams dengan WebRTC AWS CloudTrail. CloudTrail menangkap semua panggilan API untuk Amazon Kinesis Video Streams dengan WebRTC sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari konsol Amazon Kinesis Video Streams dan panggilan kode ke Amazon Kinesis Video Streams dengan operasi WebRTC API. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon S3, termasuk acara untuk Amazon Kinesis Video Streams dengan WebRTC. Jika Anda tidak membuat konfigurasi jejak, Anda masih dapat melihat kejadian terbaru dalam konsol CloudTrail di Riwayat peristiwa. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke Amazon Kinesis Video Streams dengan WebRTC, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, termasuk cara mengonfigurasi dan mengaktifkannya, lihat Panduan AWS CloudTrail Pengguna.

Amazon Kinesis Video Streams dengan WebRTC dan CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas peristiwa yang didukung terjadi di Amazon Kinesis Video Streams dengan WebRTC, aktivitas tersebut CloudTrail direkam dalam suatu peristiwa bersama dengan peristiwa layanan lainnya dalam riwayat Acara. AWS Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk Amazon Kinesis Video Streams dengan WebRTC, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua Wilayah AWS. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengkonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Amazon Kinesis Video Streams dengan WebRTC mendukung pencatatan tindakan berikut sebagai peristiwa dalam file log: CloudTrail

- CreateSignalingChannel
- DeleteSignalingChannel
- DescribeSignalingChannel
- GetSignalingChannelEndpoint
- ListSignalingChannels
- ListTagsForResource
- TagResource
- UntagResource
- UpdateSignalingChannel

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan itu dibuat dengan kredenal pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi lain, lihat Elemen userIdentity CloudTrail .

Contoh: Entri file log

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber mana pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan CreateSignalingChanneltindakan.

```
{
   "eventVersion":"1.05",
   "userIdentity":{
      "type":"IAMUser",
      "principalId":"EX_PRINCIPAL_ID",
      "arn":"arn:aws:iam::123456789012:user/Alice",
      "accountId":"123456789012",
      "accessKeyId":"EXAMPLE_KEY_ID",
      "userName":"Alice"
   },
   "eventTime":"2019-11-19T22:49:04Z",
   "eventSource":"kinesisvideo.amazonaws.com",
   "eventName":"CreateSignalingChannel",
   "awsRegion":"us-west-2",
   "sourceIPAddress":"127.0.0.1",
   "userAgent":"aws-sdk-java/unknown-version Linux/x.xx",
   "requestParameters":{
      "channelName":"YourChannelName"
   },
   "responseElements":{
      "channelARN":"arn:aws:kinesisvideo:us-west-2:123456789012:channel/
YourChannelName/1574203743620"
   },
   "requestID":"df3c99c4-1d97-49da-8569-7de6c92b4856",
   "eventID":"bb74bac2-964c-49b0-903a-3501c6bde632"
}
```

Referensi API

Topik

WebSocket titik akhir APIs

Berikut ini adalah Amazon Kinesis Video Streams WebSocket dengan APIs endpoint WebRTC:

Topik

ConnectAsMaster

Terhubung sebagai master ke saluran pensinyalan yang ditentukan oleh titik akhir. Pustaka WebSocket -complaint apa pun dapat digunakan untuk terhubung ke titik akhir websocket (WSS) aman yang diperoleh dari panggilan API. `GetSignalingChannelEndpoint` Nama Sumber Daya Amazon (ARN) dari saluran pensinyalan harus disediakan sebagai parameter string kueri. Ada titik akhir terpisah untuk menghubungkan sebagai master dan sebagai penampil. Jika lebih dari satu klien terhubung sebagai master ke saluran tertentu, maka permintaan terbaru diutamakan. Metadata koneksi yang ada ditimpa oleh yang baru.

Permintaan

```
"X-Amz-ChannelARN": "string"
```

- X-AMZ-channelarn - ARN dari saluran pensinyalan.
  - Jenis: string
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 1024.
  - Pola: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`
  - Wajib: Ya

Respons

200 OK kode status HTTP dengan badan kosong.

Kesalahan

- InvalidArgumentException

  Parameter yang ditentukan melebihi batasannya, tidak didukung, atau tidak dapat digunakan. Untuk informasi selengkapnya, lihat pesan yang dikembalikan.

  Kode Status HTTP: 400
- AccessDeniedException

  Penelepon tidak berwenang untuk mengakses saluran yang diberikan atau token telah kedaluwarsa.

  Kode Status HTTP: 403
- ResourceNotFoundException

  Saluran itu tidak ada.

  Kode Status HTTP: 404
- ClientLimitExceededException

  Ketika API dipanggil pada tingkat yang terlalu tinggi. Untuk informasi selengkapnya, lihat Amazon Kinesis Video Streams dengan kuota layanan WebRTC dan Error Retries dan Exponential Backoff di. AWS

  Kode Status HTTP: 400

Batas/Pelambatan

API ini dibatasi pada tingkat akun jika API dipanggil pada tingkat yang terlalu tinggi. Kesalahan kembali saat dibatasi dengan. `ClientLimitExceededException`

Idempoten

Jika koneksi sudah ada untuk clientID dan saluran yang ditentukan, metadata koneksi diperbarui dengan informasi baru.

Coba lagi perilaku

Ini dihitung sebagai panggilan API baru.

Panggilan bersamaan

Panggilan bersamaan diizinkan, metadata koneksi diperbarui untuk setiap panggilan.

ConnectAsViewer

Terhubung sebagai penampil ke saluran pensinyalan yang ditentukan oleh titik akhir. Pustaka WebSocket yang sesuai dapat digunakan untuk terhubung ke titik akhir websocket (WSS) aman yang diperoleh dari panggilan API. `GetSignalingChannelEndpoint` Nama Sumber Daya Amazon (ARN) dari saluran pensinyalan dan ID klien harus disediakan sebagai parameter string kueri. Ada titik akhir terpisah untuk menghubungkan sebagai master dan sebagai penampil. Jika ada koneksi yang ada dengan sama `ClientId` seperti yang ditentukan dalam permintaan, koneksi baru diutamakan. Metadata koneksi ditimpa dengan informasi baru.

Permintaan

```
"X-Amz-ChannelARN": "string",
"X-Amz-ClientId": "string"
```

- X-AMZ-channelarn - ARN dari saluran pensinyalan.
  - Jenis: string
  - Panjang batasan: Panjang minimum 1. Panjang maksimal 1024
  - Pola: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`
  - Wajib: Ya
- X-Amz- ClientId - Pengidentifikasi unik untuk klien.
  - Jenis: string
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `^(?!(?i)AWS_.*)[a-zA-Z0-9_.-]`

> **Note**
>
> `X-Amz-ClientId`Tidak bisa mulai dengan`AWS_`.

- Wajib: Ya

Respons

200 OK kode status HTTP dengan badan kosong.

Kesalahan

- InvalidArgumentException

  Parameter yang ditentukan melebihi batasannya, tidak didukung, atau tidak dapat digunakan. Untuk informasi selengkapnya, lihat pesan yang dikembalikan.

  Kode Status HTTP: 400
- AccessDeniedException

  Penelepon tidak berwenang untuk mengakses saluran yang diberikan atau token telah kedaluwarsa.

  Kode Status HTTP: 403
- ResourceNotFoundException

  Saluran itu tidak ada.

  Kode Status HTTP: 404
- ClientLimitExceededException

  Ketika API dipanggil pada tingkat yang terlalu tinggi atau ketika ada lebih dari jumlah maksimum pemirsa yang didukung yang terhubung ke saluran. Untuk informasi selengkapnya, lihat Amazon Kinesis Video Streams dengan kuota layanan WebRTC dan Error Retries dan Exponential Backoff di. AWS

  Kode Status HTTP: 400

Batas/Pelambatan

API ini dibatasi pada tingkat akun jika API dipanggil pada tingkat yang terlalu tinggi atau ketika jumlah maksimum penonton yang didukung terhubung ke saluran lebih dari jumlah maksimum yang didukung. Kesalahan kembali saat dibatasi dengan. `ClientLimitExceededException`

Idempoten

Jika koneksi sudah ada untuk yang ditentukan `ClientId` dan saluran, metadata koneksi diperbarui dengan informasi baru.

Coba lagi perilaku

Ini dihitung sebagai panggilan API baru.

Panggilan bersamaan

Panggilan bersamaan diizinkan, metadata koneksi diperbarui untuk setiap panggilan.

Penerimaan pesan asinkron

Semua pesan respons dikirim secara asinkron ke penerima sebagai peristiwa (misalnya, penawaran SDP atau pengiriman jawaban SDP). Berikut ini adalah struktur pesan acara.

Peristiwa

```
{
    "senderClientId": "string",
    "messageType": "string",
    "messagePayload": "string",
    "statusResponse": {
        "correlationId": "string",
        "errorType": "string",
        "statusCode": "string",
        "description": "string"
    }
}
```

- senderClientId- Pengidentifikasi unik untuk klien pengirim.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak
- MessageType - Jenis acara.
  - Jenis: ENUM

  - Jenis yang Valid: `SDP_OFFERSDP_ANSWER,ICE_CANDIDATE,GO_AWAY,,RECONNECT_ICE_SERVER, STATUS_RESPONSE`
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya
- MessagePayLoad - Konten pesan yang dikodekan base64.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 10K.
  - Wajib: Tidak
- CorrelationID - Pengidentifikasi unik dari pesan yang statusnya dimaksudkan. Ini adalah CorrelationID yang sama yang disediakan dalam pesan klien (misalnya, penawaran SDP, jawaban SDP, atau kandidat ICE).
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya
- ErrorType - Nama untuk mengidentifikasi kesalahan secara unik.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak
- StatusCode - kode status HTTP sesuai dengan sifat respon.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak
- description - Deskripsi string yang menjelaskan status.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 1K.
  - Wajib: Tidak

SendSdpOffer

Mengirim penawaran ke penerima target. Prasyaratnya adalah klien harus sudah terhubung ke WebSocket titik akhir yang diperoleh dari API. `GetSignalingChannelEndpoint`

Jika jenis pengirim adalah penampil, maka ia mengirimkan penawaran ke master. Juga, tidak perlu menentukan `RecipientClientId` dan nilai tertentu untuk `RecipientClientId` diabaikan.
Jika jenis pengirim adalah master, penawaran dikirim ke penampil target yang ditentukan oleh. `RecipientClientId` `RecipientClientId`adalah masukan yang diperlukan dalam kasus ini.

Aplikasi klien master diizinkan mengirim penawaran ke penampil mana pun, sedangkan aplikasi klien penampil hanya diizinkan mengirim penawaran ke aplikasi klien utama. Jika aplikasi klien penampil mencoba mengirim penawaran ke aplikasi klien penampil lain, permintaan tersebut TIDAK akan dihormati. Jika ada penawaran luar biasa untuk klien yang sama yang belum dikirimkan, itu ditimpa dengan penawaran baru.

Permintaan

```
{
    "action": "SDP_OFFER",
    "recipientClientId": "string",
    "messagePayload": "string",
    "correlationId": "string"
}
```

- action - Jenis pesan yang sedang dikirim.
  - Jenis: ENUM
  - Nilai valid: `SDP_OFFER`, `SDP_ANSWER`, `ICE_CANDIDATE`
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya
- recipientClientId- Pengidentifikasi unik untuk penerima.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya

- MessagePayLoad - Konten pesan yang disandikan basis-64.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 10K.
  - Wajib: Ya
- CorrelationID - Sebuah identifier unik untuk pesan. Ini adalah parameter opsional.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak

Respons

Jika pesan berhasil diterima oleh backend pensinyalan, tidak ada respons yang dikembalikan. Jika layanan mengalami kesalahan dan jika `correlationId` ditentukan dalam permintaan, rincian kesalahan dikembalikan sebagai `STATUS_RESPONSE` pesan. Untuk informasi selengkapnya, lihat the section called “Penerimaan pesan asinkron”.

Kesalahan

- InvalidArgumentException

  Parameter yang ditentukan melebihi batasannya, tidak didukung, atau tidak dapat digunakan. Untuk informasi selengkapnya, lihat pesan yang dikembalikan.

  Kode Status HTTP: 400
- ClientLimitExceededException

  Ketika API dipanggil pada tingkat yang terlalu tinggi. Untuk informasi selengkapnya, lihat Amazon Kinesis Video Streams dengan kuota layanan WebRTC dan Error Retries dan Exponential Backoff di. AWS

  Kode Status HTTP: 400

Batas/Pelambatan

API ini dibatasi pada tingkat akun jika API dipanggil pada tingkat yang terlalu tinggi. Kesalahan kembali saat dibatasi dengan. `ClientLimitExceededException`

Idempoten

API ini tidak idempoten.

Coba lagi perilaku

Ini dihitung sebagai panggilan API baru.

Panggilan bersamaan

Panggilan bersamaan diizinkan. Penawaran dikirim satu kali per setiap panggilan.

SendSdpAnswer

Mengirimkan jawaban ke penerima target. Prasyaratnya adalah klien harus sudah terhubung ke WebSocket titik akhir yang diperoleh dari API. `GetSignalingChannelEndpoint`

Jika jenis pengirim adalah penampil, maka ia mengirimkan jawaban ke master. Juga, tidak perlu menentukan `RecipientClientId` dan nilai tertentu untuk `RecipientClientId` diabaikan. Jika jenis pengirim adalah master, jawabannya dikirim ke penampil target yang ditentukan oleh. `RecipientClientId` `RecipientClientId`adalah masukan yang diperlukan dalam kasus ini.

Aplikasi klien master diizinkan mengirim jawaban ke penampil mana pun, sedangkan aplikasi klien penampil hanya diizinkan mengirim jawaban ke aplikasi klien utama. Jika aplikasi klien penampil mencoba mengirim jawaban ke aplikasi klien penampil lain, permintaan tersebut TIDAK akan dihormati. Jika ada jawaban yang luar biasa untuk klien yang sama yang belum disampaikan, itu ditimpa dengan jawaban baru.

Permintaan

```
{
    "action": "SDP_ANSWER",
    "recipientClientId": "string",
    "messagePayload": "string",
    "correlationId": "string"
}
```

- action - Jenis pesan yang sedang dikirim.
  - Jenis: ENUM

  - Nilai valid: `SDP_OFFER`, `SDP_ANSWER`, `ICE_CANDIDATE`
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya
- recipientClientId- Pengidentifikasi unik untuk penerima.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya
- MessagePayLoad - Konten pesan yang disandikan basis-64.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 10K.
  - Wajib: Ya
- CorrelationID - Sebuah identifier unik untuk pesan.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak

Respons

Tidak ada respons yang dikembalikan jika pesan berhasil diterima oleh backend pensinyalan. Jika layanan mengalami kesalahan dan jika `correlationId` ditentukan dalam permintaan, rincian kesalahan dikembalikan sebagai `STATUS_RESPONSE` pesan. Untuk informasi selengkapnya, lihat Penerimaan pesan asinkron.

Kesalahan

- InvalidArgumentException

  Parameter yang ditentukan melebihi batasannya, tidak didukung, atau tidak dapat digunakan. Untuk informasi selengkapnya, lihat pesan yang dikembalikan.

  Kode Status HTTP: 400

- ClientLimitExceededException

  Dikembalikan ketika API dipanggil pada tingkat yang terlalu tinggi. Untuk informasi selengkapnya, lihat Amazon Kinesis Video Streams dengan kuota layanan WebRTC dan Error Retries dan Exponential Backoff di. AWS

  Kode Status HTTP: 400

Batas/Pelambatan

API ini dibatasi pada tingkat akun jika API dipanggil pada tingkat yang terlalu tinggi. Kesalahan dikembalikan saat dibatasi dengan. `ClientLimitExceededException`

Idempoten

API ini tidak idempoten.

Coba lagi perilaku

Ini dihitung sebagai panggilan API baru.

Panggilan bersamaan

Panggilan bersamaan diizinkan. Penawaran dikirim satu kali per setiap panggilan.

SendIceCandidate

Mengirim kandidat ICE ke penerima target. Prasyaratnya adalah klien harus sudah terhubung ke WebSocket titik akhir yang diperoleh dari API. `GetSignalingChannelEndpoint`

Jika jenis pengirim adalah penampil, maka ia mengirim kandidat ICE ke master. Juga, tidak perlu menentukan `RecipientClientId` dan nilai tertentu untuk `RecipientClientId` diabaikan. Jika jenis pengirim adalah master, kandidat ICE dikirim ke target yang ditentukan oleh. `RecipientClientId` `RecipientClientId`adalah masukan yang diperlukan dalam kasus ini.

Aplikasi klien master diizinkan mengirim kandidat ICE ke penampil mana pun, sedangkan aplikasi klien penampil hanya diizinkan mengirim kandidat ICE ke aplikasi klien utama. Jika aplikasi klien penampil mencoba mengirim kandidat ICE ke aplikasi klien penampil lain, permintaan tersebut TIDAK akan dihormati.

Permintaan

```
{
    "action": "ICE_CANDIDATE",
    "recipientClientId": "string",
    "messagePayload": "string",
    "correlationId": "string"
}
```

- action - Jenis pesan yang sedang dikirim.
  - Jenis: ENUM
  - Nilai valid: `SDP_OFFER`, `SDP_ANSWER`, `ICE_CANDIDATE`
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Ya
- recipientClientId- Pengidentifikasi unik untuk penerima.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak
- MessagePayLoad - Konten pesan yang dikodekan base64.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 10K.
  - Wajib: Ya
- CorrelationID - Sebuah identifier unik untuk pesan.
  - Tipe: String
  - Panjang batasan: Panjang minimum 1. Panjang maksimum 256.
  - Pola: `[a-zA-Z0-9_.-]+`
  - Wajib: Tidak

Respons

Tidak ada respons yang dikembalikan jika pesan berhasil diterima oleh backend pensinyalan. Jika layanan mengalami kesalahan dan jika `correlationId` ditentukan dalam permintaan, rincian kesalahan dikembalikan sebagai `STATUS_RESPONSE` pesan. Untuk informasi selengkapnya, lihat Penerimaan pesan asinkron.

Kesalahan

- InvalidArgumentException

  Parameter yang ditentukan melebihi batasannya, tidak didukung, atau tidak dapat digunakan. Untuk informasi selengkapnya, lihat pesan yang dikembalikan.

  Kode Status HTTP: 400

- ClientLimitExceededException

  Ketika API dipanggil pada tingkat yang terlalu tinggi. Untuk informasi selengkapnya, lihat Amazon Kinesis Video Streams dengan kuota layanan WebRTC dan Error Retries dan Exponential Backoff di. AWS

  Kode Status HTTP: 400

Batas/Pelambatan

API ini dibatasi pada tingkat akun jika API dipanggil pada tingkat yang terlalu tinggi. Kesalahan kembali saat dibatasi dengan. `ClientLimitExceededException`

Idempoten

API ini tidak idempoten.

Coba lagi perilaku

Ini dihitung sebagai panggilan API baru.

Panggilan bersamaan

Panggilan bersamaan diizinkan. Penawaran dikirim satu kali per setiap panggilan.

Putuskan sambungan

Klien dapat menutup koneksi kapan saja. WebSocket-pustaka yang sesuai mendukung fungsionalitas dekat. Ketika koneksi ditutup, layanan menandai klien sebagai offline untuk saluran pensinyalan tertentu dan tidak mencoba mengirimkan pesan apa pun. Perilaku yang sama juga berlaku jika terjadi batas waktu koneksi idle.

Layanan ini juga mengirimkan indikasi pemutusan ke klien, misalnya, selama penerapan atau pemeliharaan server. Berikut ini adalah pesan indikasi yang ditentukan:

- GO_AWAY: Pesan ini digunakan untuk memulai shutdown koneksi. Ini memungkinkan klien untuk memproses pesan sebelumnya dengan anggun, memutuskan sambungan, dan menyambung kembali ke saluran pensinyalan jika diperlukan.
- RECONNECT_ICE_SERVER: Pesan ini digunakan untuk memulai shutdown koneksi relay dan memungkinkan klien untuk memutuskan sambungan dengan anggun, mendapatkan konfigurasi server ICE baru, dan menyambung kembali ke server relai jika diperlukan.

Memecahkan Masalah Amazon Kinesis Video Streams dengan WebRTC

Gunakan informasi berikut untuk memecahkan masalah umum yang mungkin Anda temui dengan Amazon Kinesis Video Streams dengan WebRTC.

Masalah membangun peer-to-peer sesi

WebRTC dapat membantu meringankan masalah yang terjadi karena:

- Terjemahan alamat jaringan (NAT)
- firewall
- Proksi antar rekan

WebRTC menyediakan kerangka kerja untuk membantu bernegosiasi dan memelihara koneksi selama rekan-rekan terhubung. Ini juga menyediakan mekanisme untuk menyampaikan media melalui TURN server jika peer-to-peer koneksi tidak dapat dinegosiasikan.

Mengingat semua komponen yang diperlukan untuk membangun koneksi, ada baiknya memahami beberapa alat yang tersedia untuk membantu memecahkan masalah yang terkait dengan membuat sesi.

Topik

Penawaran dan jawaban Session Description Protocol (SDP)

Session Description Protocol (SDP) menawarkan dan menjawab menginisialisasi sesi RTC antara rekan-rekan.

Untuk mempelajari lebih lanjut tentang protokol SDP, lihat spesifikasinya.

- Penawaran dihasilkan oleh “pemirsa” yang ingin terhubung ke rekan yang terhubung ke saluran pensinyalan sebagai “master” di Kinesis Video Streams dengan WebRTC.
- Jawaban dihasilkan oleh penerima penawaran.

Baik penawaran dan jawaban dihasilkan di sisi klien, meskipun mereka mungkin berisi kandidat ICE yang telah dikumpulkan hingga saat itu.

Kinesis Video Streams SDK WebRTC untuk C menyertakan variabel lingkungan sederhana yang dapat Anda atur untuk mencatat SDP. Ini berguna untuk memahami penawaran yang diterima dan jawaban yang dihasilkan.

SDPs Untuk masuk `stdout` dari SDK, setel variabel lingkungan berikut:`export DEBUG_LOG_SDP=TRUE`. Anda juga dapat mencatat penawaran dan jawaban SDP di klien JavaScript berbasis menggunakan `sdpOffer` acara tersebut. Untuk melihat ini ditunjukkan, lihat GitHub.

Untuk informasi tambahan, lihat the section called “Pantau Kinesis Video Streams dengan WebRTC”.

Jika jawaban SDP tidak dikembalikan, ada kemungkinan rekan tidak dapat menerima penawaran SDP, karena penawaran tersebut tidak mengandung codec media yang kompatibel. Anda mungkin melihat log yang mirip dengan berikut ini:

```
I/webrtc_video_engine.cc: (line 808): SetSendParameters: {codecs:
 [VideoCodec[126:H264]], conference_mode: no, extensions: [], extmap-allow-mixed:
 false, max_bandwidth_bps: -1, mid: video1}
E/webrtc_video_engine.cc: (line 745): No video codecs supported.
E/peer_connection.cc: (line 6009): Failed to set remote video description send
 parameters for m-section with mid='video1'. (INVALID_PARAMETER)
E/peer_connection.cc: (line 3097): Failed to set remote offer sdp: Failed to set remote
 video description send parameters for m-section with mid='video1'.
E/KinesisVideoSdpObserver: onSetFailure(): Error=Failed to set remote offer sdp: Failed
 to set remote video description send parameters for m-section with mid='video1'.
D/KVSWebRtcActivity: Received SDP offer for client ID: null. Creating answer
E/peer_connection.cc: (line 2373): CreateAnswer: Session error code: ERROR_CONTENT.
 Session error description: Failed to set remote video description send parameters for
 m-section with mid='video1'..
E/KinesisVideoSdpObserver: onCreateFailure(): Error=Session error code: ERROR_CONTENT.
 Session error description: Failed to set remote video description send parameters for
 m-section with mid='video1'..
```

Saat Anda meninjau konten penawaran SDP, cari baris yang dimulai `a=rtpmap` untuk melihat codec media mana yang diminta.

```
...
a=rtpmap:126 H264/90000
...
a=rtpmap:111 opus/48000/2
...
```

Jika Anda menggunakan Safari sebagai penampil untuk terhubung ke master yang mengirim media H.265 dan Anda menemukan yang berikut:

- `InvalidAccessError: Failed to set remote answer sdp: Called with SDP without DTLS fingerprint.`
- `InvalidAccessError: Failed to set remote answer sdp: rtcp-mux must be enabled when BUNDLE is enabled.`

Konfirmasikan masalahnya adalah dengan penawaran SDP yang dihasilkan oleh browser. Dalam penawaran SDP, cari baris mulai`a=rtpmap`, dan periksa apakah ada garis untuk H.265. Seharusnya terlihat seperti ini:

```
a=rtpmap:104 H265/90000
```

Jika tidak ada, aktifkan codec H.265 untuk WebRTC di pengaturan Safari.

Di navigasi atas Safari, lakukan hal berikut:

- Pilih Safari > Pengaturan... > Lanjutan. Pilih kotak Tampilkan fitur untuk pengembang web.
- Pilih Bendera Fitur. Pilih kotak codec WebRTC H265.

Mulai ulang browser Anda agar perubahan diterapkan.

Evaluasi generasi kandidat ICE

Kandidat ICE dihasilkan oleh setiap klien yang melakukan panggilan ke `STUN` server. Untuk Kinesis Video Streams dengan WebRTC`STUN`, server adalah. `stun:stun.kinesisvideo.{aws-region}.amazonaws.com:443`

Selain memanggil `STUN` server untuk mendapatkan kandidat, klien sering juga memanggil `TURN` server. Mereka melakukan panggilan ini sehingga server relay dapat digunakan sebagai fallback jika peer-to-peer koneksi langsung tidak dapat dibuat.

Anda dapat menggunakan alat-alat berikut untuk menghasilkan kandidat ICE:

- Sampel WebRTC Trickle ICE, yang menggunakan Trickle ICE untuk mengumpulkan kandidat
- IceTest.Info

Dengan kedua alat ini. Anda dapat memasukkan informasi STUN dan TURN server untuk mengumpulkan kandidat.

Untuk mendapatkan informasi TURN server dan kredensi yang diperlukan untuk Kinesis Video Streams dengan WebRTC, Anda dapat memanggil operasi API. GetIceServerConfig

AWS CLI Panggilan berikut menunjukkan cara mendapatkan informasi ini untuk digunakan dalam dua alat ini.

```
export CHANNEL_ARN="YOUR_CHANNEL_ARN"

aws kinesisvideo get-signaling-channel-endpoint \
    --channel-arn $CHANNEL_ARN \
    --single-master-channel-endpoint-configuration Protocols=WSS,HTTPS,Role=MASTER
```

Output dari `get-signaling-channel-endpoint`perintah mengembalikan respons yang terlihat seperti ini:

```
{
  "ResourceEndpointList": [
    {
      "Protocol": "HTTPS",
      "ResourceEndpoint": "https://your-endpoint.kinesisvideo.us-east-1.amazonaws.com"
    },
    {
      "Protocol": "WSS",
      "ResourceEndpoint": "wss://your-endpoint.kinesisvideo.us-east-1.amazonaws.com"
    }
  ]
}
```

Gunakan `ResourceEndpoint` nilai HTTPS untuk mendapatkan daftar TURN server sebagai berikut:

```
export ENDPOINT_URL="https://your-endpoint.kinesisvideo.us-east-1.amazonaws.com"
```

```
aws kinesis-video-signaling get-ice-server-config \
    --channel-arn $CHANNEL_ARN \
    --service TURN \
    --client-id my-amazing-client \
    --endpoint-url $ENDPOINT_URL
```

Respons berisi rincian TURN server, termasuk titik akhir untuk TCP dan UDP dan kredensional yang diperlukan untuk mengaksesnya.

> **Note**
>
> Nilai TTL dalam respons menentukan durasi, dalam hitungan detik, yang valid untuk kredensil ini. Gunakan nilai ini dalam sampel WebRTC Trickle ICE atau IceTestdi.Info untuk menghasilkan kandidat ICE menggunakan titik akhir layanan yang dikelola Kinesis Video Streams.

Tentukan kandidat mana yang digunakan untuk membangun koneksi

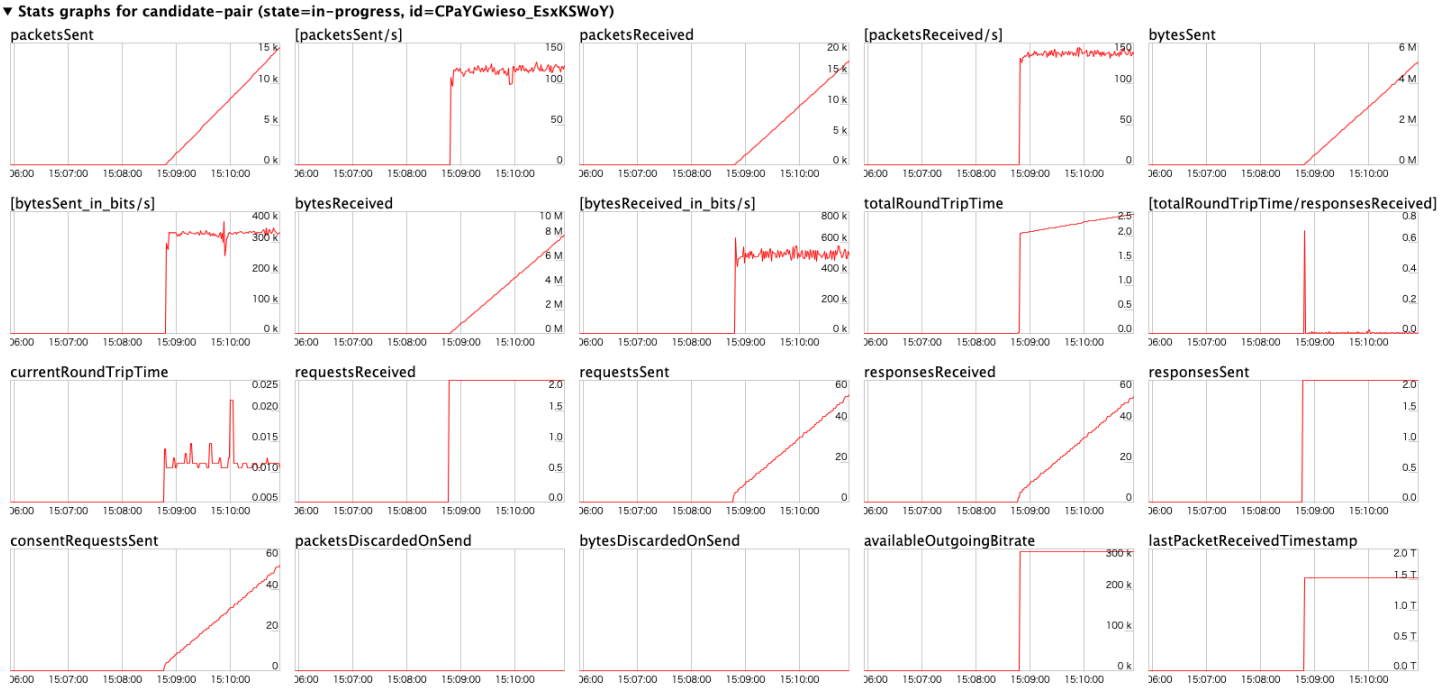
Akan sangat membantu untuk memahami kandidat mana yang digunakan untuk berhasil mendirikan sesi. Jika Anda memiliki klien berbasis browser yang menjalankan sesi yang telah ditetapkan, Anda dapat menentukan informasi ini di Google Chrome dengan menggunakan utilitas internal webrtc bawaan.

Buka sesi WebRTC di satu tab browser.

Di tab lain, buka`chrome://webrtc-internals/`. Anda dapat melihat semua informasi tentang sesi Anda yang sedang berlangsung di tab ini.

Anda akan melihat informasi tentang koneksi yang dibuat. Contoh:

Anda juga dapat mengonfirmasi metrik seperti berikut untuk koneksi yang dibuat.



Batas waktu terkait ICE

Nilai batas waktu default ditetapkan untuk ICE di file. KvsRtcConfiguration Default harus cukup untuk sebagian besar pengguna, tetapi Anda mungkin perlu menyesuaikannya untuk meningkatkan peluang membangun koneksi melalui jaringan yang buruk. Anda dapat mengonfigurasi default ini dalam aplikasi.

Tinjau log untuk pengaturan default:

```
2024-01-08 19:43:44.433 INFO    iceAgentValidateKvsRtcConfig():
 iceLocalCandidateGatheringTimeout: 10000 ms
 iceConnectionCheckTimeout: 12000 ms
 iceCandidateNominationTimeout: 12000 ms
 iceConnectionCheckPollingInterval: 50 ms
```

Jika Anda memiliki kualitas jaringan yang buruk dan ingin meningkatkan peluang koneksi, coba sesuaikan nilai-nilai berikut:

- `iceLocalCandidateGatheringTimeout`- Tingkatkan batas waktu tunggu ini untuk mengumpulkan kandidat potensial tambahan untuk mencoba koneksi. Tujuannya adalah untuk mencoba semua pasangan kandidat yang mungkin, jadi jika Anda berada di jaringan yang buruk, tingkatkan batas ini untuk memberi lebih banyak waktu untuk berkumpul.

Misalnya, jika kandidat host tidak bekerja dan server refleksif (srflx) atau kandidat relay perlu dicoba, Anda mungkin perlu menambah batas waktu ini. Karena jaringan yang buruk, para kandidat dikumpulkan perlahan dan aplikasi tidak ingin menghabiskan lebih dari 20 detik pada langkah ini. Meningkatkan batas waktu memberikan lebih banyak waktu untuk mengumpulkan kandidat potensial untuk mencoba koneksi.

> **Note**
>
> Kami merekomendasikan bahwa nilai ini kurang dari`iceCandidateNominationTimeout`, karena langkah nominasi perlu memiliki waktu untuk bekerja dengan kandidat baru.

- `iceConnectionCheckTimeout`- Tingkatkan batas waktu ini di jaringan yang tidak stabil atau lambat, di mana pertukaran paket dan permintaan/respons yang mengikat membutuhkan waktu. Peningkatan batas waktu ini memungkinkan setidaknya satu pasangan kandidat untuk diadili untuk dinominasikan oleh rekan lainnya.
- `iceCandidateNominationTimeout`- Tingkatkan batas waktu ini untuk memastikan pasangan calon dengan kandidat estafet lokal dicoba.

  Misalnya, jika dibutuhkan sekitar 15 detik untuk mengumpulkan kandidat estafet lokal pertama, atur batas waktu ke nilai lebih dari 15 detik untuk memastikan pasangan kandidat dengan kandidat estafet lokal dicoba untuk berhasil. Jika nilainya disetel ke kurang dari 15 detik, SDK akan kalah dalam mencoba pasangan kandidat potensial, yang menyebabkan kegagalan pembentukan koneksi.

  > **Note**
  >
  > Kami merekomendasikan bahwa nilai ini lebih besar dari`iceLocalCandidateGatheringTimeout`, agar memiliki efek.

- `iceConnectionCheckPollingInterval`- Nilai ini default hingga 50 milidetik per spesifikasi. Mengubah nilai ini mengubah frekuensi pemeriksaan konektivitas dan, pada dasarnya, transisi mesin status ICE.

  Dalam pengaturan jaringan kinerja tinggi yang andal dengan sumber daya sistem yang baik, Anda dapat mengurangi nilai untuk membantu dalam pembentukan koneksi yang lebih cepat.

Meningkatkan nilai dapat membantu mengurangi beban jaringan, tetapi pembentukan koneksi bisa melambat.

> **Important**
>
> Kami tidak menyarankan untuk mengubah default ini.

Riwayat dokumen untuk Amazon Kinesis Video Streams dengan panduan pengembang WebRTC

Perubahan	Deskripsi	Tanggal
Organisasi konten yang direvisi	Merevisi tata letak dan organisasi dalam Amazon Kinesis Video Streams dengan Panduan Pengembang WebRTC.	September 18, 2024
Publikasi awal	Publikasi awal Amazon Kinesis Video Streams dengan WebRTC Developer Guide. Pelajari selengkapnya	4 November 2019

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