



Low-Latency Streaming User Guide

# Amazon IVS



# Amazon IVS: Low-Latency Streaming User Guide

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

---

# Table of Contents

<b>What is IVS Low-Latency Streaming?</b> .....	<b>1</b>
Latency .....	1
Global Solution, Regional Control .....	2
Streaming and Viewing are Global .....	2
Control is Regional .....	2
Your Channel's Region .....	3
<b>Getting Started with IVS</b> .....	<b>4</b>
Step 1: Create an AWS Account .....	4
Step 2: Set Up Root and Administrative Users .....	5
Secure Your AWS Account Root User .....	5
Create an Administrative User .....	5
Step 3: Set Up IAM Permissions .....	6
Use an Existing Policy for IVS Permissions .....	6
Optional: Create a Custom Policy for Amazon IVS Permissions .....	6
Create a New User and Add Permissions .....	8
Add Permissions to an Existing User .....	9
Step 4: Create a Channel with Optional Recording .....	10
Auto-Record to Amazon S3 .....	10
Console Instructions .....	11
CLI Instructions .....	19
Step 5: Set Up Streaming Software .....	23
Streaming with the Amazon IVS Broadcast SDK .....	24
Streaming with the Amazon IVS Console .....	25
Streaming with OBS Studio using RTMPS .....	26
Streaming with OBS Studio using SRT .....	27
Streaming a Recorded Video with FFmpeg using RTMPS .....	28
Streaming a Recorded Video with FFmpeg using SRT .....	29
Step 6: View Your Live Stream .....	30
Viewing with the Amazon IVS Player SDKs .....	30
Viewing with the Amazon IVS Console .....	30
Step 7: Check Your Service-Quota Limits (Optional) .....	31
Step 8: Prevent Undesired Content and Viewers (Recommended) .....	31
Console Instructions (Playback Restriction Policy) .....	31
CLI Instructions (Playback Restriction Policy) .....	32

How to Disable Recording .....	33
<b>Enabling Multiple Hosts on an IVS Stream .....</b>	<b>34</b>
Getting Started .....	34
Console Instructions .....	34
CLI Instructions .....	36
Broadcasting a Stage: Client-Side versus Server-Side Composition .....	36
Demo .....	37
1. Create a Stage .....	38
2. Distribute Participant Tokens .....	40
3. Join the Stage .....	40
4. Broadcast the Stage .....	43
<b>Monitoring .....</b>	<b>46</b>
Prerequisites .....	46
Access Stream Session Data .....	47
Console Instructions .....	47
AWS SDK Instructions .....	48
CLI Instructions .....	49
Filter Streams by Health .....	50
Console Instructions .....	51
CLI Instructions .....	51
CloudWatch Health Dimension for ConcurrentStreams .....	51
Access CloudWatch Metrics .....	51
CloudWatch Console Instructions .....	52
CLI Instructions .....	53
CloudWatch Metrics: IVS Low-Latency Streaming .....	53
<b>IVS Broadcast SDK .....</b>	<b>59</b>
Platform Requirements .....	59
Native Platforms .....	59
Desktop Browsers .....	60
Mobile Browsers .....	60
Webviews .....	61
Required Device Access .....	61
Support .....	61
Versioning .....	62
Web Guide .....	62
.....	62

---

Getting Started .....	63
Known Issues and Workarounds .....	69
Android Guide .....	71
Getting Started .....	72
Advanced Use Cases .....	75
Known Issues and Workarounds .....	81
iOS Guide .....	82
Getting Started .....	83
Advanced Use Cases .....	88
How iOS Chooses Camera Resolution and Frame Rate .....	97
Known Issues and Workarounds .....	98
Mixer Guide .....	99
Terminology .....	100
Canvas Properties .....	102
Slot Properties .....	103
Configuring a Broadcast Session for Mixing .....	106
Adding Slots .....	108
Removing Slots .....	109
Animations with Transitions .....	109
Mirroring the Broadcast .....	111
Custom Image Sources .....	113
Android .....	113
iOS .....	114
<b>IVS Player SDK .....</b>	<b>116</b>
Browser & Platform Requirements .....	117
Desktop Browsers .....	118
Mobile Browsers .....	118
Native Platforms .....	119
Reducing Latency in Third-Party Players .....	120
iOS Safari .....	120
Audio-Only Playback .....	120
Support .....	121
Versioning .....	121
Web Guide .....	122
Getting Started .....	122
Framework Integrations .....	125

---

Working With Content Security Policy .....	126
Known Issues and Workarounds .....	126
Android Guide .....	128
Getting Started .....	129
SDK Size .....	132
Known Issues and Workarounds .....	132
iOS Guide .....	132
Getting Started .....	133
SDK Size .....	139
Known Issues and Workarounds .....	140
Video.js Integration .....	140
Getting Started .....	140
Events .....	143
Errors .....	144
Plugins .....	144
Content Security Policy .....	145
Functions .....	145
currentTime .....	147
dispose .....	148
duration .....	148
getIVSEvents .....	149
getIVSPlayer .....	150
load .....	150
play .....	151
playbackRate .....	151
seekable .....	152
JW Player Integration .....	152
Getting Started .....	152
Events .....	154
Errors .....	155
Content Security Policy .....	156
Limitations .....	156
<b>Embedding Metadata within a Video Stream .....</b>	<b>157</b>
What is Timed Metadata? .....	157
Setting Up IAM Permissions .....	157
Inserting Timed Metadata .....	158

Using the AWS CLI .....	158
Using the Amazon IVS API .....	159
Using the IVS Broadcast SDK .....	159
Consuming Timed Metadata .....	159
Sample Demo: Quiz App .....	160
Viewing Timed Metadata .....	161
For More Information .....	161
<b>Setting Up Private Channels .....</b>	<b>162</b>
Workflow for Private Channels .....	163
Create or Import a Playback Key .....	164
Creating a New Key Pair .....	164
Importing an Existing Public Key .....	165
API Request .....	166
Enable Playback Authorization on Channels .....	166
Console Instructions .....	166
CLI Instructions .....	166
API Request (Create and Update) .....	167
Generate and Sign Playback Tokens .....	168
Token Schema .....	168
Instructions .....	170
List Playback Keys .....	171
Console Instructions .....	171
CLI Instructions .....	171
API Request .....	172
Delete Playback Keys .....	172
Console Instructions .....	172
CLI Instructions .....	172
API Request .....	173
Get Information about Playback Keys .....	173
Console Instructions .....	173
CLI Instructions .....	173
API Request .....	174
Revoke Viewer Sessions .....	174
CLI Instructions .....	174
API Request .....	175
<b>Auto-Record to Amazon S3 .....</b>	<b>176</b>

S3 Prefix .....	176
Recording Contents .....	177
Byte-Range Playlists .....	178
Thumbnails .....	178
Merge Fragmented Streams .....	179
Eligibility .....	179
Known Issue .....	180
JSON Metadata Files .....	180
Example: recording_started.json .....	184
Example: recording_ended.json .....	186
Example: recording_failed.json .....	187
Discovering the Renditions of a Recording .....	188
Playback of Recorded Content from Private Buckets .....	190
Amazon CloudFront Distribution .....	190
Playback from Amazon CloudFront .....	190
<b>Using Amazon EventBridge with IVS .....</b>	<b>193</b>
Creating Amazon EventBridge Rules for Amazon IVS .....	197
Examples: Stream State Change .....	197
Examples: Stream Health Change .....	199
Examples: Limit Breach .....	200
Examples: Recording State Change .....	203
<b>Logging IVS API Calls with AWS CloudTrail .....</b>	<b>206</b>
Amazon IVS Information in CloudTrail .....	206
Understanding Amazon IVS Log File Entries .....	207
<b>Security .....</b>	<b>209</b>
Data Protection .....	209
Identity and Access Management .....	211
Audience .....	211
How Amazon IVS Works with IAM .....	211
Identities .....	212
Policies .....	212
Authorization Based on Amazon IVS Tags .....	213
Roles .....	213
Privileged and Unprivileged Access .....	214
Best Practices for Policies .....	214
Identity-Based Policy Examples .....	214



Troubleshooting .....	219
Managed Policies for Amazon IVS .....	221
IVSReadOnlyAccess .....	221
IVSFullAccess .....	222
Policy Updates .....	222
Using Service-Linked Roles .....	223
Service-Linked Role Permissions for Amazon IVS .....	223
Creating a Service-Linked Role for Amazon IVS .....	224
Editing a Service-Linked Role for Amazon IVS .....	224
Deleting a Service-Linked Role for Amazon IVS .....	225
Supported Regions for Amazon IVS Service-Linked Roles .....	225
Logging and Monitoring .....	225
Incident Response .....	225
Resilience .....	226
Amazon IVS Video Data Plane .....	226
Infrastructure Security .....	227
API Calls .....	227
Streaming and Playback .....	227
<b>Service Quotas .....</b>	<b>228</b>
Service Quota Increases .....	228
API Call Rate Quotas .....	228
Other Quotas .....	230
Service Quotas Integration with CloudWatch Usage Metrics .....	234
Creating a CloudWatch Alarm for Usage Metrics .....	236
<b>Streaming Configuration .....</b>	<b>237</b>
Prerequisites .....	237
Reducing Latency .....	237
Avoid Third-Party Streaming/Forwarding Services .....	238
Encoder Settings .....	238
Stream Ingest: Codecs and Ingest Protocols .....	238
Resolution/Bitrate/FPS .....	239
Channel Types .....	240
Video Settings .....	246
Audio Settings .....	246
Use CBR, Not VBR .....	246
Use Progressive Signals .....	247

Network Requirements .....	247
Closed Captioning .....	247
Stream with FFmpeg .....	249
Stream with the Amazon IVS Broadcast SDK .....	250
Testing the Stream .....	250
<b>Troubleshooting FAQs .....</b>	<b>251</b>
Broadcasting and Encoding .....	251
What is stream starvation? .....	252
Why did the stream suddenly stop? .....	252
What happens when I switch networks while streaming? .....	253
How can I have multi-region redundancy with IVS? .....	253
How do I troubleshoot an IVS Web Broadcast SDK session? .....	255
How do I use Google Chrome's WebRTC-internals metrics to evaluate an IVS Web Broadcast SDK session? .....	255
Monitoring and Events .....	257
How do I monitor stream-starvation events? .....	257
How do I use Amazon CloudWatch to monitor IVS service quotas? .....	258
How do I diagnose stream instability using IVS Stream Health? .....	258
Stream Playback .....	266
How do I debug IVS player behaviors? .....	266
Why did playback freeze/stop for all viewers? .....	266
What is causing the IVS player to buffer? .....	266
Auto-Record to Amazon S3 .....	267
Why is some recording content missing? .....	267
Can KMS-S3 encryption be used with auto-record to S3? .....	268
Miscellaneous Topics .....	268
What does the "pending verification" error mean? .....	268
Can I estimate the cost of IVS usage? .....	269
<b>Undesired Content and Viewers .....</b>	<b>270</b>
Detecting Undesired Content .....	270
Anomaly Detection .....	270
Custom Content Moderation .....	271
Preventing Undesired Content and Viewers .....	272
Stop the Stream and Reset the Stream Key .....	272
Use Private Channels .....	272
Use Playback Restriction Policies .....	273

<b>Costs</b> .....	<b>274</b>
Live Video .....	274
Auto-Record to Amazon S3 .....	275
Storing Recorded Video .....	275
Serving Recorded Video .....	278
<b>Resources &amp; Support</b> .....	<b>279</b>
Resources .....	279
Demos .....	279
Partner Solutions .....	279
Analytics .....	280
Interactivity .....	280
Face and Background Filters .....	280
Support .....	281
<b>Glossary</b> .....	<b>282</b>
<b>Document History</b> .....	<b>300</b>
Low-Latency Streaming User Guide Changes .....	300
IVS Low-Latency Streaming API Reference Changes .....	399
Stage API Reference Changes .....	406
IVS Chat API Documentation Changes .....	407
<b>Release Notes</b> .....	<b>410</b>
June 13, 2024 .....	410
Amazon IVS Broadcast SDK: Android 1.19.0, iOS 1.19.0 (Low-Latency Streaming) .....	410
June 13, 2024 .....	411
IVS Broadcast SDK: Web 1.13.0 (Low-Latency Streaming) .....	411
June 6, 2024 .....	412
IVS Player SDK: Android 1.29.0, iOS 1.29.0 .....	412
June 6, 2024 .....	413
IVS Player SDK: Web 1.29.0 .....	413
May 20, 2024 .....	414
IVS Broadcast SDK: Web 1.12.0 (Low-Latency Streaming) .....	414
May 16, 2024 .....	414
Amazon IVS Broadcast SDK: Android 1.18.0, iOS 1.18.0 (Low-Latency Streaming) .....	414
May 9, 2024 .....	415
IVS Player SDK: Web 1.28.0 .....	415
May 9, 2024 .....	416
IVS Player SDK: Android 1.28.0, iOS 1.28.0 .....	416

May 6, 2024 .....	417
IVS Broadcast SDK: Web 1.11.0 (Low-Latency Streaming) .....	417
April 30, 2024 .....	417
IVS Broadcast SDK: Web 1.10.1 (Low-Latency Streaming) .....	417
April 30, 2024 .....	418
Amazon IVS Broadcast SDK: Android 1.15.2, iOS 1.15.2 (Low-Latency Streaming) .....	418
April 22, 2024 .....	419
Amazon IVS Broadcast SDK: Android 1.17.0, iOS 1.17.0 (Low-Latency Streaming) .....	419
April 11, 2024 .....	420
Amazon IVS Player SDK: Mobile & Web 1.27.0 .....	420
April 4, 2024 .....	422
Secure Reliable Transport (SRT) Ingest Support .....	422
March 21, 2024 .....	423
Amazon IVS Broadcast SDK: Android 1.16.0, iOS 1.16.0, Web 1.10.0 (Low-Latency Streaming) .....	423
March 14, 2024 .....	424
Amazon IVS Player SDK 1.26.0 .....	424
March 13, 2024 .....	426
Amazon IVS Broadcast SDK: Android 1.15.1, iOS 1.15.1 (Low-Latency Streaming) .....	426
February 29, 2024 .....	427
Amazon IVS Player SDK: Web 1.25.0 .....	427
February 22, 2024 .....	428
Amazon IVS Broadcast SDK: Android 1.15.0, iOS 1.15.0, Web 1.9.0 (Low-Latency Streaming) .....	428
February 15, 2024 .....	429
Amazon IVS Player SDK: Mobile 1.25.0 .....	429
February 1, 2024 .....	430
Amazon IVS Broadcast SDK: Android 1.14.1, iOS 1.14.1, Web 1.8.0 (Low-Latency Streaming) .....	430
January 31, 2024 .....	432
Tokenless Playback Restrictions .....	432
January 25, 2024 .....	432
Audio-Only Playback .....	432
January 18, 2024 .....	432
Amazon IVS Player SDK 1.24.0 .....	432
January 3, 2024 .....	434

Amazon IVS Broadcast SDK: Android 1.13.4, iOS 1.13.4, Web 1.7.0 (Low-Latency Streaming) .....	434
December 4, 2023 .....	435
Amazon IVS Broadcast SDK: Android 1.13.2 and iOS 1.13.2 (Low-Latency Streaming) .....	435
November 21, 2023 .....	436
Amazon IVS Broadcast SDK: Android 1.13.1 (Low-Latency Streaming) .....	436
November 17, 2023 .....	437
Amazon IVS Broadcast SDK: Android 1.13.0 and iOS 1.13.0 (Low-Latency Streaming) .....	437
November 14, 2023 .....	439
Amazon IVS Player SDK 1.23.0 .....	439
October 16, 2023 .....	441
Amazon IVS Broadcast SDK: Web 1.6.0 (Low-Latency Streaming) .....	441
October 12, 2023 .....	441
Amazon IVS Broadcast SDK: Android 1.12.1 (Low-Latency Streaming) .....	441
October 3, 2023 .....	442
Amazon IVS Player SDK 1.22.0 .....	442
October 2, 2023 .....	444
In-Console Streaming .....	444
September 14, 2023 .....	444
Amazon IVS Broadcast SDK: Web 1.5.2 (Low-Latency Streaming) .....	444
August 23, 2023 .....	444
Amazon IVS Broadcast SDK: Web 1.5.1, Android 1.12.0, and iOS 1.12.0 (Low-Latency Streaming) .....	444
August 23, 2023 .....	446
Amazon IVS Broadcast SDK: Android 1.7.6 (Low-Latency Streaming) .....	446
August 22, 2023 .....	446
Amazon IVS Player SDK 1.21.0 .....	446
August 7, 2023 .....	448
Amazon IVS Broadcast SDK: Web 1.5.0, Android 1.11.0, and iOS 1.11.0 .....	448
July 17, 2023 .....	449
R2S3 Rendition Filtering & Thumbnail Enhancements .....	449
July 14, 2023 .....	450
Amazon IVS Player SDK 1.20.0 .....	450
July 13, 2023 .....	452
Amazon IVS Broadcast SDK: Web 1.4.0, Android 1.10.0, and iOS 1.10.0 .....	452
June 28, 2023 .....	455

Viewer Session Revocation for Private Channels .....	455
June 27, 2023 .....	456
Amazon IVS Broadcast SDK: iOS 1.9.1 .....	456
June 27, 2023 .....	457
Amazon IVS Broadcast SDK 1.7.5 .....	457
June 16, 2023 .....	458
Amazon IVS Broadcast SDK: Web 1.3.3 .....	458
June 2, 2023 .....	458
Advanced Channel Types .....	458
June 1, 2023 .....	459
Amazon IVS Broadcast SDK: Android 1.9.0 and iOS 1.9.0 .....	459
May 23, 2023 .....	463
Amazon IVS Player SDK 1.19.0 .....	463
May 16, 2023 .....	464
Amazon IVS Broadcast SDK: iOS 1.8.1 .....	464
May 16, 2023 .....	465
Amazon IVS Broadcast SDK 1.7.4 .....	465
May 11, 2023 .....	466
Multiple Hosts Health .....	466
May 1, 2023 .....	466
Amazon IVS Web Broadcast SDK 1.3.2 .....	466
April 27, 2023 .....	467
Stage Participant Increase .....	467
April 4, 2023 .....	467
Amazon IVS Player SDK 1.18.0 .....	467
March 30, 2023 .....	469
RTMP Support .....	469
March 29, 2023 .....	469
Single-Use Tokens for Private Channels .....	469
March 28, 2023 .....	470
Amazon IVS Web Broadcast SDK 1.3.1 .....	470
March 23, 2023 .....	470
Support for Multiple Hosts on a Stream (Stage Resource) .....	470
March 23, 2023 .....	470
Amazon IVS Broadcast SDK: Android 1.8.0, iOS 1.8.0, Web 1.3.0 .....	470
March 2, 2023 .....	473

Amazon IVS Broadcast SDK: Android 1.7.3 .....	473
February 28, 2023 .....	474
Amazon IVS Player SDK 1.17.0 .....	474
February 16, 2023 .....	475
Byte-Range Tags and Manifest Files for Auto-Record to S3 .....	475
January 31, 2023 .....	475
Amazon IVS Chat Client Messaging SDK: Android 1.1.0 .....	475
January 17, 2023 .....	476
Amazon IVS Player SDK 1.16.0 .....	476
December 9, 2022 .....	478
Timestamp Added to Auto-Record to S3 Manifest Files .....	478
December 6, 2022 .....	478
Amazon IVS Broadcast SDK: Android 1.7.2 .....	478
November 17, 2022 .....	479
Chat Logging .....	479
November 9, 2022 .....	480
Amazon IVS Chat Client Messaging SDK: JavaScript 1.0.2 .....	480
November 1, 2022 .....	480
Amazon IVS Player SDK 1.14.0 .....	480
October 18, 2022 .....	482
Amazon IVS Chat Client Messaging SDK: JavaScript 1.0.1 .....	482
October 6, 2022 .....	482
Amazon IVS Broadcast SDK 1.7.1 .....	482
September 22, 2022 .....	483
Amazon IVS Broadcast SDK 1.7.0 .....	483
September 20, 2022 .....	485
Amazon IVS Player SDK 1.13.0 .....	485
September 15, 2022 .....	487
Vertical Video Improvement (Final Release) .....	487
September 12, 2022 .....	487
Amazon IVS Broadcast SDK 1.5.2: iOS .....	487
September 8, 2022 .....	488
Amazon IVS Chat Client Messaging SDK: Android 1.0.0 and iOS 1.0.0 .....	488
September 2, 2022 .....	489
Amazon IVS Web Broadcast SDK 1.2.0 .....	489
August 30, 2022 .....	489

---

Merge Fragmented Streams .....	489
August 9, 2022 .....	490
Amazon IVS Web Player SDK 1.12.0 .....	490
July 28, 2022 .....	490
Amazon IVS iOS Broadcast SDK 1.5.1 .....	490
July 21, 2022 .....	491
Amazon IVS Web Broadcast SDK .....	491
July 14, 2022 .....	492
Amazon IVS iOS Player SDK 1.8.3 .....	492
June 28, 2022 .....	492
Amazon IVS Player Web SDK 1.11.0 .....	492
June 22, 2022 .....	493
Amazon IVS Broadcast SDK 1.5.0 .....	493
June 9, 2022 .....	496
Vertical Video Improvement .....	496
May 24, 2022 .....	498
Amazon IVS Web and Android Player SDK 1.10.0 .....	498
April 28, 2022 .....	500
Stream Health Updates .....	500
April 26, 2022 .....	500
Amazon IVS Chat .....	500
April 22, 2022 .....	501
Amazon IVS iOS Player SDK 1.8.2 .....	501
April 19, 2022 .....	502
Amazon IVS Broadcast SDK 1.4.0 .....	502
March 31, 2022 .....	504
Amazon IVS iOS Player SDK 1.8.1 .....	504
March 3, 2022 .....	505
Amazon IVS Broadcast SDK 1.3.0 .....	505
March 1, 2022 .....	507
Amazon IVS Player SDK 1.8.0 .....	507
February 3, 2022 .....	509
Amazon IVS Broadcast SDK: Android 1.2.1 .....	509
January 20, 2022 .....	510
Amazon IVS Player SDK 1.7.0 .....	510
January 18, 2022 .....	513



R2S3 Thumbnail Configuration .....	513
December 9, 2021 .....	514
Amazon IVS Broadcast SDK 1.2.0 .....	514
November 23, 2021 .....	516
Amazon IVS Player SDK 1.6 .....	516
November 18, 2021 .....	519
Stream Health .....	519
October 20, 2021 .....	520
Amazon IVS Broadcast SDK 1.1.0: Android and iOS .....	520
September 29, 2021 .....	524
Amazon IVS Player SDK: Android 1.5.1 .....	524
September 28, 2021 .....	524
Amazon IVS Player SDK 1.5.0 .....	524
September 8, 2021 .....	527
Amazon IVS Player SDK 1.4.1 .....	527
August 13, 2021 .....	529
ListTagsForResource API Endpoint .....	529
August 10, 2021 .....	529
Amazon IVS Player SDK 1.4.0 .....	529
July 27, 2021 .....	533
Amazon IVS Broadcast SDK: Android 1.0.0 and iOS 1.0.0 .....	533
June 1, 2021 .....	534
Amazon IVS Player SDK: Android 1.3.3 and iOS 1.3.3 .....	534
May 19, 2021 .....	534
Amazon IVS Player SDK: Android 1.3.2 .....	534
May 5, 2021 .....	535
Amazon IVS Player SDK 1.3 .....	535
April 26, 2021 .....	539
Service Quotas Integration with CloudWatch Usage Metrics .....	539
April 13, 2021 .....	539
New CloudWatch Metrics .....	539
April 7, 2021 .....	539
Auto-Record to S3 (R2S3) .....	539
January 28, 2021 .....	540
Amazon IVS Player SDK: JW Player Integration 1.2.0 .....	540
December 16, 2020 .....	540

---

Amazon IVS Player: SDK for Android 1.2.1 .....	540
November 23, 2020 .....	540
Amazon IVS Player SDK 1.2.0 .....	540
November 12, 2020 .....	542
New Event Field, stream_id .....	542
November 9, 2020 .....	543
Add Metadata Viewing to Console .....	543
October 30, 2020 .....	543
CloudFormation Support .....	543
October 27, 2020 .....	543
Higher Limits for Channels, CCV, and CCB .....	543
October 9, 2020 .....	544
New Service Quotas and EventBridge Event .....	544
Amazon IVS Player: SDK for Web 1.1.2 .....	544
October 7, 2020 .....	544
Amazon IVS Player SDK 1.1.0 .....	544
September 14, 2020 .....	548
New Event Field, channel_name .....	548
August 19, 2020 .....	548
Playback Authorization (Private Channels) .....	548
August 11, 2020 .....	549
Amazon IVS Player: SDK for iOS 1.0.6 .....	549
August 5, 2020 .....	549
Using Amazon EventBridge with Amazon IVS .....	549
July 15, 2020 .....	549
Player Version 1.0 .....	549

# What is Amazon IVS Low-Latency Streaming?

Amazon Interactive Video Service (IVS) is a managed, live-video streaming service that allows you to:

- Create channels and start streaming in minutes.
- Build engaging, interactive experiences alongside low-latency live video.
- Distribute video at scale to a range of devices and platforms.
- Easily integrate into websites and apps.

Amazon IVS lets you focus on building your own interactive application and audience experience. With Amazon IVS, you don't need to manage infrastructure or develop and configure components of your video workflows, to be secure, reliable, and cost effective.

Amazon IVS supports streaming via several ingest protocols:

- RTMP (Real-Time Messaging Protocol), an industry standard for transmitting video over a network.
- RTMPS, the secure version of RTMP, running over TLS.
- SRT (Secure Reliable Transport), a relatively new, open-source protocol. SRT is designed to improve streaming over unreliable networks and protect against jitter, packet loss, and network bandwidth fluctuations.

In addition to the product documentation here, see <https://ivs.rocks/>, a dedicated site to browse published content (demos, code samples, blog posts), estimate cost, and experience Amazon IVS through live demos.

## Latency

*Latency* is the delay from when a camera captures a live stream to when the stream appears on a viewer's screen. Amazon IVS has functionality that can deliver video as follows:

- Low latency — Amazon IVS channels can deliver video with latency under 5 seconds.
- Real-time latency — IVS stages can deliver video with latency under 300ms. All participants in the stage experience this enhanced "real-time latency." (Note that if the stage is broadcast to an IVS channel, channel viewers get low latency.)

For a traditional Over-The-Top (OTT) stream, latency may be as high as 30 seconds.

Low latency is a critical component in building good interactive user experiences that enrich the audience experience. It allows the streamer, the brand, and the community to connect with live audiences in a direct and personal way.

Observed latency can vary between users due to:

- The geographic locations of the streamer and viewers.
- Network type and speed.
- Individual components in the streaming chain.
- Streaming protocols and output formats.

For more information, see [Reducing Latency](#) in *Amazon IVS Streaming Configuration*.

## Global Solution, Regional Control

### Streaming and Viewing are Global

You can use Amazon IVS to stream to viewers worldwide:

- When you stream, Amazon IVS automatically ingests video at a location near you.
- Viewers can watch your live streams globally via the Amazon IVS content-delivery network.

Another way of saying this is that the "data plane" is global. The data plane refers to streaming/ingesting and viewing.

### Control is Regional

While the Amazon IVS data plane is global, the "control plane" is regional. The control plane refers to the Amazon IVS console, API, and resources (channels, stream keys, playback key pairs, and recording configurations).

Another way of saying this is that Amazon IVS is a "regional AWS service." That is, Amazon IVS resources in each region are independent of similar resources in other regions. For example, a channel that you create in one region is independent of channels you create in other regions.

When you use resources (e.g., create a channel), you must specify the region in which it will be created. Subsequently, when you manage resources, you must do so from the same region where they were created.

If you use the ...	You specify the region by ...
Amazon IVS console	Using the <b>Select a Region</b> drop-down in the top right of the navigation bar.
Amazon IVS API	Using the appropriate service endpoint. See the <a href="#">Amazon IVS Low-Latency Streaming API Reference</a> .  (If you access the API through an SDK, set up the SDK's region parameter. See <a href="#">Tools to Build on AWS</a> .)
AWS CLI	Either: <ul style="list-style-type: none"><li>• Appending <code>--region &lt;aws-region&gt;</code> to your CLI command.</li><li>• Putting the region in your local AWS configuration file.</li></ul>

*Remember, regardless of the region in which a channel was created, you can stream to Amazon IVS from anywhere, and viewers can watch from anywhere.*

## Your Channel's Region

Your channel's region is part of the ARN (Amazon Resource Name) that is assigned when you create the channel. When you create a channel:

- The Amazon IVS console shows the ARN in the **General configuration** area of the page. Subsequently, the console always shows your region (location) on the top right.
- The Amazon IVS API returns the ARN in the channel object's `arn` field.

# Getting Started with IVS Low-Latency Streaming

This document takes you through the steps to set up your first Amazon Interactive Video Service (IVS) live stream.

## Topics

- [Step 1: Create an AWS Account](#)
- [Step 2: Set Up Root and Administrative Users](#)
- [Step 3: Set Up IAM Permissions](#)
- [Step 4: Create a Channel with Optional Recording](#)
- [Step 5: Set Up Streaming Software](#)
- [Step 6: View Your Live Stream](#)
- [Step 7: Check Your Service-Quota Limits \(Optional\)](#)
- [Step 8: Prevent Undesired Content and Viewers \(Recommended\)](#)
- [How to Disable Recording](#)

## Step 1: Create an AWS Account

To use Amazon IVS, you need an AWS account. If you don't already have one, you are prompted to create it when you sign up. To create an AWS account:

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call or text message and entering a verification code. Also, you will have to provide billing information, although the basic tier of service is free. You are not charged for any AWS services that you sign up for unless you use them.

3. After creating the account, you will get one email with your **Sign-in URL** and **User Name** and another email (from your AWS account administrator) with your password. You must change the password during your first sign-in.

If you want to use an existing AWS account, ensure that it uses an AWS region that is supported for Amazon IVS:

1. Navigate to the [Amazon IVS Console](#). If you see the usual IVS console page (showing "Global Solution, regional content"), you're fine; skip to [Step 2: Set Up Root and Administrative Users](#). If you are redirected to an AWS "unsupported region" page, you need to select a new region.
2. Select the appropriate tab (**Live streaming**, for IVS; **Stream chat**, for IVS Chat), then select one of the listed regions. *Note which region you choose; you will need it later.*

At any time, you can view your AWS account activity and manage your account by going to <https://aws.amazon.com/> and choosing **My Account**.

## Step 2: Set Up Root and Administrative Users

When you sign up for an AWS account, an AWS account root user is created. The root user has access to all AWS services and resources in the account. As a security best practice, [assign administrative access to an administrative user](#) and use the root user only to perform [tasks that require root user access](#).

### Secure Your AWS Account Root User

1. To sign in as the administrative user in the IAM Identity Center, use the sign-in URL that was sent to your email address when you created the IAM Identity Center user. For help signing in using an IAM Identity Center user, see [Signing in to the AWS access portal](#) in the *AWS Sign-In User Guide*.

For help signing in using root user, see [Signing in as the root user](#) in the *AWS Sign-In User Guide*.

2. Turn on multi-factor authentication (MFA) for your root user.

For instructions, see [Enable a virtual MFA device for your AWS account root user \(console\)](#) in the *IAM User Guide*.

### Create an Administrative User

You should create an administrative user so that you do not use the root user for everyday tasks.

- For your daily administrative tasks, assign administrative access to an administrative user in AWS IAM Identity Center (successor to AWS Single Sign-On). For instructions, see [Getting started](#) in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.

- To sign in as the administrative user in the IAM Identity Center, use the sign-in URL that was sent to your email address when you created the IAM Identity Center user. For help signing in using an IAM Identity Center user, see [Signing in to the AWS access portal](#) in the *AWS Sign-In User Guide*.

## Step 3: Set Up IAM Permissions

Next, you must create an AWS Identity and Access Management (IAM) policy that gives users a basic set of permissions (e.g., to create an Amazon IVS channel, get streaming information, and auto-record-to-S3) and assign that policy to users. You can either assign the permissions when creating a [new user](#) or add permissions to an [existing user](#). Both procedures are given below.

For more information (for example, to learn about IAM users and policies, how to attach a policy to a user, and how to constrain what users can do with Amazon IVS), see:

- [Creating an IAM User](#) in the *IAM User Guide*
- The information in [Amazon IVS Security](#) on IAM and "Managed Policies for IVS."
- For record-to-S3 functionality: [Using Service-Linked Roles](#) and [Auto-Record to Amazon S3](#) in the *Amazon IVS User Guide*

You can either use an existing AWS managed policy for Amazon IVS or create a new policy that customizes the permissions you want to grant to a set of users, groups, or roles. Both approaches are described below.

### Use an Existing Policy for IVS Permissions

In most cases, you will want to use an AWS managed policy for Amazon IVS. They are described fully in the [Managed Policies for IVS](#) section of *IVS Security*.

- Use the `IVSReadOnlyAccess` AWS managed policy to give your application developers access to all IVS Get and List API endpoints (for both low-latency and real-time streaming).
- Use the `IVSFullAccess` AWS managed policy to give your application developers access to all IVS API endpoints (for both low-latency and real-time streaming).

### Optional: Create a Custom Policy for Amazon IVS Permissions

Follow these steps:



1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**, then choose **Create policy**. A **Specify permissions** window opens..
3. In the **Specify permissions** window, choose the **JSON** tab, and copy and paste the following IVS policy to the **Policy editor** text area. (The policy does not include all Amazon IVS actions. You can add/delete (Allow/Deny) endpoint access permissions as needed. See [IVS Low-Latency Streaming API Reference](#) for details on IVS endpoints.)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ivs:CreateChannel",
        "ivs:CreateRecordingConfiguration",
        "ivs:GetChannel",
        "ivs:GetRecordingConfiguration",
        "ivs:GetStream",
        "ivs:GetStreamKey",
        "ivs:GetStreamSession",
        "ivs:ListChannels",
        "ivs:ListRecordingConfigurations",
        "ivs:ListStreamKeys",
        "ivs:ListStreams",
        "ivs:ListStreamSessions"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarms",
        "cloudwatch:GetMetricData",
        "s3:CreateBucket",
        "s3:GetBucketLocation",
        "s3:ListAllMyBuckets",
        "servicequotas:ListAWSDefaultServiceQuotas",
        "servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota",
        "servicequotas:ListServiceQuotas",
        "servicequotas:ListServices",

```

```
        "servicequotas:ListTagsForResource"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "iam:AttachRolePolicy",
      "iam:CreateServiceLinkedRole",
      "iam:PutRolePolicy"
    ],
    "Resource":
"arn:aws:iam::*:role/aws-service-role/ivs.amazonaws.com/
AWSServiceRoleForIVSRecordToS3*"
  }
]
```

4. Still in the **Specify permissions** window, choose **Next** (scroll to the bottom of the window to see this). A **Review and create** window opens.
5. On the **Review and create** window, enter a **Policy name** and optionally add a **Description**. Make a note of the policy name, as you will need it when creating users (below). Choose **Create policy** (at the bottom of the window).
6. You are returned to the IAM console window, where you should see a banner confirming that your new policy was created.

## Create a New User and Add Permissions

### IAM User Access Keys

IAM Access keys consist of an access key ID and a secret access key. They are used to sign programmatic requests that you make to AWS. If you don't have access keys, you can create them from the AWS Management Console. As a best practice, do not create root-user access keys.

*The only time that you can view or download a secret access key is when you create access keys. You cannot recover them later.* However, you can create new access keys at any time; you must have permissions to perform the required IAM actions.

Always store access keys securely. Never share them with third parties (even if an inquiry seems to come from Amazon). For more information, see [Managing access keys for IAM users](#) in the *IAM User Guide*.

## Procedure

Follow these steps:

1. In the navigation pane, choose **Users**, then choose **Create user**. A **Specify user details** window opens.
2. In the **Specify user details** window:
  - a. Under **User details**, type the new **User name** to be created.
  - b. Check **Provide user access to the AWS Management Console**.
  - c. When prompted, select **I want to create an IAM user**.
  - d. Under **Console password**, select **Autogenerated password**.
  - e. Check **Users must create a new password at next sign-in**.
  - f. Choose **Next**. A **Set permissions** window opens.
3. Under **Set permissions**, select **Attach policies directly**. A **Permissions policies** window opens.
4. In the search box, enter an IVS policy name (either an AWS managed policy or your previously created custom policy). When it is found, check the box to select the policy.
5. Choose **Next** (at the bottom of the window). A **Review and create** window opens.
6. On the **Review and create** window, confirm that all user details are correct, then choose **Create user** (at the bottom of the window).
7. The **Retrieve password** window opens, containing your **Console sign-in details**. *Save this information securely for future reference*. When you are done, choose **Return to users list**.

## Add Permissions to an Existing User

Follow these steps:

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**, then choose an existing user name to be updated. (Choose the name by clicking on it; do not check the selection box.)

3. On the **Summary** page, on the **Permissions** tab, choose **Add permissions**. An **Add permissions** window opens.
4. Select **Attach existing policies directly**. A **Permissions policies** window opens.
5. In the search box, enter an IVS policy name (either an AWS managed policy or your previously created custom policy). When the policy is found, check the box to select the policy.
6. Choose **Next** (at the bottom of the window). A **Review** window opens.
7. On the **Review** window, select **Add permissions** (at the bottom of the window).
8. On the **Summary** page, confirm that the IVS policy was added.

## Step 4: Create a Channel with Optional Recording

An Amazon IVS channel stores configuration information related to your live stream. You first create a channel and then contribute video to it using the channel's stream key to start your live stream.

As part of channel creation, the following items are assigned:

- An *ingest server* identifies a specific Amazon IVS component that receives the stream, along with an ingestion protocol (RTMPS or RTMP).
- Amazon IVS assigns a *stream key* when you create a channel and uses it to authorize streaming. ***Treat the stream key like a secret, since it allows anyone to stream to the channel.***
- A *playback URL* identifies the endpoint to start playback for a specific channel. This endpoint can be used globally. It automatically selects the best location from the Amazon IVS global content delivery network for a viewer to stream the video. (Note that Amazon IVS does not support custom domains for playback. *Do not proxy the playback URL with your own domain; that does not work and will cause issues.*)

You can create a channel — with or without recording — through the Amazon IVS console or the AWS CLI. Channel creation and recording are discussed below.

### Auto-Record to Amazon S3

You have the option of enabling recording for a channel. If the auto-record to S3 feature is enabled, all streams on the channel are recorded and saved to an Amazon S3 storage bucket that you own. Subsequently, the recording is available for on-demand playback.

*Setting this up is an advanced option.* By default, recording is disabled when a channel is created.

Before you can set up a channel for recording, you must create a *recording-configuration*. This is a resource which specifies an Amazon S3 location where the recorded streams for the channel are stored. You can create and manage recording configurations using the console or CLI; both procedures are given below. After you create the recording configuration, you associate it with a channel either when you create the channel (as described below) or later, by updating an existing channel. (In the API, see [CreateChannel](#) and [UpdateChannel](#).) You can associate multiple channels with the same recording configuration. You can delete a recording configuration that is no longer associated with any channels.

Keep in mind the following constraints:

- You must own the S3 bucket. That is, the account that sets up a channel to be recorded must own the S3 bucket where recordings will be stored.
- The channel, recording configuration, and S3 location must be in the same AWS region. If you create channels in other regions and want to record them, you must also set up recording configurations and S3 buckets in those regions.

Recording to your S3 bucket requires authorization with your AWS credentials. To give IVS the required access, an AWS IAM [Service-Linked Role](#) (SLR) is created automatically when the recording configuration is created: the SLR is limited to give IVS write permission only on the specific bucket.

To disable recording after you have enabled it, see [the section called “How to Disable Recording”](#).

Note that network issues between the streaming location and AWS or within AWS could result in some data loss while recording your stream. In these cases, Amazon IVS prioritizes the live stream over the recording. For redundancy, record locally via your streaming tool.

For more information (including how to set up post-processing or VOD playback on your recorded files), see [Auto-Record to Amazon S3](#).

## Console Instructions

These steps are divided into three phases: initial channel setup, set up to auto-record to Amazon S3 (optional), and final channel creation.

### Initial Channel Setup


1. Open the [Amazon IVS console](#).

(You can also access the Amazon IVS console through the [AWS Management Console](#).)

2. From the navigation bar, use the **Select a Region** drop-down to choose a region. Your new channel will be created in this region.
3. In the **Get started** box (top right), choose **Create Channel**.
4. Under **Channel configuration**, accept the **Default configuration**. Optionally, specify a **Channel name**. Channel names are not unique, but they provide a way for you to distinguish channels other than the channel ARN (Amazon Resource Name).

Note: **Custom configuration** can be used for specifying certain non-default values, such as channel type or RTMP (instead of RTMPS) ingest. Custom specifications are not documented here.

# Create channel [Info](#)

A channel is a unique configuration for streams. It includes broadcast configuration details (a server URL and stream key) for streaming software/hardware, and a playback URL for playing the stream. Channel configuration may affect pricing. [Amazon IVS Pricing](#) 

## ► How Amazon Interactive Video Service works

### Setup

#### Channel name

Maximum length: 128 characters. May include numbers, letters, underscores (\_) and hyphens (-).

#### Channel configuration

**Default configuration**  
Use the default video latency and configuration, optimized for live interactions.

**Custom configuration**  
Specify your own channel type and video latency configuration.

#### Channel type [Info](#)

Standard (broadcast and deliver live video up to 1080p Full HD, with transcoding and 1080p pass-through)

#### Video latency [Info](#)

Low (best for low-latency interactions with viewers)

#### Playback authorization [Info](#)

Disabled

#### Insecure ingest [Info](#)

Disabled

### Restrict playback [Info](#)

#### Playback restriction policy [Info](#)

Restrict playback by country and origin.

Enable playback restriction

5. If you want to auto-record to Amazon S3, continue with [Set Up to Auto-Record to Amazon S3 \(Optional\)](#) below. Otherwise, skip that and proceed directly to [Final Channel Creation](#).

## Set Up to Auto-Record to Amazon S3 (Optional)

Follow these steps to enable recording while creating a new channel:


1. On the **Create channel** page, under **Record and store streams**, turn on **Enable automatic recording**. Additional fields display, to choose an existing **Recording configuration** or create a new one.

### Record and store streams [Info](#)

**Auto-record to S3** [Info](#)  
For improved redundancy, always record locally via your streaming tool.

Enable automatic recording

Recording configuration

 **Associated costs**  
There are four cost components to consider when enabling record to S3: storage, request and data retrieval, data transfer, and data management. [Estimate data use.](#)

► **Tags** [Info](#)  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

2. Choose **Create recording configuration**. A new window opens, with options for creating an Amazon S3 bucket and attaching it to the new recording configuration.



## Create recording configuration ✕

### Recording configuration name – *optional*

*recording-configuration-1*

Maximum length: 128 characters. May include numbers, letters, underscores ( `_` ) and hyphens ( `-` ).

### Recording configuration

**Default configuration**  
Use the default settings for auto-recording video and thumbnails.

**Custom configuration**  
Specify your own video and thumbnail recording options.

### Recorded renditions

All renditions

### Thumbnail recording

Record at an interval - 60s

### Target thumbnail resolution

Source (same resolution as input stream)

### Thumbnail storage

Store thumbnails sequentially

### Merge fragmented streams

Disabled

### Storage

- Create a new Amazon S3 bucket
- Select an existing Amazon S3 bucket

### Bucket name

*ivs-stream-archive*

The bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#).

3. Fill out the fields:
  - a. Optionally enter a **Recording configuration name**.
  - b. Under **Recording configuration** accept the **Default configuration**. Note: **Custom configuration** can be used for specifying certain non-default values such as recorded renditions or merge fragmented streams. Custom specifications are not documented here.
  - c. Enter a **Bucket Name**.
4. Choose **Create recording configuration**, to create a new recording-configuration resource with a unique ARN. Typically, creation of the recording configuration takes a few seconds, but it can be up to 20 seconds. When the recording configuration is created, you are returned to the **Create channel** window. There, the **Record and store streams** area shows your new **Recording configuration**, with its **State** as **Active** and the S3 bucket (**Storage**) that you created.

## Record and store streams [Info](#)

**Auto-record to S3** [Info](#)

For improved redundancy, always record locally via your streaming tool.

Enable automatic recording

**Recording configuration**

configuration-1 ▼

↻

Create recording configuration

<p><b>State</b></p> <p><span style="color: green;">✔</span> Active</p>	<p><b>Storage</b></p> <p>s3-bucket-name <a href="#">↗</a></p>	<p><b>Recording prefix</b> <a href="#">Info</a></p> <p>s3://ivs-r2s3-ivsstoragebucket-1kem14abgbit8/ivs/v1/298083573632/&lt;attached_channel_id&gt;/</p>
<p><b>Recorded renditions</b></p> <p>All renditions</p>	<p><b>Merge fragmented streams</b></p> <p>Disabled</p>	<p><b>Thumbnail recording</b></p> <p>At 60-second intervals</p>
<p><b>Thumbnail storage</b></p> <p>Store thumbnails sequentially</p>	<p><b>Thumbnail resolution</b></p> <p>Source (same resolution as input stream)</p>	

**Associated costs** i

There are four cost components to consider when enabling record to S3: storage, request and data retrieval, data transfer, and data management. [Estimate data use.](#)

▶ **Tags** [Info](#)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Cancel
Create channel

## Final Channel Creation

1. Choose **Create channel**, to create a new channel with a unique ARN. A channel details page is displayed for the new channel; keep this open. (Note: if you did not enable recording, **Auto-record to S3** is set to **Disabled** and there is no **Recording configuration** section on the screen.)

# channel-live Info

[Edit](#) [Delete](#)

## General configuration

Channel name channel-live	Channel type Standard	Video latency Ultra-low
Playback authorization Disabled	Auto-record to S3 Enabled	ARN arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh

## ▶ Live stream

## Stream configuration Info

[Reset stream key](#)

Stream key [Show](#)  
 .....

Ingest server  
 rtmps://a1b2c3d4e5f6.global-contribute.live-video.net:443/app/

## ▶ Other ingest options

## Playback configuration Info

Playback URL  
 https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8

## Recording configuration Info

[Manage](#)

Recording configuration <a href="#">configuration-1</a>	Storage <a href="#">s3-bucket-name</a>	Recording prefix s3://qazwsxedcrfv-deleteme-1/ivs/v1/1234567891012/a1b2c3d4e5f6/
--	---	---

## Stream sessions Info

< 1 >

Stream ID	Start time	Duration
-----------	------------	----------

No past streams to display  
No past streams

## Tags Info

[Manage tags](#)

## 2. Important:

- In the **Stream configuration** area, note the **Ingest server** and **Stream key**. You will use these in the next step, to set up streaming.
- In the **Playback configuration** area, note the **Playback URL**. You will use it later, to play back your stream.

**Note:** To see SRT values (endpoint and passphrase), expand **Other ingest options**.

## CLI Instructions

Creating a channel with the AWS CLI is an advanced option and requires that you first download and configure the CLI on your machine. For details, see the [AWS Command Line Interface User Guide](#).

Follow one of the two procedures below, depending on whether you want to create a channel with or without recording enabled.

### Create a Channel without Recording

1. Run the `create-channel` command and pass in an optional name:

```
aws ivs create-channel --name test-channel
```

2. This returns a new channel:

```
{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "authorized": false,
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": false,
    "latencyMode": "LOW",
    "name": "channel-live",
    "playbackRestrictionPolicyArn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/abcdABCDefgh",
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "recordingConfigurationArn": "none",
    "srt": {
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
```

```

    "passphrase":
      "ZU5A3yrjGAKghUNDr0c5NXBhsPrjImtcKMNB1uh7oImwJQ3ijeyClvMKx1pPcGAMziICJ",
    },
    "tags": {},
    "type": "STANDARD"
  },
  "streamKey": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6",
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "tags": {},
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef"
  }
}

```

3. **Important:** Note the `ingestEndpoint`, `streamKey` value, and `playbackUrl`. You will use these to set up streaming and playback.

## Create a Channel with Recording

**Prerequisite:** Before starting this procedure, create an Amazon S3 bucket and note its ARN. See [Getting Started with Amazon S3](#). The S3 bucket must be in the same region where you will create a recording configuration; see the known issue in Step 1 below.

Then follow these steps to create the channel:

1. Run the `create-recording-configuration` command and pass in the ARN of an existing Amazon S3 bucket:

```
aws ivs create-recording-configuration --name configuration-1 --destination-configuration s3={bucketName=test-bucket}
```

Optionally, pass the `thumbnail-configuration` parameter to manually set the thumbnail-recording mode and thumbnail interval:

```
aws ivs create-recording-configuration --name configuration-1 --destination-configuration s3={bucketName=s3_bucket_name} --thumbnail-configuration recordingMode="INTERVAL",targetIntervalSeconds=60
```

Optionally, pass the `recording-reconnect-window-seconds` parameter to enable merge fragmented streams functionality:

```
aws ivs create-recording-configuration --name configuration-1 --destination-configuration s3={bucketName=test-bucket} --recording-reconnect-window-seconds 60
```

**Known issue:** In the us-east-1 region, if you use the AWS CLI to create a recording configuration, it returns success even if the S3 bucket is in a different region. In this case, the state of the recording configuration is CREATE\_FAILED (instead of ACTIVE). (In other regions, the CLI correctly returns failure if the bucket is in a different region.)

**Workaround:** Ensure that your S3 bucket is in the same region as the recording configuration. If you create a recording configuration in a different region as your S3 bucket, delete that recording configuration and create a new one with an S3 bucket from the correct region.

2. This returns a new recording configuration with a unique ARN. The state of the recording configuration is CREATING, indicating that it is in the process of being created.

```
{
  "recordingConfiguration": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
mhndauNa01te",
    "name": "configuration-1",
    "destinationConfiguration": {
      "s3": {
        "bucketName": "s3_bucket_name"
      }
    },
    "recordingReconnectWindowSeconds": 60,
    "state": "CREATING",
    "tags": {},
    "thumbnailConfiguration": {
      "recordingMode": "INTERVAL",
      "targetIntervalSeconds": 60
    }
  }
}
```

3. Typically, creation of the recording configuration takes a few seconds, but it can be up to 20 seconds. To check that the recording configuration has been created, run the `get-recording-configuration` command:

```
aws ivs get-recording-configuration --arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/mhndauNa01te"
```

4. This returns a response indicating that the recording configuration was created (state is ACTIVE):

```
{
  "recordingConfiguration": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/mhndauNa01te",
    "name": "configuration-1",
    "destinationConfiguration": {
      "s3": {
        "bucketName": "s3_bucket_name"
      }
    },
    "recordingReconnectWindowSeconds": 60,
    "state": "ACTIVE",
    "tags": {},
    "thumbnailConfiguration": {
      "recordingMode": "INTERVAL",
      "targetIntervalSeconds": 60
    }
  }
}
```

5. To create a channel and enable recording on it, run the `create-channel` command and pass in the recording-configuration ARN:

```
aws ivs create-channel --name channel-live --recording-configuration-arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/mhndauNa01te"
```

Alternately, to enable recording on an existing channel, run the `update-channel` command and pass in the recording-configuration ARN:

```
aws ivs update-channel --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" --recording-configuration-arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/mhndauNa01te"
```



6. This returns a channel object with a non-"none" value for `recordingConfigurationArn`, indicating that recording is enabled. (The response below is from `create-channel`. The `update-channel` response does not include the `streamKey` object.)

```
{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "authorized": false,
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": false,
    "latencyMode": "LOW",
    "name": "channel-live",
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/mhndauNa01te",
    "srt": {
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
      "passphrase":
        "ZU5A3yrjGAKghUNDr0c5NXBhsPrj1mtcKMNBluh7oImwJQ3ijeyClvMKxlpPcGAMziICJ",
    },
    "tags": {},
    "type": "STANDARD"
  },
  "streamKey": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6",
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "tags": {},
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef"
  }
}
```

7. **Important:** Note the `ingestEndpoint`, `streamKey` value, and `playbackUrl`. You will use these to set up streaming and playback.

## Step 5: Set Up Streaming Software

You can stream (low-latency) to Amazon IVS with:

- The native [IVS broadcast SDKs](#), which support RTMPS. We recommended this, especially for production scenarios.

- The [Amazon IVS console](#) — This is suitable for testing streams.
- Other streaming software and hardware encoders — You can use any streaming encoder that supports the RTMP, RTMPS, or SRT protocols. Several examples are described below, using Open Broadcast Software (OBS) and FFmpeg with RTMPS and SRT. RTMPS enables high security via use of an encrypted TLS stream.

Key encoder settings are keyframe interval (2 seconds) and resolution/bitrate/frame rate (which are interrelated). For more detail on encoder settings, see:

- [Streaming Configuration](#) in the *Amazon IVS User Guide*
- This blog post: [Setting Up for Streaming with Amazon Interactive Video Service](#)

#### Notes:

- The maximum duration of Amazon IVS streams is 48 hours. After that, the stream is terminated and the streaming session is disconnected. A successful reconnect (automatically or manually) starts a new stream.
- If your encoder stops sending data (e.g., due to a temporary network issue), Amazon IVS waits for 30 seconds. If no broadcaster data is received during this time, Amazon IVS disconnects.

## Streaming with the Amazon IVS Broadcast SDK

To broadcast from your iOS or Android applications, you can use the Amazon IVS broadcast SDK. The broadcast SDK leverages the Amazon IVS architecture and will see continual improvement and new features, alongside Amazon IVS. As a native mobile broadcast SDK, it is designed to minimize the performance impact on your application and on the devices with which your users access your application.

To broadcast from ...	You can use ...	Notes
Your Android or iOS applications	Amazon IVS Android or iOS broadcast SDK	As a native mobile broadcast SDK, it is designed to minimize the performance impact on your application and on the devices with which your users access your application.

To broadcast from ...	You can use ...	Notes
A web environment	Amazon IVS Web broadcast SDK	As a web broadcast SDK, the Amazon IVS Web Broadcast SDK allows you to broadcast from web environments using WebRTC. It boasts cross-browser and cross-platform support.

For details, see [IVS Broadcast SDK](#).

## Streaming with the Amazon IVS Console

1. Open the [Amazon IVS console](#).

(You can also access the Amazon IVS console through the [AWS Management Console](#).)

2. In the navigation pane, select **Channels**. (If the nav pane is collapsed, expand it by selecting the hamburger icon.)
3. Select the channel to which you want to broadcast, to go to its details page.
4. Select the **Broadcast** tab. (Tabs are below the **General Configuration** section.)
5. You will be prompted to grant the IVS console access to your camera and microphone; **Allow** those permissions.
6. Toward the bottom of the **Broadcast** tab, use the dropdown boxes to select input devices for the microphone and camera.
7. To begin streaming, select **Start broadcasting**.
8. To view the live stream, go to the **Playback** tab.

**Note:** After you start the stream, expect a brief delay (usually under 30 seconds) before it is viewable in the playback tab.

You can use this feature to simultaneously broadcast to multiple channels.

**Note:** Streaming from the console consumes resources, and you will incur live-video input costs. To learn more, see [Live Video Input Costs](#) on the IVS Pricing page.

## Streaming with OBS Studio using RTMPS

[\(OBS Studio\)](#) is a free, open-source software suite for recording and live streaming. OBS provides real-time source and device capture, scene composition, encoding, recording, and streaming.

Follow these steps to get up and running quickly with OBS Studio v30.2 or later:

1. Download and install the software: <https://obsproject.com/download>.
2. Run the OBS Studio **Auto-Configuration Wizard**, which appears when you load OBS Studio for the first time. Follow the steps and accept the defaults.
3. At **Stream Information**, choose **Amazon IVS** from the **Service** dropdown and enter the **Stream Key**.

If you created the channel with the Amazon IVS console, the **Stream Key** you enter in OBS is the **Stream key** from the console: `sk_us-west-2_abcd1234efgh5678ijkl`

If you created the channel with the AWS CLI, the **Stream Key** you enter in OBS is the **streamKey value** from the CLI response `sk_us-west-2_abcd1234efgh5678ijkl`

4. For **Video Output Resolution** and **Bitrate**, refer to [Channel Types](#) in *Amazon IVS Streaming Configuration*. If either value chosen by the OBS wizard exceeds the values allowed by Amazon IVS, you should manually adjust the values to avoid a failed connection to Amazon IVS. After the wizard completes:
  - a. To adjust video resolution, use **Settings > Video > Output (Scaled) Resolution**.
  - b. To adjust video bitrate, use **Settings > Output > Streaming > Video Bitrate**.
5. We recommend a 2-second **Keyframe Interval** to improve the stream stability and avoid buffering in the viewer playback. After the wizard completes, go to **Settings > Output > Output Mode**, select **Advanced**, and on the **Streaming** tab, ensure that **Keyframe Interval** is 2.
6. In the OBS Studio main window, choose **Start Streaming**.

For more on streaming with OBS Studio, see [OBS Studio Quickstart](#).

You can modify your OBS settings manually later:

1. Choose **Settings > Stream**.
2. Choose **Amazon IVS** from the dropdown.
3. Paste in the **Stream Key**.

You can run the wizard again at any time: choose **Tools > Auto-Configuration Wizard**.

Optionally, in **Settings > General**, enable local recording to save your live stream for later use. As mentioned earlier, network issues between the broadcast and AWS or within AWS could result in some data loss while recording your stream. In these cases, Amazon IVS prioritizes the live stream over the recording. Recording locally via your streaming tool provides redundancy.

It's advisable to check for OBS Studio updates regularly and update to the most current version. (For instance, if you get a "Failed to connect to server" error, you may be using an old version of OBS Studio that does not support RTMPS.)

## Streaming with OBS Studio using SRT

Follow these steps to get up and running quickly with the Secure Reliable Transport protocol:

1. Download and install the software: <https://obsproject.com/download>.
2. Run the OBS Studio **Auto-Configuration Wizard**, which appears when you load OBS Studio for the first time. Follow the steps and accept the defaults.
3. At **Stream Information**, choose **Custom...** from the **Service** dropdown and enter the **Server (Ingest server)** and **Stream Key**.

If you created the channel with the AWS CLI:

- The **Server** you enter in OBS is a combination of five things:

- An ingestion protocol: `srt://`
- The **endpoint** from the `srt` struct in the CLI response:

```
a1b2c3d4e5f6.srt.live-video.net
```

- A port: `9000`
- A streamid, which is the **streamKey value** from the CLI response:

```
sk_us-west-2_abcd1234efgh5678ijkl
```

- A passphrase, used to encrypt the content. Use this only if **insecure ingest** is not enabled.

```
ZU5A3yrjGAKghUNDr0c5NXBhsPrj1mtcKMNB1uh7oImwJQ3ijeyClvMKxlpPcGAMziICJ
```

The complete entry is:

```
srt://a1b2c3d4e5f6.srt.live-video.net:9000?streamid=sk_us-west-2_abcd1234efgh5678ijkl&passphrase=ZU5A3yrjGAKghUNDr0c5NXBhsPrj1mtcKMNB1uh7oImwJQ3ijeyClvMKxlpPcGAMziICJ
```

- The **Stream Key** you enter in OBS will remain empty for the SRT protocol.
4. For **Video Output Resolution** and **Bitrate**, refer to [Channel Types](#) in *Amazon IVS Streaming Configuration*. If either value chosen by the OBS wizard exceeds the values allowed by Amazon IVS, you should manually adjust the values to avoid a failed connection to Amazon IVS. After the wizard completes:
    - a. To adjust video resolution, use **Settings > Video > Output (Scaled) Resolution**.
    - b. To adjust video bitrate, use **Settings > Output > Streaming > Video Bitrate**.
  5. We recommend a 2-second **Keyframe Interval** to improve the stream stability and avoid buffering in the viewer playback. After the wizard completes, go to **Settings > Output > Output Mode**, select **Advanced**, and on the **Streaming** tab, ensure that **Keyframe Interval** is 2.
  6. In the OBS Studio main window, choose **Start Streaming**.

You can modify your OBS settings manually later:

1. Choose **Settings > Stream**.
2. Choose **Custom** from the dropdown.
3. Paste in the **Server** and/or **Stream Key**.

You can run the wizard again at any time: choose **Tools > Auto-Configuration Wizard**.

Optionally, in **Settings > General**, enable local recording to save your live stream for later use. As mentioned earlier, network issues between the broadcast and AWS or within AWS could result in some data loss while recording your stream. In these cases, Amazon IVS prioritizes the live stream over the recording. Recording locally via your streaming tool provides redundancy.

It's advisable to check for OBS Studio updates regularly and update to the most current version. (For instance, if you get a "Failed to connect to server" error, you may be using an old version of OBS Studio that does not support RTMPS.)

## Streaming a Recorded Video with FFmpeg using RTMPS

Follow these steps:

1. Download and install FFmpeg: <https://www.ffmpeg.org/download.html>.
2. Set \$VIDEO\_FILEPATH to the location of an MP4 video to stream:

```
VIDEO_FILEPATH=/home/test/my_video.mp4
```

3. Set `STREAM_KEY` to your StreamKey **value**:

```
STREAM_KEY=sk_us-west-2_abcd1234efgh5678ijkl
```

4. Set `INGEST_ENDPOINT` to your **ingestEndpoint** (from the AWS CLI):

```
INGEST_ENDPOINT=a1b2c3d4e5f6.global-contribute.live-video.net
```

5. Start streaming with the following terminal command (this is all one line):

```
ffmpeg -re -stream_loop -1 -i $VIDEO_FILEPATH -r 30 -c:v libx264 -pix_fmt yuv420p  
-profile:v main -preset veryfast -x264opts "nal-hrd=cbr:no-scenecut" -minrate  
3000 -maxrate 3000 -g 60 -c:a aac -b:a 160k -ac 2 -ar 44100 -f flv rtmps://  
$INGEST_ENDPOINT:443/app/$STREAM_KEY
```

*Note, the above command is an example. For production streaming, tune the parameters to your needs.*

## Streaming a Recorded Video with FFmpeg using SRT

1. Download and install FFmpeg: <https://www.ffmpeg.org/download.html>. If you are using an old/compiled version of FFmpeg, build a new version with the `--enable-libsrt` flag.
2. Verify that SRT is available for use in FFmpeg: run the following command and ensure that `libsrt` is in the output. If `libsrt` is not there, rebuild or get a newer version of FFmpeg which supports SRT.

```
ffmpeg -version | grep enable-libsrt
```

3. Set `VIDEO_FILEPATH` to the location of an MP4 video to stream:

```
VIDEO_FILEPATH=/home/test/my_video.mp4
```

4. Set `STREAM_KEY` to your StreamKey **value**:

```
STREAM_KEY=sk_us-west-2_abcd1234efgh5678ijkl
```

5. Set `INGEST_ENDPOINT` to your **endpoint** (from the AWS CLI under the `srt` object):

```
INGEST_ENDPOINT=a1b2c3d4e5f6.srt.live-video.net
```

6. Set `PASSPHRASE` to your **passphrase** (from the AWS CLI under the `srt` object). Use passphrase only if insecure ingest is not enabled for the channel.

```
PASSPHRASE=ZU5A3yrjGakghUNDr0c5NXBhsPrj1mtcKMNBluh7oImwJQ3ijeyClvMKxlpPcGAMziICJ
```

7. Start streaming with the following terminal command (this is all one line):

```
ffmpeg -re -i $VIDEO_FILEPATH -c copy -f mpegts "srt://$INGEST_ENDPOINT:9000?streamid=$STREAM_KEY&passphrase=$PASSPHRASE"
```

## Step 6: View Your Live Stream

You can view your live stream with:

- The native [IVS player SDKs](#).
- The [Amazon IVS console](#).

### Viewing with the Amazon IVS Player SDKs

1. Set up the IVS Player. Start with the [IVS Player SDK overview](#), then read the appropriate platform-specific Player guide(s).
2. From the [Amazon IVS console](#), get the **Playback URL** that was generated when you created your channel. (See [Final Channel Creation](#) earlier in this *Getting Started* guide.)
3. Call `player.load()` with the playback URL.

### Viewing with the Amazon IVS Console

1. Open the [Amazon IVS console](#).

(You can also access the Amazon IVS console through the [AWS Management Console](#).)

2. On the navigation pane, choose **Live channels**. (If the nav pane is collapsed, first open it by choosing the hamburger icon.)



3. Choose the channel whose stream you want to view, to go to a details page for that channel.

The live stream is playing in the **Live stream** section of the page.

**Note:** Playback from the console consumes resources, and you will incur live-video output costs. To learn more, see [Live Video Output Costs](#) on the IVS Pricing page.

**Note:** After you start streaming, there is a short delay (up to 30 seconds, usually less) before your stream can be viewed in the console.

## Step 7: Check Your Service-Quota Limits (Optional)

All accounts have limits on the number of concurrent viewers and concurrent broadcasts. *Ensure that your limits are adequate and request an increase if needed, especially if you are planning a large streaming event.* For details, see [Service Quotas \(Low-Latency Streaming\)](#).

## Step 8: Prevent Undesired Content and Viewers (Recommended)

Malicious users may try to re-stream undesirable content (e.g., professional sports) on your platform, or try to embed your platform's streams on another website without permission. This kind of streaming can dramatically increase the amount of live-streamed video that your application is serving as well as the costs associated with it, without adding value to your business. In addition to providing you with controls to stop active streams, Amazon IVS provides resources to help detect and prevent this kind of behavior in the first place; see [Undesired Content and Viewers](#).

To constrain playback to specific origins and/or countries, use a playback restriction policy.

### Console Instructions (Playback Restriction Policy)

1. Create a playback restriction policy
  - a. [Open the Amazon IVS console](#). On the left navigation pane, select **Playback security > Playback restriction policies**.
  - b. Select **Create policy**.
  - c. Optionally, name the policy.
  - d. Optionally, toggle **Strict origin enforcement** (see note below).
  - e. Specify **Allowed countries** and **Allowed origins**.

- f. Select **Create policy**.
2. Attach this policy to a new or existing channel
    - a. Create a new channel or edit an existing channel.
    - b. In the **Restrict playback section** (of the **Create channel** or **Update channel** window), select **Enable playback restriction**.
    - c. From the **Playback restriction policy** drop-down list, select the policy you created in Step 1.
    - d. Select **Create channel** (for a new channel) or **Save** (to update an existing channel).

**Note on strict origin enforcement:** This is an optional setting that can be used to strengthen the origin restriction specified with allowed origins. By default, the origin restriction applies only to the multivariant playlist. If strict origin enforcement is enabled, the server will enforce a requirement that the requesting origin matches the token for all playback requests (including multivariant playlist, variant playlist, and segments). This means that all clients (including non-browser clients) will have to provide a valid origin-request header with each request. Use the `setOrigin` method to set the header in the IVS iOS and Android player SDKs. It is set automatically in web browsers except iOS Safari. For iOS Safari, you need to add `crossorigin="anonymous"` to the video element, to ensure that the origin request header is sent. Example: `<video crossorigin="anonymous"></video>`.

**Note on mapping between IP addresses and countries:** IVS determines the location of your users by using a third-party database. The accuracy of the mapping between IP addresses and countries varies by region. Based on recent tests, the overall accuracy is 99.8%. If IVS can't determine a user's location, IVS serves the content that the user requested.

## CLI Instructions (Playback Restriction Policy)

1. Create a playback restriction policy. Here is an example. *For the `allowed-countries` and `allowed-origins` fields, replace the example values below with your actual values, or delete one or both fields, depending on your use case.*

```
aws ivs create-playback-restriction-policy --name test-playback-restriction-policy
--enable-strict-origin-enforcement --allowed-countries "US","JP" --allowed-origins
"https://example1.com","https://*.example2.com"
```

This returns a new playback restriction policy. For its fields, see [PlaybackRestrictionPolicy](#) in the *IVS Low-Latency Streaming API Reference*.

2. Attach the new policy to a channel. For an existing channel, run `update-channel` and pass in the ARN of the playback restriction policy created in the previous step:

```
aws ivs update-channel --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" --playback-restriction-policy-arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/abcdABCDefgh"
```

For a new channel, include the `--playback-restriction-policy-arn` statement during [channel creation](#).

## How to Disable Recording

To disable Amazon S3 recording on an existing channel:

- Console — On the details page for the relevant channel, in the **Record and store** streams section, choose **Disabled** and then choose **Save Channel**. This removes the recording configuration's association with the channel; streams on that channel will no longer be recorded.
- CLI — Run the `update-channel` command and pass in the recording-configuration ARN as an empty string:

```
aws ivs update-channel --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" --recording-configuration-arn ""
```

This returns a channel object with an empty string for `recordingConfigurationArn`, indicating that the recording is disabled.

# Enabling Multiple Hosts on an Amazon IVS Stream

Amazon Interactive Video Service (IVS) enables developers to build applications that combine video and audio from multiple broadcasters (also referred to as *hosts*) into one live stream.

Use cases include:

- Guest spots – Broadcasters can invite viewers into the broadcast. This opens the door to collaborative content like karaoke and Q&A.
- Versus (VS) mode – Broadcasters are matched with each other to compete (e.g., in a singing competition).
- Group broadcasts – Multiple speakers can converse with each other in front of a large audience.

To add multiple broadcasters to a live stream, you need to use both IVS Real-Time Streaming and IVS Low-Latency Streaming. IVS Real-Time Streaming is used to combine video and audio streams; Low-Latency Streaming, to broadcast the combined stream to viewers.

Real-Time Streaming provides a resource called a stage, a virtual space where broadcasters (hosts) can exchange audio and video in real time. You can then broadcast a stage to channels to reach a larger audience, and you can build applications where audience members can be brought "on stage" to contribute to the live conversation.

For more information about IVS Real-Time Streaming, see:

- [IVS Real-Time Streaming User Guide](#)
  - The IVS Broadcast SDKs incorporate real-time functionality. See the Guides for those SDKs: [Web](#), [Android](#), and [iOS](#), especially the sections on "Publishing and Subscribing."
- [IVS Real-Time Streaming API Reference](#)

## Getting Started

### Console Instructions

To create a new stage and a participant token for it, follow these steps:

1. Open the [Amazon IVS console](#).

(You can also access the Amazon IVS console through the [AWS Management Console](#).)

2. On the left navigation pane, select **Stages**, then select **Create stage**. The **Create stage** window appears.

Amazon IVS > Video > Stages > Create stage

## Create stage [Info](#)

A stage allows participants to send and receive video and audio with others in real time. You can broadcast a stage to a channel, allowing viewers to see and hear stage participants without needing to join the stage directly. [Learn more](#)

▶ **How Amazon IVS stages work**

### Setup

Stage name – *optional*

Maximum length: 128 characters. May include numbers, letters, underscores (\_) and hyphens (-).

▶ **Tags** [Info](#)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Cancel **Create stage**

3. Optionally enter a **Stage name**. Select **Create stage** to create the stage. The stage details page appears, for the new stage.
4. Select **Create a participant token**.
5. In the **Create a participant token** dialog, enter a User ID and select **Create a participant token**. The token appears at the top of the **Participant tokens** table. Click the "Copy token" icon (to the left of the participant token) to copy the token.

## CLI Instructions

Using the AWS CLI is an advanced option and requires that you first download and configure the CLI on your machine. For details, see the [AWS Command Line Interface User Guide](#).

Now you can use the CLI to create and manage resources. The stage API is under the `ivs-realtime` namespace. For example, to create a stage:

```
aws ivs-realtime create-stage --name "test-stage"
```

The response is:

```
{
  "stage": {
    "arn": "arn:aws:ivs:us-west-2:376666121854:stage/VSWjvX5X0kU3",
    "name": "test-stage"
  }
}
```

To create a participant token for that stage:

```
aws ivs-realtime create-participant-token --stage-arn arn:aws:ivs:us-west-2:376666121854:stage/VSWjvX5X0kU3
```

The response is:

```
{
  "participant": {
    "participantId": "jFpWmveENolS",
    "expirationTime": "2022-08-26T19:17:00+00:00",
    "token":
      "eyJhbGciOiJIUzI1NiIsInR5cGEiOiJ1cm5pdCI7eyJ1c2Vybm90eSI6ImVzLnJlcnVzIn0=eyJhbGciOiJIUzI1NiIsInR5cGEiOiJ1cm5pdCI7eyJ1c2Vybm90eSI6ImVzLnJlcnVzIn0=TaKj1lW9Qac6c5xBrdAk"
  }
}
```

## Broadcasting a Stage: Client-Side versus Server-Side Composition

When developers want to broadcast a stage to an IVS channel, they have two choices:

- With *client-side composition*, a host connects to a stage, downloads videos from other hosts, combines them into one stream, and broadcasts the mixed stream to an IVS channel. This approach allows for a high degree of layout flexibility: the app developer can control the look of the composition using the mixer API. However, client-side composition requires more client CPU resources to create the composition and more bandwidth to broadcast it. Also, if the host broadcasting the stage has network issues, they may impact the live stream for viewers.

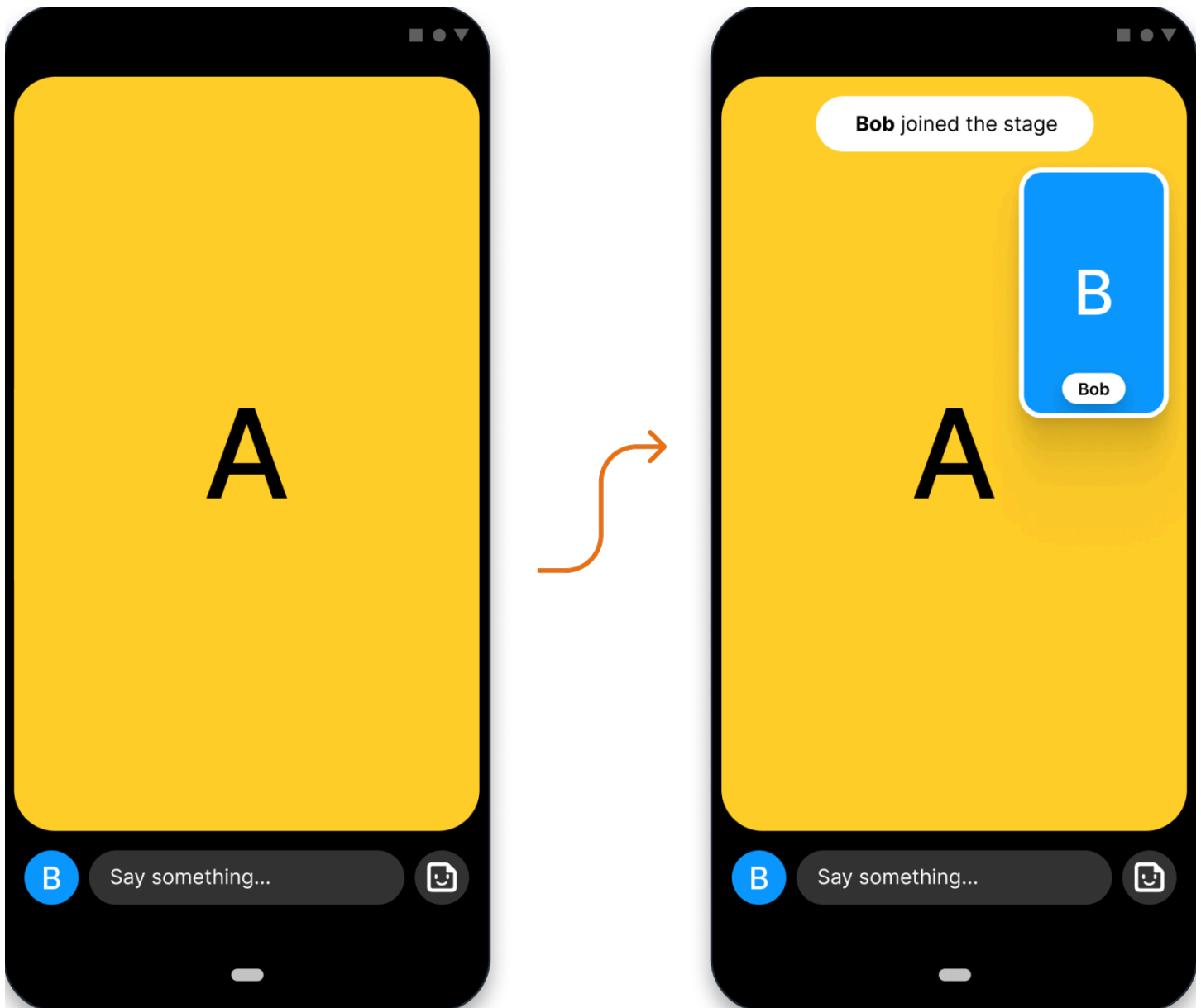
Client-side composition is the preferred choice when users need a highly personalized view of the broadcast content, such as incorporating overlays and customizing elements that aren't compatible with server-side composition.

- With *server-side composition*, clients offload the composition and broadcasting of an IVS stage to a cloud service. Server-side composition and RTMP broadcast to a channel are invoked through IVS control-plane endpoints in the stage's home region. Server-side composition offers numerous benefits, making it an attractive choice for users seeking efficient and reliable live streaming.
  - **Reduced client load** — With server-side composition, the burden of combining audio and video sources is shifted from individual client devices to the server itself. Server-side composition eliminates the need for client devices to use their CPU and network resources for compositing the view and transmitting it to IVS.
  - **Resilience** — By centralizing the composition process on the server, the broadcast becomes more robust. Even if a publisher device experiences technical limitations or network fluctuations, the server can adapt and provide a smoother stream to all the audience.
  - **Bandwidth efficiency** — Since the server handles the composition, stage publishers do not have to spend extra bandwidth broadcasting the video to an IVS channel.

For more information, see [Server-Side Composition](#) in the *IVS Real-Time User Guide*.

## Demo

Scenario: Alice (A) is broadcasting to her Amazon IVS channel and wants to invite Bob (B) on stage as a guest. (In a real broadcast, A and B would be images of Alice and Bob.)



## 1. Create a Stage

Here is a [CreateStage](#) request using the Amazon IVS Stage API:

```
POST /CreateStage HTTP/1.1
Content-type: application/json
{
  "name": "string",
  "participantTokenConfigurations": [
    {
      "userId": "9529828585",
      "attributes": {"displayName": "Alice"}
```



```
    },
    {
      "userId": "4875935192",
      "attributes": {"displayName": "Bob"}
    }
  ]
}
```

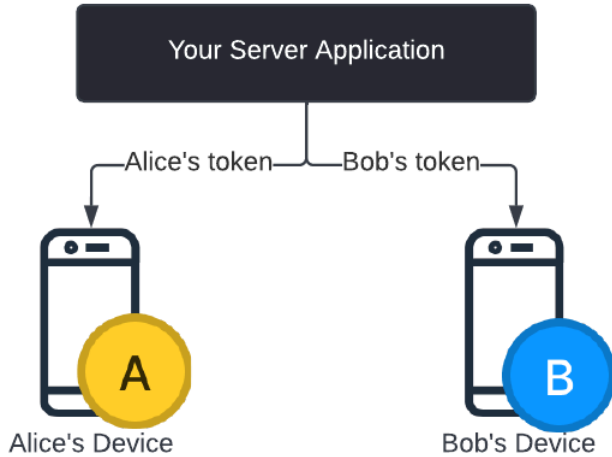
You can pre-create participant tokens when you create a stage, as is done here. You also can create tokens for an existing stage, by calling [CreateParticipantToken](#). For each participant, you can pass in a custom `userId` and set of attributes. (**Important:** The attributes and `userId` request fields are exposed to all stage participants. These should not be used for personally identifying, confidential, or sensitive information.)

Here is the network response to the request above:

```
HTTP/1.1 200
Content-type: application/json
{
  "stage": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
    "name": "alice-stage"
  },
  "participantTokens": [
    {
      "participantId": "e94e506e-f7...",
      "token": "eyJhbGciOiJ...\"",
      "userId": "9529828585",
      "attributes": {"displayName": "Alice"},
      "expirationTime": number
    },
    {
      "participantId": "b5c6a79a-6e...",
      "token": "eyJhbGciOiJ...\"",
      "userId": "4875935192",
      "attributes": {"displayName": "Bob"},
      "expirationTime": number
    }
  ]
}
```

## 2. Distribute Participant Tokens

The client now has a token for Alice (A) and Bob (B). By default, tokens are valid for 1 hour; optionally you can pass in a custom duration when you create the stage. Tokens can be used to join a stage.

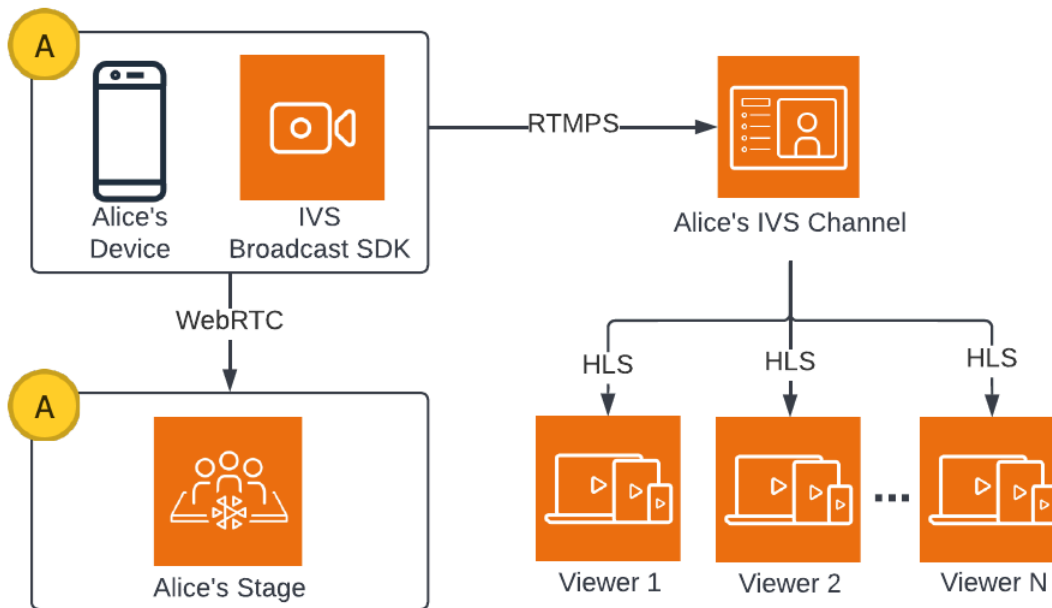


You will need a way to distribute tokens from your server to each client (e.g., via a WebSocket channel). We do not provide this functionality.

## 3. Join the Stage

Participants can join the stage via the Amazon IVS Broadcast SDK on Android or iOS. You can configure the video quality of each participant. Here we show Alice joining the stage first.

Here is an architecture overview:



And here is an Android code sample for joining the stage. The code snippet below would run on Alice's device. In the `join()` call, Alice joins the stage. The figure above shows the result of this code execution: Alice has joined the stage and is publishing to it (in addition to broadcasting to her channel, which she started doing in step 1).

```
// Create streams with the front camera and first microphone.
var deviceDiscovery = DeviceDiscovery(context)
var devices : List<Device> = deviceDiscovery.listLocalDevices()
var publishStreams = ArrayList<LocalStageStream>()

// Configure video quality if desired
var videoConfiguration = StageVideoConfiguration()

// Create front camera stream
var frontCamera = devices.find { it.descriptor.type ==
    Device.Descriptor.DeviceType.Camera && it.descriptor.position ==
    Device.Descriptor.Position.FRONT }
var cameraStream = ImageLocalStageStream(frontCamera, videoConfiguration)
publishStreams.add(cameraStream)

// Create first microphone stream
var microphone = devices.find { it.descriptor.type ==
    Device.Descriptor.DeviceType.Microphone }
var microphoneStream = AudioLocalStageStream(microphone)
publishStreams.add(microphoneStream)
```

```
// A basic Stage.Strategy implementation that indicates the user always wants to
// publish and subscribe to other participants.
// Provides the front camera and first microphone as publish streams.

override fun shouldPublishFromParticipant(stage: Stage, participantInfo:
    ParticipantInfo) : Boolean {
    return true
}

override fun shouldSubscribeToParticipant(stage: Stage, participantInfo:
    ParticipantInfo) : Stage.SubscribeType {
    return Stage.SubscribeType.AUDIO_VIDEO
}

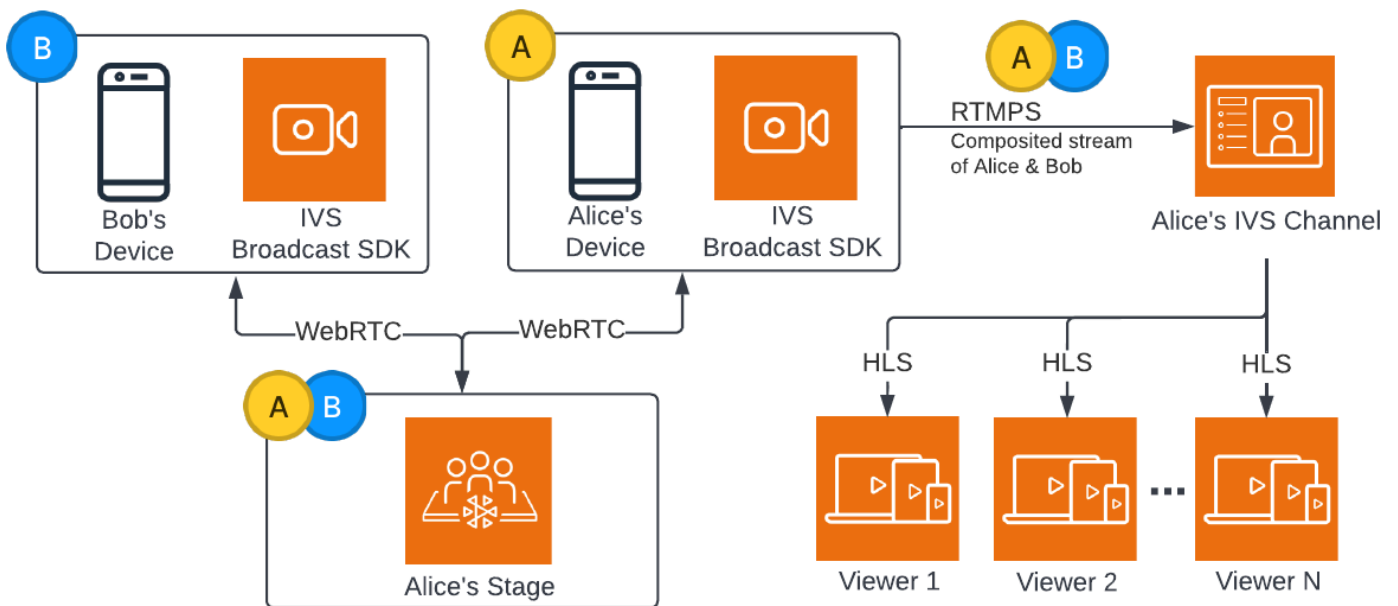
override fun stageStreamsToPublishForParticipant(stage: Stage, participantInfo:
    ParticipantInfo): List<LocalStageStream> {
    return publishStreams
}

// Create Stage using the strategy and join
var stage = Stage(context, token, strategy)

try {
    stage.join()
} catch (exception: BroadcastException) {
    // handle join exception
}
```

## 4. Broadcast the Stage

### Client-Side Composition



Here is an Android code sample for broadcasting the stage:

```
var broadcastSession = BroadcastSession(context, broadcastListener, configuration,
    null)

// StageRenderer interface method to be notified when remote streams are available
override fun onStreamsAdded(stage: Stage, participantInfo: ParticipantInfo, streams:
    List<StageStream>) {

    var id = participantInfo.participantId

    // Create mixer slot for remote participant
    var slot = BroadcastConfiguration.Mixer.Slot.with { s ->
        s.name = id
        // Set other properties as desired
        ...
        s
    }

    broadcastSession.mixer.addSlot(slot)

    // Attach remote stream devices, bind to mixer slot
```

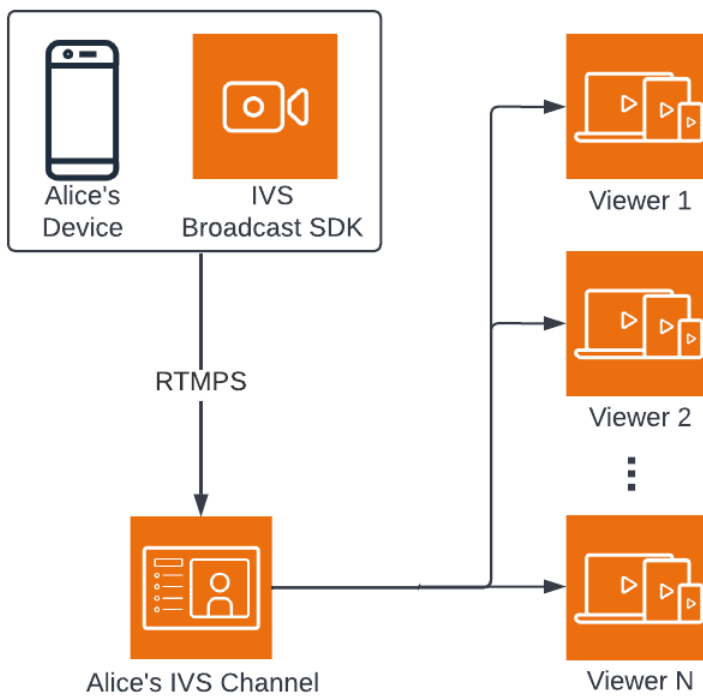
```
streams.forEach { stream ->
    broadcastSession.attachDevice(stream.getDevice())
    broadcastSession.mixer.bind(stream.getDevice(), id)
}
}

// Start broadcasting
try {
    broadcastSession.start(IVS_RTMP_URL, IVS_STREAM_KEY)
} catch (exception: BroadcastException) {
    // handle exception
}
```

The Android and iOS Amazon IVS Broadcast SDKs have callbacks triggered by the status of participants (e.g., `onStreamsAdded` and `onStreamsRemoved`), to simplify building a dynamic UI. This is shown in the first part of the code sample: when Bob's video and audio are available, Alice is notified via an `onStreamsAdded` callback.

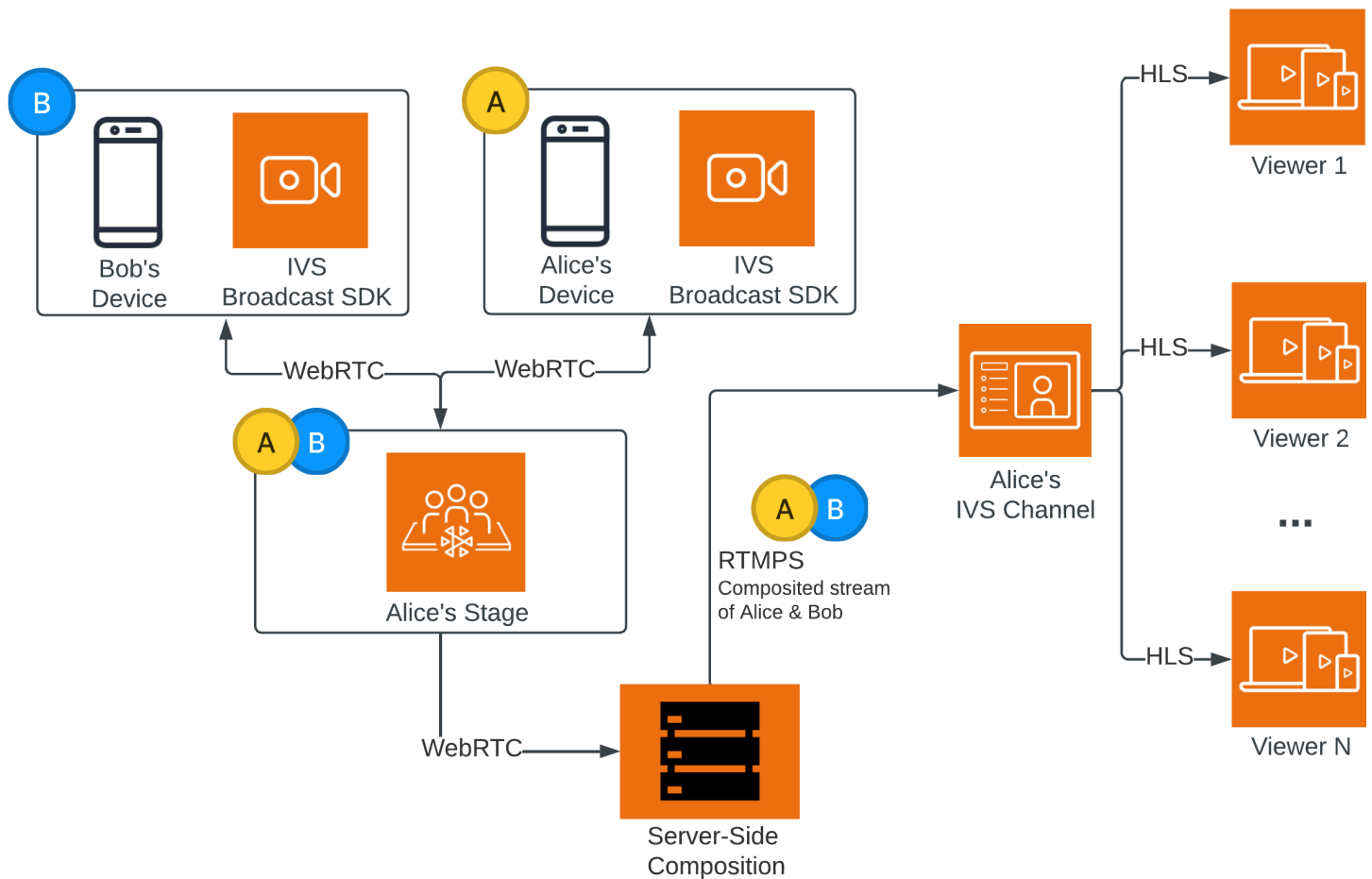
Alice can then add Bob's video and audio to the mixer, to be included in the RTMP broadcast for the wider audience of her channel. This is shown in the remainder of the code sample.

Now Alice is broadcasting to multiple viewers, via the Amazon IVS Android Broadcast SDK. Here is what this looks like architecturally:



## Server-Side Composition

For comparison, here is how [server-side composition](#) works. (For details, see [Server-Side Composition](#) in the *IVS Real-Time User Guide*.)



# Monitoring Amazon IVS Low-Latency Streaming

You can monitor Amazon Interactive Video Service (IVS) resources using Amazon CloudWatch. CloudWatch collects and processes raw data from Amazon IVS into readable, near real-time metrics. These statistics are kept for 15 months, so you can gain a historical perspective on how your web application or service performs. You can set alarms for certain thresholds and send notifications or take actions when those thresholds are met. For details, see the [CloudWatch User Guide](#).

The timestamp on a metric represents the start of the period during which metric data is accumulated. For example, suppose you get a per-minute `LiveDeliveredTime` metric sum of 300 seconds at 01:02:00. This would mean that 5 minutes' worth of video was served to viewers during the 1-minute period from 01:02:00 to 01:02:59.

For metrics designated as high resolution, the first data point appears several seconds after stream start. We recommend you specify a 5-second period when making the metric requests. (See [Resolution](#) in the Amazon CloudWatch User Guide.) For other metrics, data is emitted within 1 minute of the timestamp to which it refers.

The high-resolution metrics are rolled up over time. Resolution effectively decreases as the metrics age. Here is the schedule:

- 1-second metrics are available for 3 hours.
- 60-second metrics are available for 15 days.
- 5-minute metrics are available for 63 days.
- 1-hour metrics are available for 455 days (15 months).

For current information on data retention, search for "retention period" in [Amazon CloudWatch FAQs](#).

## Prerequisites

- You must have an AWS account with sufficient IAM permissions to interface with the Stream Health APIs and CloudWatch metrics. For specific steps, see [Getting Started with IVS Low-Latency Streaming](#).



- You must create a channel and start a stream. Relevant information is in the [IVS Low-Latency Streaming User Guide](#):
  - For instructions on creating a channel, see [Create a Channel](#) in *Getting Started with IVS Low-Latency Streaming*.
  - For instructions on starting a stream, see [Set Up Streaming Software](#) in *Getting Started with IVS Low-Latency Streaming*.
  - For encoder-configuration details, see [Amazon IVS Streaming Configuration](#).

## Access Stream Session Data

Using the `listStreamSessions` endpoint, you can access a list of streams that a channel has had for up to 60 days. This list may include a live stream session (denoted by an empty `endTime`).

You can get the session data for a specific stream through the `getStreamSession` endpoint. If you do not specify the `streamId` parameter, the endpoint returns the latest session. In addition, you can periodically call the endpoint to get your stream's latest events (up to the most recent 500).

## Console Instructions

1. Open the [Amazon IVS console](#).  
(You also can access the Amazon IVS console through the [AWS Management Console](#).)
2. On the navigation pane, choose **Channels**. (If the nav pane is collapsed, first open it by choosing the hamburger icon.)
3. Choose the channel to go to its details page.
4. Scroll down the page until you see the **Stream sessions** section.
5. Select the Stream ID of the session you want to access to view its session details, including charts for the Amazon CloudWatch high-resolution metrics.

Alternatively, if one or more channels are already live:

1. Open the [Amazon IVS console](#).
2. On the navigation pane, choose **Live channels**. (If the nav pane is collapsed, first open it by choosing the hamburger icon.)
3. Select a live channel from the list to access its session details inside a split view.

## AWS SDK Instructions

Accessing stream-session data with the AWS SDK is an advanced option and requires that you first download and configure the SDK on your application. Below are instructions for the AWS SDK using JavaScript.

**Prerequisite:** To use the code sample below, you need to load the AWS JavaScript SDK into your application. For details, see [Getting started with the AWS SDK for JavaScript](#).

```
// This first call lists up to 50 stream sessions for a given channel.
const AWS = require("aws-sdk");
const REGION = 'us-west-2';
let channelArn = USE_YOUR_CHANNEL_ARN_HERE;

AWS.config.getCredentials(function(err) {
  if (err) console.log(err.stack);
  // credentials not loaded
  else {
    console.log("Access key:", AWS.config.credentials.accessKeyId);
  }
});

AWS.config.update({region: REGION});
var ivs = new AWS.IVS();

// List Stream Sessions
async function listSessions(arn) {
  const result = await ivs.listStreamSessions({"channelArn": arn}).promise();
  console.log(result.streamSessions);
}
listSessions(channelArn);

// Get Stream Session
async function getSession(arn, id) {
  const result = await ivs.getStreamSession({"channelArn": arn, "streamId":
id}).promise();
  console.log(result);

  // This function polls every 3 seconds and prints the latest IVS stream events.
  setInterval(function(){
    console.log(result.streamSession.truncatedEvents);
  }, 3000);
}
```

```
getSession(channelArn);
```

## CLI Instructions

Accessing stream-session data with the AWS CLI is an advanced option and requires that you first download and configure the CLI on your machine. For details, see the [AWS Command Line Interface User Guide](#).

### 1. List streams sessions:

```
aws ivs list-stream-sessions --channel-arn <arn>
```

### 2. Get stream session data for a specific stream using its streamId:

```
aws ivs get-stream-session --channel-arn <arn> --stream-id <streamId>
```

Here is a sample response to the `get-stream-session` call:

```
{
  "streamSession": {
    "startTime": "2021-10-22T00:03:57+00:00",
    "streamId": "st-1FQzeLONMT9XTKI431eLSol",
    "truncatedEvents": [
      {
        "eventTime": "2021-10-22T00:09:30+00:00",
        "name": "Session Ended",
        "type": "IVS Stream State Change"
      },
      {
        "eventTime": "2021-10-22T00:09:30+00:00",
        "name": "Stream End",
        "type": "IVS Stream State Change"
      },
      {
        "eventTime": "2021-10-22T00:03:57+00:00",
        "name": "Stream Start",
        "type": "IVS Stream State Change"
      },
      {
        "eventTime": "2021-10-22T00:03:50+00:00",
        "name": "Session Created",
```

```

        "type": "IVS Stream State Change"
    }
],
"endTime": "2021-10-22T00:09:31+00:00",
"ingestConfiguration": {
    "audio": {
        "channels": 2,
        "codec": "mp4a.40.2",
        "sampleRate": 48000,
        "targetBitrate": 160000
    },
    "video": {
        "avcLevel": "4.0",
        "avcProfile": "Baseline",
        "codec": "avc1.42C028",
        "encoder": "obs-output module (libobs version 27.0.1)",
        "targetBitrate": 3500000,
        "targetFramerate": 30,
        "videoHeight": 1080,
        "videoWidth": 1920
    }
},
"channel": {
    "name": "",
    "ingestEndpoint": "3f234d592b38.global-contribute.live-video.net",
    "authorized": false,
    "latencyMode": "LOW",
    "recordingConfigurationArn": "",
    "type": "STANDARD",
    "playbackUrl": "https://3f234d592b38.us-west-2.playback.live-video.net/api/video/v1/us-west-2.991729659840.channel.dY7LsluQX1gV.m3u8",
    "arn": "arn:aws:ivs:us-west-2:991729659840:channel/dY7LsluQX1gV"
}
}
}

```

## Filter Streams by Health

To easily find which streams are experiencing issues, you can use `listStreams` to filter live streams by “health.”

## Console Instructions

1. Open the [Amazon IVS console](#).

(You also can access the Amazon IVS console through the [AWS Management Console](#).)

2. On the navigation pane, choose **Live channels**. (If the nav pane is collapsed, first open it by choosing the hamburger icon.)
3. Select the search field for **Filter by health**.
4. In the drop-down list, select filtering by **Health = STARVING**.

After filtering, you can go to a channel's details page and select the channel's live-stream session, to access input-configuration details and stream events.

## CLI Instructions

Using the AWS CLI is an advanced option and requires that you first download and configure the CLI on your machine. For details, see the [AWS Command Line Interface User Guide](#).

To filter streams by health (e.g. STARVING):

```
aws ivs list-streams --filter-by health=STARVING
```

## CloudWatch Health Dimension for ConcurrentStreams

You can filter ConcurrentStreams by a specific Health. See [CloudWatch Metrics: IVS Low-Latency Streaming](#).

## Access CloudWatch Metrics

Amazon CloudWatch collects and processes raw data from Amazon IVS into readable, near-real-time metrics. These statistics are kept for 15 months, so you can gain a historical perspective on how your web application or service performs. You can set alarms for certain thresholds and send notifications or take actions when those thresholds are met. For details, see the [CloudWatch User Guide](#).

Note that CloudWatch metrics are rolled up over time. Resolution effectively decreases as the metrics age. Here is the schedule:

- 1-second metrics are available for 3 hours.
- 60-second metrics are available for 15 days.
- 5-minute metrics are available for 63 days.
- 1-hour metrics are available for 455 days (15 months).

When you call `getMetricData` you can specify a period of 1, 5 (recommended), 10, 30 or any multiple of 60 seconds for high-resolution metrics.

## CloudWatch Console Instructions

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the side navigation, expand the **Metrics** dropdown, then select **All metrics**.
3. On the **Browse** tab, using the unlabeled dropdown at the left, select your “home” region, where your channel(s) was(were) created. For more on regions, see [Global Solution, Regional Control](#). For a list of supported regions, see the [Amazon IVS page](#) in the *AWS General Reference*.
4. At the bottom of the **Browse** tab, select the **IVS** namespace.
5. Do one of the following:
  - a. In the search bar, enter your resource ID (part of the ARN, `arn:::ivs:channel/<resource id>`).  
  
Then select **IVS > By Channel**.
  - b. If **IVS** appears as a selectable service under **AWS Namespaces**, select it. It will be listed if you use Amazon IVS and it is sending metrics to Amazon CloudWatch. (If **IVS** is not listed, you do not have any Amazon IVS metrics.)  
  
Then choose a *dimension* grouping as desired; available dimensions are listed in [CloudWatch Metrics](#) below.
6. Choose metrics to add to the graph. Available metrics are listed in [CloudWatch Metrics](#) below.

You also can access your stream session’s CloudWatch chart from the stream session’s details page, by selecting the **View in CloudWatch** button.

## CLI Instructions

You also can access the metrics using the AWS CLI. This requires that you first download and configure the CLI on your machine. For details, see the [AWS Command Line Interface User Guide](#).

Then, to access Amazon IVS low-latency streaming metrics using the AWS CLI:

- At a command prompt, run:

```
aws cloudwatch list-metrics --namespace AWS/IVS
```

For more information, see [Using Amazon CloudWatch Metrics](#) in the *Amazon CloudWatch User Guide*.

## CloudWatch Metrics: IVS Low-Latency Streaming

Amazon IVS provides the following metrics in the **AWS/IVS** namespace.

Metric	Dimension	Description
ConcurrentViews	None	<p>A count of concurrent views across all your live channels. A <i>view</i> is a unique viewing session which is actively downloading or playing video. (For a more detailed definition, see the <a href="#">Glossary</a>.) If channels are live but in aggregate have no views, the value of this metric is 0. If no channels are live, the metric has no data points.</p> <p>Unit: Count</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of concurrent views over the configured interval.</p>
ConcurrentViews	Channel	Filters <code>ConcurrentViews</code> by channel ARN. If a channel is live but has no views, the value

Metric	Dimension	Description
		<p>of this metric is 0. If a channel is not live, the metric has no data points.</p> <p>This metric provides data for a channel, not a stream. To see concurrent views for a particular streaming session on a given channel, evaluate the <code>ConcurrentViews</code> metric for that channel between the start and end times of the streaming session.</p> <p>Unit: Count</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of concurrent views over the configured interval.</p>
ConcurrentStreams	None	<p>A count of your channels which are streaming live. If no channels are live, this metric has no data points.</p> <p>Unit: Count</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of concurrent streams over the configured interval.</p>



Metric	Dimension	Description
ConcurrentStreams	Health	<p>Filters ConcurrentStreams by channel health. If no channels are live, this metric has no data points.</p> <p>Unit: Count</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of concurrent streams for a specific Health over the configured interval.</p>
IngestAudioBitrate	Channel	<p><b>(High-resolution metric)</b> The amount of audio data Amazon IVS receives when you stream. A higher bitrate takes up more of your available internet bandwidth.</p> <p>Unit: Bits/second</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of ingest audio bitrates over the configured interval</p>
IngestFramerate	Channel	<p><b>(High-resolution metric)</b> How often video frames are received by Amazon IVS when you stream.</p> <p>Unit: Count/second</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of ingest framerates over the configured interval</p>

Metric	Dimension	Description
IngestVideoBitrate	Channel	<p><b>(High-resolution metric)</b> The amount of video data Amazon IVS receives when you stream. A higher bitrate takes up more of your available internet bandwidth. Higher bitrate can improve video quality, but only up to a certain point.</p> <p>Unit: Bits/second</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of ingest video bitrates over the configured interval</p>
KeyframeInterval	Channel	<p><b>(High-resolution metric)</b> The point in the video stream where the entire frame is sent instead of just the differences from the previous frame.</p> <p>Unit: Seconds</p> <p>Valid statistics: Average, Maximum, Minimum — Average number, largest number, or smallest number (respectively) of keyframe intervals over the configured interval</p>
LiveDeliveredTime	None	<p>Total real-time duration of video served to all viewers.</p> <p>Unit: Seconds</p> <p>Valid statistic: Sum</p>

Metric	Dimension	Description
LiveDeliveredTime	Channel	Filters LiveDeliveredTime by channel. Channel values are the channel's resource-id , which is the last part of an <a href="#">ARN</a> .  Unit: Seconds  Valid statistic: Sum
LiveDeliveredTime	Channel, ViewerCountryCode	Filters LiveDeliveredTime by channel and viewer's country code. Channel values are the channel's resource-id , which is the last part of an <a href="#">ARN</a> . Country values are two-character ISO 3166-1 country codes. This allows you to answer the question: where are my viewers watching from? If the viewer's country cannot be determined, it is shown as UNKNOWN.  Unit: Seconds  Valid statistic: Sum
LiveInputTime	None	Real-time duration of video stream.  Unit: Seconds  Valid statistic: Sum
LiveInputTime	Channel	Filters LiveInputTime by channel. Channel values are the channel's resource-id , which is the last part of an <a href="#">ARN</a> .  Unit: Seconds  Valid statistic: Sum

Metric	Dimension	Description
RecordedTime	None	Real-time duration of recorded video.  Unit: Seconds  Valid statistic: Sum
RecordedTime	Channel	Filters RecordedTime by channel. Channel values are the channel's resource-id , which is the last part of an <a href="#">ARN</a> .  Unit: Seconds  Valid statistic: Sum

# IVS Broadcast SDK (Low-Latency Streaming)

The Amazon Interactive Video Services (IVS) Low-Latency Streaming broadcast SDK is for developers who are building applications with Amazon IVS. This SDK is designed to leverage the Amazon IVS architecture and will see continual improvement and new features, alongside Amazon IVS. As a native broadcast SDK, it is designed to minimize the performance impact on your application and on the devices with which your users access your application.

Your application can leverage the key features of the Amazon IVS broadcast SDK:

- **High quality streaming** — The broadcast SDK supports high quality streaming. Capture video from your camera and encode it at up to 1080p quality for a high quality viewing experience.
- **Automatic Bitrate Adjustments** — Smartphone users are mobile, so their network conditions can change throughout the course of a broadcast. The Amazon IVS broadcast SDK automatically adjusts the video bitrate to accommodate changing network conditions.
- **Portrait and Landscape Support** — No matter how your users hold their devices, the image appears right-side up and properly scaled. The broadcast SDK supports both portrait and landscape canvas sizes. It automatically manages the aspect ratio when the users rotate their device away from the configured orientation.
- **Secure Streaming** — Your user's broadcasts are encrypted using TLS, so they can keep their streams secure.
- **External Audio Devices** — The Amazon IVS broadcast SDK supports audio jack, USB, and Bluetooth SCO external microphones.

## Platform Requirements

### Native Platforms

Platform	Supported Versions
Android	5.0 (Lollipop) and later
iOS	13+

Platform	Supported Versions
	If broadcasting is essential to your application, specify Metal as a requirement for downloading your app from the Apple App Store, using <a href="#">UIRequiredDeviceCapabilities</a> .

IVS supports a minimum of 4 major iOS versions and 6 major Android versions. Our current version support may extend beyond these minimums. Customers will be notified via SDK release notes at least 3 months in advance of a major version no longer being supported.

## Desktop Browsers

Browser	Supported Platforms	Supported Versions
Chrome	Windows, macOS	Two major versions (current and most recent prior version)
Firefox	Windows, macOS	Two major versions (current and most recent prior version)
Edge	Windows 8.1 and later	Two major versions (current and most recent prior version)  Excludes Edge Legacy
Safari	macOS	Two major versions (current and most recent prior version)

## Mobile Browsers

Browser	Supported Versions
Chrome for iOS, Safari for iOS	Two major versions (current and most recent prior version)

Browser	Supported Versions
Chrome for iPadOS, Safari for iPadOS	Two major versions (current and most recent prior version)
Chrome for Android	Two major versions (current and most recent prior version)

## Webviews

The Web broadcast SDK does not provide support for webviews or weblike environments (TVs, consoles, etc). For mobile implementations, see the Low-Latency Streaming Broadcast SDK Guide for [Android](#) and for [iOS](#).

## Required Device Access

The broadcast SDK requires access to the device's cameras and microphones, both those built into the device and those connected through Bluetooth, USB, or audio jack.

## Support

If you encounter a broadcast error or other issue with your stream, determine the unique playback session identifier via the broadcast API.

For this Amazon IVS Broadcast SDK:	Use this:
Android	<code>getSessionId</code> function on Broadcast Session
iOS	<code>sessionId</code> property of <code>IVSBroadcastSession</code>
Web	<code>getSessionId</code> function

Share this broadcast session identifier with AWS support. With it, they can get information to help troubleshoot your issue.

**Note:** The broadcast SDK is continually improved. See [Amazon IVS Release Notes](#) for available versions and fixed issues. If appropriate, before contacting support, update your version of the broadcast SDK and see if that resolves your issue.

## Versioning

The Amazon IVS broadcast SDKs use [semantic versioning](#).

For this discussion, suppose:

- The latest release is 4.1.3.
- The latest release of the prior major version is 3.2.4.
- The latest release of version 1.x is 1.5.6.

Backward-compatible new features are added as minor releases of the latest version. In this case, the next set of new features will be added as version 4.2.0.

Backward-compatible, minor bug fixes are added as patch releases of the latest version. Here, the next set of minor bug fixes will be added as version 4.1.4.

Backward-compatible, major bug fixes are handled differently; these are added to several versions:

- Patch release of the latest version. Here, this is version 4.1.4.
- Patch release of the prior minor version. Here, this is version 3.2.5.
- Patch release of the latest version 1.x release. Here, this is version 1.5.7.

Major bug fixes are defined by the Amazon IVS product team. Typical examples are critical security updates and selected other fixes necessary for customers.

**Note:** In the examples above, released versions increment without skipping any numbers (e.g., from 4.1.3 to 4.1.4). In reality, one or more patch numbers may remain internal and not be released, so the released version could increment from 4.1.3 to, say, 4.1.6.

## IVS Broadcast SDK: Web Guide (Low-Latency Streaming)

The IVS Low-Latency Streaming Web Broadcast SDK gives developers the tools to build interactive, real-time experiences on the web.

**Latest version of Web broadcast SDK:** 1.13.0 ([Release Notes](#))



**Reference documentation:** For information on the most important methods available in the Amazon IVS Web Broadcast SDK, see <https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference>. Make sure the most current version of the SDK is selected.

**Sample code:** The samples below are a good place to get started quickly with the SDK:

- [Single broadcast to an IVS channel \(HTML and JavaScript\)](#)
- [Single broadcast with screen share to an IVS channel \(React Source Code\)](#)

**Platform requirements:** See [Amazon IVS Broadcast SDK](#) for a list of supported platforms.

## Getting Started

### Install the Library

Note that the `IVSBroadcastClient` leverages [reflect-metadata](#), which extends the global `Reflect` object. Although this should not create any conflicts, there may be rare instances where this could cause unwanted behavior.

### Using a Script Tag

The Web broadcast SDK is distributed as a JavaScript library and can be retrieved at <https://web-broadcast.live-video.net/1.13.0/amazon-ivs-web-broadcast.js>.

When loaded via `<script>` tag, the library exposes a global variable in the window scope named `IVSBroadcastClient`.

### Using npm

To install the npm package:

```
npm install amazon-ivs-web-broadcast
```

You can now access the `IVSBroadcastClient` object and pull in other modules and consts such as `Errors`, `BASIC_LANDSCAPE`:

```
import IVSBroadcastClient, {
  Errors,
  BASIC_LANDSCAPE
} from 'amazon-ivs-web-broadcast';
```

## Samples

To get started quickly, see the examples below:

- [Single broadcast to an IVS channel \(HTML and JavaScript\)](#)
- [Single broadcast with screen share to an IVS channel \(React Source Code\)](#)

## Create an Instance of the AmazonIVSBroadcastClient

To use the library, you must create an instance of the client. You can do that by calling the `create` method on `IVSBroadcastClient` with the `streamConfig` parameter (specifying constraints of your broadcast like resolution and framerate). You can specify the ingest endpoint when creating the client or you can set this when you start a stream.

The ingest endpoint can be found in the AWS Console or returned by the `CreateChannel` endpoint (e.g., `UNIQUE_ID.global-contribute.live-video.net`).

```
const client = IVSBroadcastClient.create({
  // Enter the desired stream configuration
  streamConfig: IVSBroadcastClient.BASIC_LANDSCAPE,
  // Enter the ingest endpoint from the AWS console or CreateChannel API
  ingestEndpoint: 'UNIQUE_ID.global-contribute.live-video.net',
});
```

These are the common supported stream configurations. Presets are BASIC up to 480p and 1.5 Mbps bitrate, BASIC Full HD up to 1080p and 3.5 Mbps bitrate, and STANDARD (or ADVANCED) up to 1080p and 8.5 Mbps bitrate. You can customize the bitrate, frame rate, and resolution if desired. For more information, see [BroadcastClientConfig](#).

```
IVSBroadcastClient.BASIC_LANDSCAPE;
IVSBroadcastClient.BASIC_FULL_HD_LANDSCAPE;
IVSBroadcastClient.STANDARD_LANDSCAPE;
IVSBroadcastClient.BASIC_PORTRAIT;
IVSBroadcastClient.BASIC_FULL_HD_PORTRAIT;
IVSBroadcastClient.STANDARD_PORTRAIT;
```

You can import these individually if using the npm package.

**Note:** Make sure that your client-side configuration aligns with the back-end channel type. For instance, if the channel type is STANDARD, `streamConfig` should be set to one of the

`IVSBroadcastClient.STANDARD_*` values. If channel type is `ADVANCED`, you'll need to set the configuration manually as shown below (using `ADVANCED_HD` as an example):

```
const client = IVSBroadcastClient.create({
  // Enter the custom stream configuration
  streamConfig: {
    maxResolution: {
      width: 1080,
      height: 1920,
    },
    maxFramerate: 30,
    /**
     * maxBitrate is measured in kbps
     */
    maxBitrate: 3500,
  },
  // Other configuration . . .
});
```

## Request Permissions

Your app must request permission to access the user's camera and microphone, and it must be served using HTTPS. (This is not specific to Amazon IVS; it is required for any website that needs access to cameras and microphones.)

Here's an example function showing how you can request and capture permissions for both audio and video devices:

```
async function handlePermissions() {
  let permissions = {
    audio: false,
    video: false,
  };
  try {
    const stream = await navigator.mediaDevices.getUserMedia({ video: true, audio:
true });
    for (const track of stream.getTracks()) {
      track.stop();
    }
    permissions = { video: true, audio: true };
  } catch (err) {
    permissions = { video: false, audio: false };
  }
}
```

```
        console.error(err.message);
    }
    // If we still don't have permissions after requesting them display the error
    message
    if (!permissions.video) {
        console.error('Failed to get video permissions.');
```

For additional information, see the [Permissions API](#) and [MediaDevices.getUserMedia\(\)](#).

## Set Up a Stream Preview

To preview what will be broadcast, provide the SDK with a `<canvas>` element.

```
// where #preview is an existing <canvas> DOM element on your page
const previewEl = document.getElementById('preview');
client.attachPreview(previewEl);
```

## List Available Devices

To see what devices are available to capture, query the browser's [MediaDevices.enumerateDevices\(\)](#) method:

```
const devices = await navigator.mediaDevices.enumerateDevices();
window.videoDevices = devices.filter((d) => d.kind === 'videoinput');
window.audioDevices = devices.filter((d) => d.kind === 'audioinput');
```

## Retrieve a MediaStream from a Device

After acquiring the list of available devices, you can retrieve a stream from any number of devices. For example, you can use the `getUserMedia()` method to retrieve a stream from a camera.

If you'd like to specify which device to capture the stream from, you can explicitly set the `deviceId` in the `audio` or `video` section of the media constraints. Alternately, you can omit the `deviceId` and have users select their devices from the browser prompt.

You also can specify an ideal camera resolution using the `width` and `height` constraints. (Read more about these constraints [here](#).) The SDK automatically applies width and height constraints

that correspond to your maximum broadcast resolution; however, it's a good idea to also apply these yourself to ensure that the source aspect ratio is not changed after you add the source to the SDK.

```
const streamConfig = IVSBroadcastClient.BASIC_LANDSCAPE;
...
window.cameraStream = await navigator.mediaDevices.getUserMedia({
  video: {
    deviceId: window.videoDevices[0].deviceId,
    width: {
      ideal: streamConfig.maxResolution.width,
    },
    height: {
      ideal: streamConfig.maxResolution.height,
    },
  },
});
window.microphoneStream = await navigator.mediaDevices.getUserMedia({
  audio: { deviceId: window.audioDevices[0].deviceId },
});
```

## Add Device to a Stream

After acquiring the stream, you may add devices to the layout by specifying a unique name (below, this is `camera1`) and composition position (for video). For example, by specifying your webcam device, you add your webcam video source to the broadcast stream.

When specifying the video-input device, you must specify the index, which represents the “layer” on which you want to broadcast. This is synonymous to image editing or CSS, where a z-index represents the ordering of layers to render. Optionally, you can provide a position, which defines the x/y coordinates (as well as the size) of the stream source.

For details on parameters, see [VideoComposition](#).

```
client.addVideoInputDevice(window.cameraStream, 'camera1', { index: 0 }); // only
  'index' is required for the position parameter
client.addAudioInputDevice(window.microphoneStream, 'mic1');
```

## Start a Broadcast

To start a broadcast, provide the stream key for your Amazon IVS channel:

```
client
  .startBroadcast(streamKey)
  .then((result) => {
    console.log('I am successfully broadcasting!');
  })
  .catch((error) => {
    console.error('Something drastically failed while broadcasting!', error);
  });
```

## Stop a Broadcast

```
client.stopBroadcast();
```

## Swap Video Positions

The client supports swapping the composition positions of video devices:

```
client.exchangeVideoDevicePositions('camera1', 'camera2');
```

## Mute Audio

To mute audio, either remove the audio device using `removeAudioInputDevice` or set the `enabled` property on the audio track:

```
let audioStream = client.getAudioInputDevice(AUDIO_DEVICE_NAME);
audioStream.getAudioTracks()[0].enabled = false;
```

Where `AUDIO_DEVICE_NAME` is the name given to the original audio device during the `addAudioInputDevice()` call.

To unmute:

```
let audioStream = client.getAudioInputDevice(AUDIO_DEVICE_NAME);
audioStream.getAudioTracks()[0].enabled = true;
```

## Hide Video

To hide video, either remove the video device using `removeVideoInputDevice` or set the `enabled` property on the video track:

```
let videoStream = client.getVideoInputDevice(VIDEO_DEVICE_NAME).source;  
videoStream.getVideoTracks()[0].enabled = false;
```

Where `VIDEO_DEVICE_NAME` is the name given to the video device during the original `addVideoInputDevice()` call.

To unhide:

```
let videoStream = client.getVideoInputDevice(VIDEO_DEVICE_NAME).source;  
videoStream.getVideoTracks()[0].enabled = true;
```

## Known Issues and Workarounds

- The web broadcast SDK requires port 4443 to be open. VPNs and firewalls can block port 4443 and prevent you from streaming.

**Workaround:** Disable VPNs and/or configure firewalls to ensure that port 4443 is not blocked.

- Switching from landscape to portrait mode is buggy.

**Workaround:** None.

- The resolution reported in the HLS manifest is incorrect. It is set as the initially received resolution, which usually is much lower than what is possible and does not reflect any upscaling that happens during the duration of the webRTC connection.

**Workaround:** None.

- Subsequent client instances created after the initial page is loaded may not respond to `maxFramerate` settings that are different from the first client instance.

**Workaround:** Set `StreamConfig` only once, through the `IVSBroadcastClient.create` function when the first client instance is created.

- On iOS, capturing multiple video device sources is not supported by WebKit.

**Workaround:** Follow [this issue](#) to track development progress.

- On iOS, calling `getUserMedia()` once you already have a video source will stop any other video source retrieved using `getUserMedia()`.

**Workaround:** None.

- WebRTC dynamically chooses the best bitrate and resolution for the resources that are available. Your stream will not be high quality if your hardware or network cannot support it. The quality of your stream may change during the broadcast as more or fewer resources are available.

**Workaround:** Provide at least 200 kbps upload.

- If Auto-Record to Amazon S3 is enabled for a channel and the Web Broadcast SDK is used, recording to the same S3 prefix may not work, as the Web Broadcast SDK dynamically changes bitrates and qualities.

**Workaround:** None.

- When using Next.js, an `Uncaught ReferenceError: self is not defined` error may be encountered, depending on how the SDK is imported.

**Workaround:** [Dynamically import the library](#) when using Next.js.

- You may be unable to import the module using a script tag of type `module`; i.e., `<script type="module" src="...">`.

**Workaround:** The library does not have an ES6 build. Remove the `type="module"` from the script tag.

- Viewers may experience green artifacts, when watching streams from broadcasters who are using Safari on Intel-based Mac devices.

**Workaround:** Redirect broadcasters on Intel Mac devices to broadcast using Chrome.

## Safari Limitations

- Denying a permissions prompt requires resetting the permission in Safari website settings at the OS level.



- Safari does not natively detect all devices as effectively as Firefox or Chrome. For example, OBS Virtual Camera does not get detected.

## Firefox Limitations

- System permissions need to be enabled for Firefox to screen share. After enabling them, the user must restart Firefox for it to work correctly; otherwise, if permissions are perceived as blocked, the browser will throw a [NotFoundError](#) exception.
- The `getCapabilities` method is missing. This means users cannot get the media track's resolution or aspect ratio. See this [bugzilla thread](#).
- Several `AudioContext` properties are missing; e.g., latency and channel count. This could pose a problem for advanced users who want to manipulate the audio tracks.
- Camera feeds from `getUserMedia` are restricted to a 4:3 aspect ratio on MacOS. See [bugzilla thread 1](#) and [bugzilla thread 2](#).
- Audio capture is not supported with `getDisplayMedia`. See this [bugzilla thread](#).
- Framerate in screen capture is suboptimal (approximately 15fps?). See this [bugzilla thread](#).

## IVS Broadcast SDK: Android Guide (Low-Latency Streaming)

The IVS Low-Latency Streaming Android Broadcast SDK provides the interfaces required to broadcast to IVS on Android.

The `com.amazonaws.ivs.broadcast` package implements the interface described in this document. The following operations are supported:

- Set up (initialize) a broadcast session.
- Manage broadcasting.
- Attach and detach input devices.
- Manage a composition session.
- Receive events.
- Receive errors.

## Latest version of Android broadcast SDK: 1.19.0 ([Release Notes](#))

**Reference documentation:** For information on the most important methods available in the Amazon IVS Android broadcast SDK, see the reference documentation at <https://aws.github.io/amazon-ivs-broadcast-docs/1.19.0/android/>.

**Sample code:** See the Android sample repository on GitHub: <https://github.com/aws-samples/amazon-ivs-broadcast-android-sample>.

**Platform requirements:** Android 5.0 (Lollipop) or greater

## Getting Started

### Install the Library

To add the Amazon IVS Android broadcast library to your Android development environment, add the library to your module's `build.gradle` file, as shown here (for the latest version of the Amazon IVS broadcast SDK):

```
repositories {
    mavenCentral()
}
dependencies {
    implementation 'com.amazonaws:ivs-broadcast:1.19.0'
}
```

Alternately, to install the SDK manually, download the latest version from this location:

<https://search.maven.org/artifact/com.amazonaws/ivs-broadcast>

### Create the Event Listener

Setting up an event listener allows you to receive state updates, device-change notifications, errors, and session-audio information.

```
BroadcastSession.Listener broadcastListener =
    new BroadcastSession.Listener() {
    @Override
    public void onStateChanged(@NonNull BroadcastSession.State state) {
        Log.d(TAG, "State=" + state);
    }
}
```

```
    }

    @Override
    public void onError(@NonNull BroadcastException exception) {
        Log.e(TAG, "Exception: " + exception);
    }
};
```

## Request Permissions

Your app must request permission to access the user's camera and mic. (This is not specific to Amazon IVS; it is required for any application that needs access to cameras and microphones.)

Here, we check whether the user has already granted permissions and, if not, ask for them:

```
final String[] requiredPermissions =
    { Manifest.permission.CAMERA, Manifest.permission.RECORD_AUDIO };

for (String permission : requiredPermissions) {
    if (ContextCompat.checkSelfPermission(this, permission)
        != PackageManager.PERMISSION_GRANTED) {
        // If any permissions are missing we want to just request them all.
        ActivityCompat.requestPermissions(this, requiredPermissions, 0x100);
        break;
    }
}
```

Here, we get the user's response:

```
@Override
public void onRequestPermissionsResult(int requestCode,
                                       @NonNull String[] permissions,
                                       @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode,
                                     permissions, grantResults);
    if (requestCode == 0x100) {
        for (int result : grantResults) {
            if (result == PackageManager.PERMISSION_DENIED) {
                return;
            }
        }
        setupBroadcastSession();
    }
}
```

```
}  
}
```

## Create the Broadcast Session

The broadcast interface is `com.amazonaws.ivs.broadcast.BroadcastSession`. Initialize it with a preset, as shown below. If there are any errors during initialization (such as a failure to configure a codec) your `BroadcastListener` will get an error message and `broadcastSession.isReady` will be `false`.

**Important:** All calls to the Amazon IVS Broadcast SDK for Android *must* be made on the thread on which the SDK is instantiated. A call from a different thread will cause the SDK to throw a fatal error and stop broadcasting.

```
// Create a broadcast-session instance and sign up to receive broadcast  
// events and errors.  
Context ctx = getApplicationContext();  
broadcastSession = new BroadcastSession(ctx,  
                                       broadcastListener,  
                                       Presets.Configuration.STANDARD_PORTRAIT,  
                                       Presets.Devices.FRONT_CAMERA(ctx));
```

Also see [Create the Broadcast Session \(Advanced Version\)](#).

## Set the ImagePreviewView for Preview

If you want to display a preview for an active camera device, add a preview `ImagePreviewView` for the device to your view hierarchy.

```
// awaitDeviceChanges will fire on the main thread after all pending devices  
// attachments have been completed  
broadcastSession.awaitDeviceChanges(() -> {  
    for(Device device: session.listAttachedDevices()) {  
        // Find the camera we attached earlier  
        if(device.getDescriptor().type == Device.Descriptor.DeviceType.CAMERA) {  
            LinearLayout previewHolder = findViewById(R.id.previewHolder);  
            ImagePreviewView preview = ((ImageDevice)device).getPreviewView();  
            preview.setLayoutParams(new LinearLayout.LayoutParams(  
                LinearLayout.LayoutParams.MATCH_PARENT,  
                LinearLayout.LayoutParams.MATCH_PARENT));  
            previewHolder.addView(preview);  
        }  
    }  
});
```

```
    }  
  }  
});
```

## Start a Broadcast

The hostname that you receive in the `ingestEndpoint` response field of the `GetChannel` endpoint needs to have `rtmps://` prepended and `/app` appended. The complete URL should be in this format: `rtmps://{ ingestEndpoint }/app`

```
broadcastSession.start(IVS_RTMP_URL, IVS_STREAMKEY);
```

The Android broadcast SDK supports only RTMPS ingest (not insecure RTMP ingest).

## Stop a Broadcast

```
broadcastSession.stop();
```

## Release the Broadcast Session

You *must call* the `broadcastSession.release()` method when the broadcast session is no longer in use, to free the resources used by the library.

```
@Override  
protected void onDestroy() {  
    super.onDestroy();  
    previewHolder.removeAllViews();  
    broadcastSession.release();  
}
```

## Advanced Use Cases

Here we present some advanced use cases. Start with the basic setup above and continue here.

### Create a Broadcast Configuration

Here we create a custom configuration with two mixer slots that allow us to bind two video sources to the mixer. One (`custom`) is full screen and laid out behind the other (`camera`), which is smaller

and in the bottom-right corner. Note that for the custom slot we do not set a position, size, or aspect mode. Because we do not set these parameters, the slot will use the video settings for size and position.

```
BroadcastConfiguration config = BroadcastConfiguration.with($ -> {
    $.audio.setBitrate(128_000);
    $.video.setMaxBitrate(3_500_000);
    $.video.setMinBitrate(500_000);
    $.video.setInitialBitrate(1_500_000);
    $.video.setSize(1280, 720);
    $.mixer.slots = new BroadcastConfiguration.Mixer.Slot[] {
        BroadcastConfiguration.Mixer.Slot.with(slot -> {
            // Do not automatically bind to a source
            slot.setPreferredAudioInput(
                Device.Descriptor.DeviceType.UNKNOWN);
            // Bind to user image if unbound
            slot.setPreferredVideoInput(
                Device.Descriptor.DeviceType.USER_IMAGE);
            slot.setName("custom");
            return slot;
        }),
        BroadcastConfiguration.Mixer.Slot.with(slot -> {
            slot.setIndex(1);
            slot.setAspect(BroadcastConfiguration.AspectMode.FILL);
            slot.setSize(300, 300);
            slot.setPosition($.video.getSize().x - 350,
                $.video.getSize().y - 350);
            slot.setName("camera");
            return slot;
        })
    };
    return $;
});
```

## Create the Broadcast Session (Advanced Version)

Create a `BroadcastSession` as you did in the [basic example](#), but provide your custom configuration here. Also provide `null` for the device array, as we will add those manually.

```
// Create a broadcast-session instance and sign up to receive broadcast
// events and errors.
Context ctx = getApplicationContext();
```

```

broadcastSession = new BroadcastSession(ctx,
    broadcastListener,
    config, // The configuration we created above
    null); // We'll manually attach devices after

```

## Iterate and Attach a Camera Device

Here we iterate through input devices that the SDK has detected. On Android 7 (Nougat) this will only return default microphone devices, because the Amazon IVS Broadcast SDK does not support selecting non-default devices on this version of Android.

Once we find a device that we want to use, we call `attachDevice` to attach it. A lambda function is called on the main thread when attaching the input device has completed. In case of failure, you will receive an error in the Listener.

```

for(Device.Descriptor desc:
    BroadcastSession.listAvailableDevices(getApplicationContext())) {
    if(desc.type == Device.Descriptor.DeviceType.CAMERA &&
        desc.position == Device.Descriptor.Position.FRONT) {
        session.attachDevice(desc, device -> {
            LinearLayout previewHolder = findViewById(R.id.previewHolder);
            ImagePreviewView preview = ((ImageDevice)device).getPreviewView();
            preview.setLayoutParams(new LinearLayout.LayoutParams(
                LinearLayout.LayoutParams.MATCH_PARENT,
                LinearLayout.LayoutParams.MATCH_PARENT));
            previewHolder.addView(preview);
            // Bind the camera to the mixer slot we created above.
            session.getMixer().bind(device, "camera");
        });
        break;
    }
}

```

## Swap Cameras

```

// This assumes you've kept a reference called "currentCamera" that points to
// a front facing camera
for(Device device: BroadcastSession.listAvailableDevices()) {
    if(device.type == Device.Descriptor.DeviceType.CAMERA &&
        Device.position != currentCamera.position) {
        // Remove the preview view for the old device.
        // setImagePreviewTextureView is an example function
    }
}

```

```
// that handles your view hierarchy.
setImagePreviewView(null);
session.exchangeDevices(currentCamera, device, camera -> {
    // Set the preview view for the new device.
    setImagePreviewView(camera.getPreviewView());
    currentCamera = camera;
});
break;
}
}
```

## Create an Input Surface

To input sound or image data that your app generates, use `createImageInputSource` or `createAudioInputSource`. Both these methods create and attach virtual devices that can be bound to the mixer like any other device.

The `SurfaceSource` returned by `createImageInputSource` has a `getInputSurface` method, which will give you a `Surface` that you can use with the Camera2 API, OpenGL, or Vulkan, or anything else that can write to a `Surface`.

The `AudioDevice` returned by `createAudioInputSource` can receive Linear PCM data generated by `AudioRecorder` or other means.

```
SurfaceSource source = session.createImageInputSource();
Surface surface = source.getInputSurface();
session.getMixer().bind(source, "custom");
```

## Detach a Device

If you want to detach and not replace a device, detach it with `Device` or `Device.Descriptor`.

```
session.detachDevice(currentCamera);
```

## Screen and System Audio Capture

The Amazon IVS Broadcast SDK for Android includes some helpers that simplify capturing the device's screen (Android 5 and higher) and system audio (Android 10 and higher). If you want to manage these manually, you can create a custom image-input source and a custom audio-input source.



To create a screen and system audio-capture session, you must first create a permission-request intent:

```
public void startScreenCapture() {
    MediaProjectionManager manager =
        (MediaProjectionManager) getApplicationContext()
            .getSystemService(Context.MEDIA_PROJECTION_SERVICE);
    if(manager != null) {
        Intent intent = manager.createScreenCaptureIntent();
        startActivityIfNeeded(intent, SCREEN_CAPTURE_REQUEST_ID);
    }
}
```

To use this feature, you must provide a class that extends `com.amazonaws.ivs.broadcast.SystemCaptureService`. You do not have to override any of its methods, but the class needs to be there to avoid any potential collisions between services.

You also must add a couple of elements to your Android manifest:

```
<uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
<application ...>
    <service android:name=".ExampleSystemCaptureService"
        android:foregroundServiceType="mediaProjection"
        android:isolatedProcess="false" />
</application>
...
```

Your class that extends `SystemCaptureService` must be named in the `<service>` element. On Android 9 and later, the `foregroundServiceType` must be `mediaProjection`.

Once the permissions intent has returned, you may proceed with creating the screen and system audio-capture session. On Android 8 and later, you must provide a notification to be displayed in your user's Notification Panel. The Amazon IVS Broadcast SDK for Android provides the convenience method `createServiceNotificationBuilder`. Alternately, you may provide your own notification.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(requestCode != SCREEN_CAPTURE_REQUEST_ID
```

```
    || Activity.RESULT_OK != resultCode) {
        return;
    }
    Notification notification = null;
    if(Build.VERSION.SDK_INT >= 26) {
        Intent intent = new Intent(getApplicationContext(),
                                   NotificationActivity.class);

        notification = session
            .createServiceNotificationBuilder("example",
                                             "example channel", intent)
            .build();
    }
    session.createSystemCaptureSources(data,
                                       ExampleSystemCaptureService.class,
                                       Notification,
                                       devices -> {
// This step is optional if the mixer slots have been given preferred
// input device types SCREEN and SYSTEM_AUDIO
        for (Device device : devices) {
            session.getMixer().bind(device, "game");
        }
    });
}
```

## Get Recommended Broadcast Settings

To evaluate your user's connection before starting a broadcast, use the `recommendedVideoSettings` method to run a brief test. As the test runs, you will receive several recommendations, ordered from most to least recommended. In this version of the SDK, it is not possible to reconfigure the current `BroadcastSession`, so you will need to `release()` it and then create a new one with the recommended settings. You will continue to receive `BroadcastSessionTest.Results` until the `Result.status` is `SUCCESS` or `ERROR`. You can check progress with `Result.progress`.

Amazon IVS supports a maximum bitrate of 8.5 Mbps (for channels whose type is `STANDARD` or `ADVANCED`), so the `maximumBitrate` returned by this method never exceeds 8.5 Mbps. To account for small fluctuations in network performance, the recommended `initialBitrate` returned by this method is slightly less than the true bitrate measured in the test. (Using 100% of the available bandwidth usually is inadvisable.)

```
void runBroadcastTest() {
```

```
    this.test = session.recommendedVideoSettings(RTMP_ENDPOINT, RTMP_STREAMKEY,
        result -> {
            if (result.status == BroadcastSessionTest.Status.SUCCESS) {
                this.recommendation = result.recommendations[0];
            }
        });
}
```

## Using Bluetooth Microphones

To broadcast using Bluetooth microphone devices, you must start a Bluetooth SCO connection:

```
Bluetooth.startBluetoothSco(context);
// Now bluetooth microphones can be used
...
// Must also stop bluetooth SCO
Bluetooth.stopBluetoothSco(context);
```

## Known Issues and Workarounds

- Using an external microphone connected through Bluetooth can be unstable. When a Bluetooth device is connected or disconnected during a broadcasting session, microphone input may stop working until the device is explicitly detached and reattached.

**Workaround:** If you plan to use a Bluetooth headset, connect it before starting the broadcast and leave it connected throughout the broadcast.

- The broadcast SDK does not support access on external cameras connected via USB.

**Workaround:** Do not use external cameras connected via USB.

- Submitting audio data faster than realtime (using a custom audio source) results in audio drift.

**Workaround:** Do not submit audio data faster than realtime.

- Some Android 5 devices may stream a black image if the same `BroadcastSession` is used for multiple broadcasts.

**Workaround:** When stopping the `BroadcastSession`, release it and instantiate a new one.

- Android 5, 6, and 7 devices cannot receive the broadcast SDK's `onDeviceAdded` and `onDeviceRemoved` callbacks for microphones, because these Android versions allow only the system's default microphone.

**Workaround:** For these devices, the broadcast SDK uses the system's default microphone.

- When an `ImagePreviewView` is removed from a parent (e.g., `removeView()` is called at the parent), the `ImagePreviewView` is released immediately. The `ImagePreviewView` does not show any frames when it is added to another parent view.

**Workaround:** Request another preview using `getPreview`.

- Some Android video encoders cannot be configured with a video size less than 176x176. Configuring a smaller size causes an error and prevents streaming.

**Workaround:** Do not configure the video size to be less than 176x176.

## IVS Broadcast SDK: iOS Guide (Low-Latency Streaming)

The IVS Low-Latency Streaming iOS Broadcast SDK provides the interfaces required to broadcast to Amazon IVS on iOS.

The `AmazonIVSBroadcast` module implements the interface described in this document. The following operations are supported:

- Set up (initialize) a broadcast session.
- Manage broadcasting.
- Attach and detach input devices.
- Manage a composition session.
- Receive events.
- Receive errors.

**Latest version of iOS broadcast SDK:** 1.19.0 ([Release Notes](#))

**Reference documentation:** For information on the most important methods available in the Amazon IVS iOS broadcast SDK, see the reference documentation at <https://aws.github.io/amazon-ivs-broadcast-docs/1.19.0/ios/>.

**Sample code:** See the iOS sample repository on GitHub: <https://github.com/aws-samples/amazon-ivs-broadcast-ios-sample>.

**Platform requirements:** iOS 12 or greater

# Getting Started

## Install the Library

We recommend that you integrate the broadcast SDK via CocoaPods. (Alternatively, you can manually add the framework to your project.)

### Recommended: Integrate the Broadcast SDK (CocoaPods)

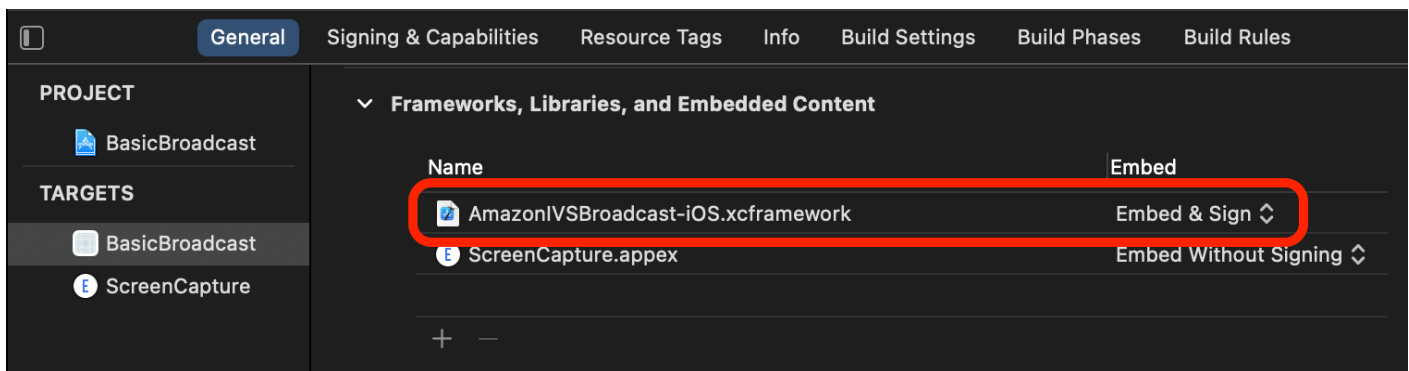
Releases are published via CocoaPods under the name AmazonIVSBroadcast. Add this dependency to your Podfile:

```
pod 'AmazonIVSBroadcast'
```

Run `pod install` and the SDK will be available in your `.xcworkspace`.

### Alternate Approach: Install the Framework Manually

1. Download the latest version from <https://broadcast.live-video.net/1.19.0/AmazonIVSBroadcast.xcframework.zip>.
2. Extract the contents of the archive. `AmazonIVSBroadcast.xcframework` contains the SDK for both device and simulator.
3. Embed `AmazonIVSBroadcast.xcframework` by dragging it into the **Frameworks, Libraries, and Embedded Content** section of the **General** tab for your application target.



## Implement IVSBroadcastSession.Delegate

Implement `IVSBroadcastSession.Delegate`, which allows you to receive state updates and device-change notifications:

```
extension ViewController : IVSBroadcastSession.Delegate {
    func broadcastSession(_ session: IVSBroadcastSession,
                        didChange state: IVSBroadcastSession.State) {
        print("IVSBroadcastSession did change state \(state)")
    }

    func broadcastSession(_ session: IVSBroadcastSession,
                        didEmitError error: Error) {
        print("IVSBroadcastSession did emit error \(error)")
    }
}
```

## Request Permissions

Your app must request permission to access the user's camera and mic. (This is not specific to Amazon IVS; it is required for any application that needs access to cameras and microphones.)

Here, we check whether the user has already granted permissions and, if not, we ask for them:

```
switch AVCaptureDevice.authorizationStatus(for: .video) {
case .authorized: // permission already granted.
case .notDetermined:
    AVCaptureDevice.requestAccess(for: .video) { granted in
        // permission granted based on granted bool.
    }
case .denied, .restricted: // permission denied.
@unknown default: // permissions unknown.
}
```

You need to do this for both `.video` and `.audio` media types, if you want access to cameras and microphones, respectively.

You also need to add entries for `NSCameraUsageDescription` and `NSMicrophoneUsageDescription` to your `Info.plist`. Otherwise, your app will crash when trying to request permissions.

## Disable the Application Idle Timer

This is optional but recommended. It prevents your device from going to sleep while using the broadcast SDK, which would interrupt the broadcast.

```
override func viewDidAppear(_ animated: Bool) {
```

```
super.viewDidAppear(animated)
UIApplication.shared.isIdleTimerDisabled = true
}
override func viewDidDisappear(_ animated: Bool) {
super.viewDidDisappear(animated)
UIApplication.shared.isIdleTimerDisabled = false
}
```

## (Optional) Set Up AVAudioSession

By default, the broadcast SDK will set up your application's AVAudioSession. If you want to manage this yourself, set `IVSBroadcastSession.applicationAudioSessionStrategy` to `noAction`. Without control of the AVAudioSession, the broadcast SDK cannot manage microphones internally. To use microphones with the `noAction` option, you can create an `IVSCustomAudioSource` and provide your own samples via an `AVCaptureSession`, `AVAudioEngine` or another tool that provides PCM audio samples.

If you are manually setting up your AVAudioSession, at a minimum you need to set the category as `.record` or `.playbackAndRecord`, and set it to `active`. If you want to record audio from Bluetooth devices, you need to specify the `.allowBluetooth` option as well:

```
do {
    try AVAudioSession.sharedInstance().setCategory(.record, options: .allowBluetooth)
    try AVAudioSession.sharedInstance().setActive(true)
} catch {
    print("Error configuring AVAudioSession")
}
```

We recommend that you let the SDK handle this for you. Otherwise, if you want to choose between different audio devices, you will need to manually manage the ports.

## Create the Broadcast Session

The broadcast interface is `IVSBroadcastSession`. Initialize it as shown below:

```
let broadcastSession = try IVSBroadcastSession(
    configuration: IVSPresets.configurations().standardLandscape(),
    descriptors: IVSPresets.devices().frontCamera(),
    delegate: self)
```

Also see [Create the Broadcast Session \(Advanced Version\)](#)

## Set the IVSImagePreviewView for Preview

If you want to display a preview for an active camera device, add the preview `IVSImagePreviewView` for the device to your view hierarchy:

```
// If the session was just created, execute the following
// code in the callback of IVSBroadcastSession.awaitDeviceChanges
// to ensure all devices have been attached.
if let devicePreview = try broadcastSession.listAttachedDevices()
    .compactMap({ $0 as? IVSImageDevice })
    .first?
    .previewView()
{
    previewView.addSubview(devicePreview)
}
```

## Start a Broadcast

The hostname that you receive in the `ingestEndpoint` response field of the `GetChannel` endpoint needs to have `rtmps://` prepended and `/app` appended. The complete URL should be in this format: `rtmps://{ ingestEndpoint }/app`

```
try broadcastSession.start(with: IVS_RTMP_URL, streamKey: IVS_STREAMKEY)
```

The iOS broadcast SDK supports only RTMPS ingest (not insecure RTMP ingest).

## Stop a Broadcast

```
broadcastSession.stop()
```

## Manage Lifecycle Events

### Audio Interruptions

There are several scenarios where the broadcast SDK will not have exclusive access to audio-input hardware. Some example scenarios that you need to handle are:

- User receives a phone call or FaceTime call
- User activates Siri



Apple makes it easy to respond to these events by subscribing to `AVAudioSession.interruptionNotification`:

```
NotificationCenter.default.addObserver(
    self,
    selector: #selector(audioSessionInterrupted(_:)),
    name: AVAudioSession.interruptionNotification,
    object: nil)
```

Then you can handle the event with something like this:

```
// This assumes you have a variable `isRunning` which tracks if the broadcast is
// currently live, and another variable `wasRunningBeforeInterruption` which tracks
// whether the broadcast was active before this interruption to determine if it should
// resume after the interruption has ended.

@objc
private func audioSessionInterrupted(_ notification: Notification) {
    guard let userInfo = notification.userInfo,
          let typeValue = userInfo[AVAudioSessionInterruptionTypeKey] as? UInt,
          let type = AVAudioSession.InterruptionType(rawValue: typeValue)
    else {
        return
    }
    switch type {
    case .began:
        wasRunningBeforeInterruption = isRunning
        if isRunning {
            broadcastSession.stop()
        }
    case .ended:
        defer {
            wasRunningBeforeInterruption = false
        }
        guard let optionsValue = userInfo[AVAudioSessionInterruptionOptionKey] as? UInt
        else { return }
        let options = AVAudioSession.InterruptionOptions(rawValue: optionsValue)
        if options.contains(.shouldResume) && wasRunningBeforeInterruption {
            try broadcastSession.start(
                with: IVS_RTMP_URL,
                streamKey: IVS_STREAMKEY)
        }
    @unknown default: break
    }
```

```
}  
}
```

## App Going Into Background

Standard applications on iOS are not allowed to use cameras in the background. There also are restrictions on video encoding in the background: since hardware encoders are limited, only foreground applications have access. Because of this, the broadcast SDK automatically terminates its session and sets its `isReady` property to `false`. When your application is about to enter the foreground again, the broadcast SDK reattaches all the devices to their original `IVSMixerSlotConfiguration` entries.

The broadcast SDK does this by responding to `UIApplicationDidEnterBackgroundNotification` and `UIApplicationWillEnterForegroundNotification`.

If you are providing custom image sources, you should be prepared to handle these notifications. You may need to take extra steps to tear them down before the stream is terminated.

See [Use Background Video](#) for a workaround that enables streaming while your application is in the background.

## Media Services Lost

In very rare cases, the entire media subsystem on an iOS device will crash. In this scenario, we can no longer broadcast. It is up to your application to respond to these notifications appropriately. At a minimum, subscribe to these notifications:

- [mediaServicesWereLostNotification](#) — Respond by stopping your broadcast and completely deallocating your `IVSBroadcastSession`. All internal components used by the broadcast session will be invalidated.
- [mediaServicesWereResetNotification](#) — Respond by notifying your users that they can broadcast again. Depending on your use case, you may be able to automatically start broadcasting again at this point.

## Advanced Use Cases

Here we present some advanced use cases. Start with the basic setup above and continue here.

## Create a Broadcast Configuration

Here we create a custom configuration with two mixer slots that allow us to bind two video sources to the mixer. One (custom) is full screen and laid out behind the other (camera), which is smaller and in the bottom-right corner. Note that for the custom slot we do not set a position, size, or aspect mode. Because we do not set these parameters, the slot uses the video settings for size and position.

```
let config = IVSBroadcastConfiguration()
try config.audio.setBitrate(128_000)
try config.video.setMaxBitrate(3_500_000)
try config.video.setMinBitrate(500_000)
try config.video.setInitialBitrate(1_500_000)
try config.video.setSize(CGSize(width: 1280, height: 720))
config.video.defaultAspectMode = .fit
config.mixer.slots = [
  try {
    let slot = IVSMixerSlotConfiguration()
    // Do not automatically bind to a source
    slot.preferredAudioInput = .unknown
    // Bind to user image if unbound
    slot.preferredVideoInput = .userImage
    try slot.setName("custom")
    return slot
  }(),
  try {
    let slot = IVSMixerSlotConfiguration()
    slot.zIndex = 1
    slot.aspect = .fill
    slot.size = CGSize(width: 300, height: 300)
    slot.position = CGPoint(x: config.video.size.width - 400, y:
config.video.size.height - 400)
    try slot.setName("camera")
    return slot
  }()
]
```

## Create the Broadcast Session (Advanced Version)

Create an `IVSBroadcastSession` as you did in the [basic example](#), but provide your custom configuration here. Also provide `nil` for the device array, as we will add those manually.

```
let broadcastSession = try IVSBroadcastSession(
    configuration: config, // The configuration we created above
    descriptors: nil, // We'll manually attach devices after
    delegate: self)
```

## Iterate and Attach a Camera Device

Here we iterate through input devices that the SDK has detected. The SDK will only return built-in devices on iOS. Even if Bluetooth audio devices are connected, they will appear as a built-in device. For more information, see [Known Issues and Workarounds](#).

Once we find a device that we want to use, we call `attachDevice` to attach it:

```
let frontCamera = IVSBroadcastSession.listAvailableDevices()
    .filter { $0.type == .camera && $0.position == .front }
    .first
if let camera = frontCamera {
    broadcastSession.attach(camera, toSlotWithName: "camera") { device, error in
        // check error
    }
}
```

## Swap Cameras

```
// This assumes you've kept a reference called `currentCamera` that points to the
// current camera.
let wants: IVSDevicePosition = (currentCamera.descriptor().position
    == .front) ? .back : .front
// Remove the current preview view since the device will be changing.
previewView.subviews.forEach { $0.removeFromSuperview() }
let foundCamera = IVSBroadcastSession
    .listAvailableDevices()
    .first { $0.type == .camera && $0.position == wants }
guard let newCamera = foundCamera else { return }
broadcastSession.exchangeOldDevice(currentCamera, withNewDevice: newCamera)
{ newDevice, _ in
    currentCamera = newDevice
    if let camera = newDevice as? IVSImageDevice {
        do {
            previewView.addSubview(try finalCamera.previewView())
        } catch {
            print("Error creating preview view \(error)")
        }
    }
}
```

```
    }  
  }  
}
```

## Create a Custom Input Source

To input sound or image data that your app generates, use `createImageSource` or `createAudioSource`. Both these methods create virtual devices (`IVSCustomImageSource` and `IVSCustomAudioSource`) that can be bound to the mixer like any other device.

The devices returned by both these methods accept a `CMSampleBuffer` through its `onSampleBuffer` function:

- For video sources, the pixel format must be `kCVPixelFormatType_32BGRA`, `420YpCbCr8BiPlanarFullRange`, or `420YpCbCr8BiPlanarVideoRange`.
- For audio sources, the buffer must contain Linear PCM data.

You cannot use an `AVCaptureSession` with camera input to feed a custom image source while also using a camera device provided by the broadcast SDK. If you want to use multiple cameras simultaneously, use `AVCaptureMultiCamSession` and provide two custom image sources.

Custom image sources primarily should be used with static content such as images, or with video content:

```
let customImageSource = broadcastSession.createImageSource(withName: "video")  
try broadcastSession.attach(customImageSource, toSlotWithName: "custom")
```

## Monitor Network Connectivity

It is common for mobile devices to temporarily lose and regain network connectivity while on the go. Because of this, it is important to monitor your app's network connectivity and respond appropriately when things change.

When the broadcaster's connection is lost, the broadcast SDK's state will change to `error` and then `disconnected`. You will be notified of these changes through the `IVSBroadcastSessionDelegate`. When you receive these state changes:

1. Monitor your broadcast app's connectivity state and call `start` with your endpoint and stream key, once your connection has been restored.

2. **Important:** Monitor the state delegate callback and ensure that the state changes to connected after calling start again.

## Detach a Device

If you want to detach and not replace a device, detach it with `IVSDevice` or `IVSDeviceDescriptor`:

```
broadcastSession.detachDevice(currentCamera)
```

## ReplayKit Integration

To stream the device's screen and system audio on iOS, you must integrate with [ReplayKit](#). The Amazon IVS broadcast SDK makes it easy to integrate ReplayKit using `IVSReplayKitBroadcastSession`. In your `RPBroadcastSampleHandler` subclass, create an instance of `IVSReplayKitBroadcastSession`, then:

- Start the session in `broadcastStarted`
- Stop the session in `broadcastFinished`

The session object will have three custom sources for screen images, app audio, and microphone audio. Pass the `CMSampleBuffers` provided in `processSampleBuffer` to those custom sources.

To handle device orientation, you need to extract ReplayKit-specific metadata from the sample buffer. Use the following code:

```
let imageSource = session.systemImageSource;
if let orientationAttachment = CMGetAttachment(sampleBuffer, key:
  RPVideoSampleOrientationKey as CFString, attachmentModeOut: nil) as? NSNumber,
    let orientation = CGImagePropertyOrientation(rawValue:
orientationAttachment.uint32Value) {
    switch orientation {
    case .up, .upMirrored:
        imageSource.setHandsetRotation(0)
    case .down, .downMirrored:
        imageSource.setHandsetRotation(Float.pi)
    case .right, .rightMirrored:
        imageSource.setHandsetRotation(-(Float.pi / 2))
    case .left, .leftMirrored:
        imageSource.setHandsetRotation((Float.pi / 2))
    }
```

```
}  
}
```

It is possible to integrate ReplayKit using `IVSBroadcastSession` instead of `IVSReplayKitBroadcastSession`. However, the ReplayKit-specific variant has several modifications to reduce the internal memory footprint, to stay within Apple's memory ceiling for broadcast extensions.

## Get Recommended Broadcast Settings

To evaluate your user's connection before starting a broadcast, use `IVSBroadcastSession.recommendedVideoSettings` to run a brief test. As the test runs, you will receive several recommendations, ordered from most to least recommended. In this version of the SDK, it is not possible to reconfigure the current `IVSBroadcastSession`, so you must deallocate it and then create a new one with the recommended settings. You will continue to receive `IVSBroadcastSessionTestResults` until the `result.status` is `Success` or `Error`. You can check progress with `result.progress`.

Amazon IVS supports a maximum bitrate of 8.5 Mbps (for channels whose type is `STANDARD` or `ADVANCED`), so the `maximumBitrate` returned by this method never exceeds 8.5 Mbps. To account for small fluctuations in network performance, the recommended `initialBitrate` returned by this method is slightly less than the true bitrate measured in the test. (Using 100% of the available bandwidth usually is inadvisable.)

```
func runBroadcastTest() {  
    self.test = session.recommendedVideoSettings(with: IVS_RTMP_URL, streamKey:  
    IVS_STREAMKEY) { [weak self] result in  
        if result.status == .success {  
            this.recommendation = result.recommendations[0];  
        }  
    }  
}
```

## Use Background Video

You can continue a non-RelayKit broadcast, even with your application in the background.

To save power and keep foreground applications responsive, iOS gives only one application at a time access to the GPU. The Amazon IVS Broadcast SDK uses the GPU at multiple stages of the video pipeline, including compositing multiple input sources, scaling the image, and encoding the

image. While the broadcasting application is in the background, there is no guarantee that the SDK can perform any of these actions.

To address this, use the `createAppBackgroundImageSource` method. It enables the SDK to continue broadcasting both video and audio while in the background. It returns an `IVSBackgroundImageSource`, which is a normal `IVSCustomImageSource` with an additional finish function. Every `CMSampleBuffer` provided to the background image source is encoded at the frame rate provided by your original `IVSVideoConfiguration`. Timestamps on the `CMSampleBuffer` are ignored.

The SDK then scales and encodes those images and caches them, automatically looping that feed when your application goes into the background. When your application returns to the foreground, the attached image devices become active again and the pre-encoded stream stops looping.

To undo this process, use `removeImageSourceOnAppBackgrounded`. You do not have to call this unless you want to explicitly revert the SDK's background behavior; otherwise, it is cleaned up automatically on deallocation of the `IVSBroadcastSession`.

**Notes:** *We strongly recommend that you call this method as part of configuring the broadcast session, before the session goes live.* The method is expensive (it encodes video), so performance of a live broadcast while this method is running may be degraded.

### Example: Generating a Static Image for Background Video

Providing a single image to the background source generates a full GOP of that static image.

Here is an example using `UIImage`:

```
// Create the background image source
guard let source = session.createAppBackgroundImageSource(withAttemptTrim: true,
  onComplete: { error in
    print("Background Video Generation Done - Error: \(error.debugDescription)")
  }) else {
  return
}

// Create a UIImage of the color red.
let ciImage = UIImage(color: .red)

// Convert the UIImage to a CVPixelBuffer
let attrs = [
  kCVPixelBufferCGImageCompatibilityKey: kCFBooleanTrue,
```



```

    kCVPixelBufferCGBitmapContextCompatibilityKey: kCFBooleanTrue,
    kCVPixelBufferMetalCompatibilityKey: kCFBooleanTrue,
] as CFDictionary

var pixelBuffer: CVPixelBuffer!
CVPixelBufferCreate(kCFAllocatorDefault,
                   videoConfig.width,
                   videoConfig.height,
                   kCVPixelFormatType_420YpCbCr8BiPlanarFullRange,
                   attrs,
                   &pixelBuffer)

let context = CIContext()
context.render(ciImage, to: pixelBuffer)

// Submit to CVPixelBuffer and finish the source
source.add(pixelBuffer)
source.finish()

```

Alternately, instead of creating a `CIImage` of a solid color, you can use bundled images. The only code shown here is how to convert a `UIImage` to a `CIImage` to use with the previous sample:

```

// Load the pre-bundled image and get it's CGImage
guard let cgImage = UIImage(named: "image")?.cgImage else {
    return
}

// Create a CIImage from the CGImage
let ciImage = CIImage(cgImage: cgImage)

```

### Example: Video with AVAssetImageGenerator

You can use an `AVAssetImageGenerator` to generate `CMSampleBuffers` from an `AVAsset` (though not an HLS stream `AVAsset`):

```

// Create the background image source
guard let source = session.createAppBackgroundImageSource(withAttemptTrim: true,
                                                         onComplete: { error in
    print("Background Video Generation Done - Error: \(error.debugDescription)")
}) else {
    return
}

```

```
// Find the URL for the pre-bundled MP4 file
guard let url = Bundle.main.url(forResource: "sample-clip", withExtension: "mp4") else
{
    return
}
// Create an image generator from an asset created from the URL.
let generator = AVAssetImageGenerator(asset: AVAsset(url: url))
// It is important to specify a very small time tolerance.
generator.requestedTimeToleranceAfter = .zero
generator.requestedTimeToleranceBefore = .zero

// At 30 fps, this will generate 4 seconds worth of samples.
let times: [NSNumber] = (0...120).map { NSNumber(time: CMTime(value: $0, timescale:
    CMTimeScale(config.video.targetFramerate))) }
var completed = 0

let context = CIContext(options: [.workingColorSpace: NSNull()])

// Create a pixel buffer pool to efficiently feed the source
let attrs = [
    kCVPixelBufferPixelFormatTypeKey: kCVPixelFormatType_420YpCbCr8BiPlanarFullRange,
    kCVPixelBufferCGImageCompatibilityKey: kCFBooleanTrue,
    kCVPixelBufferCGBitmapContextCompatibilityKey: kCFBooleanTrue,
    kCVPixelBufferMetalCompatibilityKey: kCFBooleanTrue,
    kCVPixelBufferWidthKey: videoConfig.width,
    kCVPixelBufferHeightKey: videoConfig.height,
] as NSDictionary
var pool: CVPixelBufferPool!
CVPixelBufferPoolCreate(kCFAllocatorDefault, nil, attrs, &pool)

generator.generateCGImagesAsynchronously(forTimes: times) { requestTime, image,
    actualTime, result, error in
    if let image = image {
        // convert to CIImage then CVPixelBuffer
        let ciImage = CIImage(cgImage: image)
        var pixelBuffer: CVPixelBuffer!
        CVPixelBufferPoolCreatePixelBuffer(kCFAllocatorDefault, pool, &pixelBuffer)
        context.render(ciImage, to: pixelBuffer)
        source.add(pixelBuffer)
    }
    completed += 1
    if completed == times.count {
        // Mark the source finished when all images have been processed
        source.finish()
    }
}
```

```
}  
}
```

It is possible to generate `CVPixelBuffers` using an `AVPlayer` and `AVPlayerItemVideoOutput`. However, that requires using a `CADisplayLink` and executes closer to real-time, while `AVAssetImageGenerator` can process the frames much faster.

## Limitations

Your application needs the [background audio entitlement](#) to avoid getting suspended after going into the background.

`createAppBackgroundImageSource` can be called only while your application is in the foreground, since it needs access to the GPU to complete.

`createAppBackgroundImageSource` always encodes to a full GOP. For example, if you have a keyframe interval of 2 seconds (the default) and are running at 30 fps, it encodes a multiple of 60 frames.

- If fewer than 60 frames are provided, the last frame is repeated until 60 frames are reached, regardless of the trim option's value.
- If more than 60 frames are provided and the trim option is `true`, the last N frames are dropped, where N is the remainder of the total number of submitted frames divided by 60.
- If more than 60 frames are provided and the trim option is `false`, the last frame is repeated until the next multiple of 60 frames is reached.

## How iOS Chooses Camera Resolution and Frame Rate

The camera managed by the broadcast SDK optimizes its resolution and frame rate (frames-per-second, or FPS) to minimize heat production and energy consumption. This section explains how the resolution and frame rate are selected to help host applications optimize for their use cases.

When attaching an `IVSCamera` to an `IVSBroadcastSession`, the camera is optimized for a frame rate of `IVSVideoConfiguration.targetFramerate` and a resolution of `IVSVideoConfiguration.size`. These values are provided to the `IVSBroadcastSession` on initialization.

## Known Issues and Workarounds

- A bug in ReplayKit causes rapid memory growth when plugging in a wired headset during a stream.

**Workaround:** Start the stream with the wired headset already plugged in, use a Bluetooth headset, or do not use an external microphone.

- If at any point during a ReplayKit stream you enable the microphone and then interrupt the audio session (e.g., with a phone call or by activating Siri), system audio will stop working. This is a ReplayKit bug that we are working with Apple to resolve.

**Workaround:** On an audio interruption, stop the broadcast and alert the user.

- AirPods do not record any audio if the `AVAudioSession` category is set to `record`. By default, the SDK uses `playAndRecord`, so this issue manifests only if the category is changed to `record`.

**Workaround:** If there is a chance that AirPods will be used to record audio, use `playAndRecord` even if your application is not playing back media.

- When AirPods are connected to an iOS 12 device, no other microphone can be used to record audio. Attempting to switch to an internal microphone immediately reverts back to the AirPods.

**Workaround:** None. If AirPods are connected to iOS 12, they are the only device that can record audio.

- Submitting audio data faster than realtime (using a custom audio source) results in audio drift.

**Workaround:** Do not submit audio data faster than realtime.

- Audio artifacts can appear at bitrates under 68 kbps when using a high sample rate (44100 Hz or greater) and two channels.

**Workaround:** Increase the bitrate to 68 kbps or higher, decrease the sample rate to 24000 Hz or lower, or set channels to 1.

- When echo cancellation is enabled on `IVSMicrophone` devices, only a single microphone source is returned by the `listAvailableInputSources` method.

**Workaround:** None. This behavior is controlled by iOS.

- Changing Bluetooth audio routes can be unpredictable. If you connect a new device mid-session, iOS may or may not automatically change the input route. Also, it is not possible to choose

between multiple Bluetooth headsets that are connected at the same time. This happens in both regular broadcast and stage sessions.

**Workaround:** If you plan to use a Bluetooth headset, connect it before starting the broadcast or stage and leave it connected throughout the session.

- iOS removes access to the camera when the AirPods popup appears after opening a paired AirPods case while leaving the AirPods themselves in the case. This results in the video for a broadcast or stage freezing.

**Workaround:** None. iOS completely revokes camera access while the popup is being rendered and it is impossible for third-party applications to prevent the popup.

## IVS Broadcast SDK: Mixer Guide (Low-Latency Streaming)

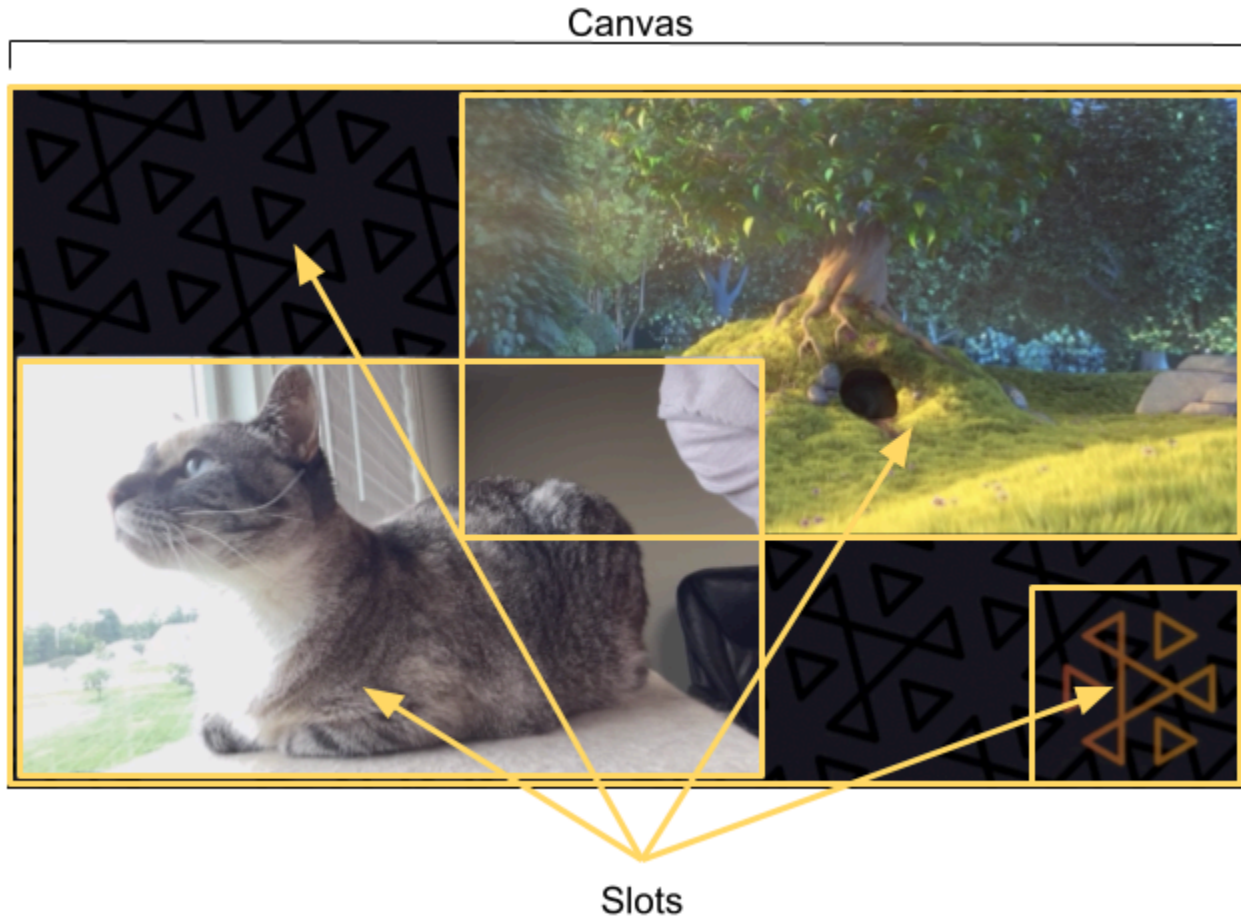
The mixer is an audio and video processing unit that takes multiple input sources and generates a single output. It is a powerful feature that lets you define and manage multiple on-screen (video) elements and audio tracks. You can combine video and audio from multiple sources such as cameras, microphones, screen captures, and audio and video generated by your app. You can use transitions to move these sources around the video that you stream to Amazon IVS, and add to and remove them mid-stream.

To access the mixer, call:

`BroadcastSession.getMixer()` on Android

`IVSBroadcastSession.mixer` on iOS

## Terminology



Term	Description
Binding	To associate an input device with a slot, the device must be <i>bound</i> to the mixer slot. This is done with the <code>Mixer.bind()</code> method. A slot can have one image input and one audio input bound to it at a time. You can unbind a device from the slot by calling <code>Mixer.unbind()</code> .
Canvas	The display extent of the video defined in your <code>BroadcastSession</code> configuration. The canvas is equal in size to your video settings and runs at the same frame rate as specified in your configuration.
Device	A hardware or software component that produces audio or image input to the <code>BroadcastSession</code> . Examples of devices are microphones, cameras, Bluetooth headsets, and virtual devices such as screen captures or custom-image inputs. With the exception of custom inputs, you generally do not need to

Term	Description
	keep a reference to the device object; instead, you can keep a copy of the device descriptor.
Device descriptor	A structure with information about an input device; e.g., its type, system address, a human-readable "friendly" name, and physical position on the mobile device. This information lets you decide if you want to use the referenced device and lets Amazon IVS access it.
Slot	<p>A container that defines a visual element's position on screen and an audio track's properties in the audio mix. A mixer can be configured with zero or more slots. Slots are given a string name that can be used to bind devices and execute transitions. The image above shows four slots:</p> <ul style="list-style-type: none"><li>• Bottom left with camera input</li><li>• Top right with movie input</li><li>• Bottom right with the Amazon IVS logo</li><li>• A full-screen background image</li></ul> <p>After configuring a session, you can add and remove slots with the <code>addSlot</code> and <code>removeSlot</code> mixer methods.</p>
Transition	<p>To move a slot to a new position or change some of its properties, use <code>Mixer.transition()</code> . This method takes:</p> <ul style="list-style-type: none"><li>• A new slot structure that represents the next state for the slot</li><li>• A duration that specifies how long the animation should take, relative to the timeline of the video. If the duration is set to 0, the transition happens on the next frame that is mixed.</li><li>• An optional callback that informs you when the animation is complete. The callback may be useful for chaining animations.</li></ul>

## Canvas Properties

Canvas properties are set based on the `BroadcastConfiguration` you provide when creating the `BroadcastSession`. Several properties in the `Audio` and `Video` structures affect the canvas:

Name	Type	Description
<code>Audio.channels</code>	Integer	Number of output channels from the audio mixer. Valid values: 1, 2. 1 channel is mono audio; 2, stereo audio. Default: 2.
<code>Audio.sampleRate</code>	AudioSampleRate	Number of audio samples per second from the audio mixer. This value should be at least twice the highest frequency in your audio signal. People can hear up to about 20 kHz, so 44.1 kHz and 48 kHz generally suffice. Default: 48 kHz.
<code>Video.defaultAspectMode</code>	AspectMode	Default aspect-ratio mode for slots. Valid values: <ul style="list-style-type: none"> <li><code>Fill</code> — Maintain the aspect ratio of the image but fill the slot. The image will be cropped if needed.</li> <li><code>Fit</code> — Maintain the aspect ratio of the image but fit the entire image into the slot. The slot may have a letterbox or pillarbox if necessary. The letter/pillarbox will be of the <code>fillColor</code> if that value was set; otherwise, transparent (which may appear black if the canvas color behind the image is black).</li> <li><code>None</code> — Do not maintain the aspect ratio of the image. The image will be scaled to match the dimensions of the slot.</li> </ul>
<code>Video.size</code>	Vec2	Size of the video canvas.
<code>Video.targetFrameRate</code>	Integer	Number of target frames per second for the canvas. On average this value should be met, but the system may drop frames under certain circumstances (e.g., high CPU load or network congestion).



## Slot Properties

Slots have several configurable properties that you can use to customize your scenes and animate. Any value that is Float or Vector is animated using linear interpolation for transitions with a duration longer than 0 seconds.

Name	Type	Description
aspect	AspectMode	<p>Aspect-ratio mode for any image rendered in the slot. Valid values:</p> <ul style="list-style-type: none"> <li><b>Fill</b> — Maintain the aspect ratio of the image but fill the slot. The image will be cropped if needed.</li> <li><b>Fit</b> — Maintain the aspect ratio of the image but fit the entire image into the slot. The slot may have a letterbox or pillarbox if necessary. The letter/pillarbox will be of the <code>fillColor</code> if that value was set; otherwise, transparent (which may appear black if the canvas color behind the image is black).</li> <li><b>None</b> — Do not maintain the aspect ratio of the image. The image will be scaled to match the dimensions of the slot.</li> </ul> <p>Default: same as the Canvas <code>aspect</code> if <code>matchCanvasAspectMode</code> is true; otherwise, <b>Fill</b>. Setting this value also sets <code>matchCanvasAspectMode</code> to false.</p>
fillColor	Vec4	<p>Fill color to be used with Aspect Fit when the slot and image aspect ratios do not match. The format is (red, green, blue, alpha). Valid value (for each channel): 0 - 1. Default: (0, 0, 0, 0).</p>
gain	Float	<p>Audio gain. This is a multiplier, so any value above 1 increases the gain; any value below 1, decreases it. Valid values: 0 - 2. Default: 1.</p>

Name	Type	Description
matchCanvasAspectMode	Boolean	If true, use the canvas's <code>Video.defaultAspectMode</code> value. This is set to false if you set the slot's aspect property. Default: true.
matchCanvasSize	Boolean	If true, the slot's size is adjusted to be equal to the size of the canvas and its position is set to (0, 0). This is set to false if you set the slot's size property. Default: true.
name	String	Name of the slot. This is used to reference the slot for bindings and transitions. Default: "default" .
position	Vec2	Slot position (in pixels), relative to the top-left corner of the canvas. The origin of the slot also is top-left.
preferredAudioInput	DeviceType	Preferred audio-input-device type. If this slot is unbound and an audio device of the specified type is attached to the session, the device automatically binds to this slot. Valid values: <ul style="list-style-type: none"> <li>Microphone — Audio hardware such as on-board microphone, pluggable headphones, or Bluetooth headphones.</li> <li>System Audio — Audio captured from the operating system, usually accompanied by a screen recording.</li> <li>User Audio — Custom audio inputs that you create.</li> <li>Unknown — There is no preferred device; the slot will always be bound manually.</li> </ul>

Name	Type	Description
preferredVideoInput	DeviceType	<p>Preferred video-input-device. If this slot is unbound and a video device of the specified type is attached to the session, the device automatically binds to this slot. Valid values:</p> <ul style="list-style-type: none"> <li>• Camera — On-board camera devices such as the front-facing, back-facing, or wide-angle camera.</li> <li>• Screen — Screen capture from the operating system.</li> <li>• User Image — Custom image and video inputs that you create.</li> <li>• Unknown — There is no preferred device; the slot will always be bound manually.</li> </ul>
size	Vec2	<p>Size of the slot, in pixels. Setting this value also sets <code>matchCanvasSize</code> to false. Default: (0, 0); however, because <code>matchCanvasSize</code> defaults to true, the rendered size of the slot is the canvas size, not (0, 0).</p>
transparency	Float	<p>Transparency of the slot. This is multiplicative with any alpha values in the image. Opacity is <math>1 - \text{transparency}</math>. Valid values: 0-1, where 0 is fully opaque and 1 is fully transparent. Default: 0.</p>
zIndex	Float	<p>Relative ordering of slots. Slots with higher <code>zIndex</code> values are drawn on top of slots with lower <code>zIndex</code> values.</p>

## Configuring a Broadcast Session for Mixing

Position 0, 0



Here, we create a scene similar to the one at the beginning of this guide, with three on-screen elements:

- Bottom-left slot for a camera.
- Bottom-right slot for a logo overlay.
- Top-right slot for a movie.

Note that the origin for the canvas is the top-left corner and this is the same for the slots. Hence, positioning a slot at (0, 0) puts it in the top-left corner with the entire slot visible.

### iOS

```
let config = IVSBroadcastConfiguration()
try config.video.setSize(CGSize(width: 1280, height: 720))
try config.video.setTargetFramerate(60)
config.video.enableTransparency = true
```

```
// Bottom Left
var cameraSlot = IVSMixerSlotConfiguration()
cameraSlot.size = CGSize(width: 320, height: 180)
cameraSlot.position = CGPoint(x: 20, y: 1280 - 200)
cameraSlot.preferredVideoInput = .camera
cameraSlot.preferredAudioInput = .microphone
cameraSlot.matchCanvasAspectMode = false
cameraSlot.zIndex = 2
try cameraSlot.setName("camera")

// Top Right
var streamSlot = IVSMixerSlotConfiguration()
streamSlot.size = CGSize(width: 640, height: 320)
streamSlot.position = CGPoint(x: 1280 - 660, y: 20)
streamSlot.preferredVideoInput = .userImage
streamSlot.preferredAudioInput = .userAudio
streamSlot.matchCanvasAspectMode = false
streamSlot.zIndex = 1
try streamSlot.setName("stream")

// Bottom Right
var logoSlot = IVSMixerSlotConfiguration()
logoSlot.size = CGSize(width: 320, height: 180)
logoSlot.position = CGPoint(x: 1280 - 340, y: 720 - 200)
logoSlot.preferredVideoInput = .userImage
logoSlot.preferredAudioInput = .unknown
logoSlot.matchCanvasAspectMode = false
logoSlot.zIndex = 3
try logoSlot.setTransparency(0.7)
try logoSlot.setName("logo")

config.mixer.slots = [ cameraSlot, streamSlot, logoSlot ]
```

## Android

```
// Bottom Left
val cameraSlot = BroadcastConfiguration.Mixer.Slot.with { s ->
    s.setSize(320, 180)
    s.position = BroadcastConfiguration.Vec2(20, 1280 - 200)
    s.preferredVideoInput = Device.Descriptor.DeviceType.CAMERA
    s.preferredAudioInput = Device.Descriptor.DeviceType.MICROPHONE
    s.matchCanvasAspectMode = false
    s.zIndex = 2
```

```
    s.name = "camera"
    s
}

// Top Right
val streamSlot = BroadcastConfiguration.Mixer.Slot.with { s ->
    s.setSize(640, 320)
    s.position = BroadcastConfiguration.Vec2(1280 - 660, 20)
    s.preferredVideoInput = Device.Descriptor.DeviceType.USER_IMAGE
    s.preferredAudioInput = Device.Descriptor.DeviceType.USER_AUDIO
    s.matchCanvasAspectMode = false
    s.zIndex = 1
    s.name = "stream"
    s
}

// Bottom Right
val logoSlot = BroadcastConfiguration.Mixer.Slot.with { s ->
    s.setSize(320, 180)
    s.position = BroadcastConfiguration.Vec2(1280 - 340, 720 - 200)
    s.preferredVideoInput = Device.Descriptor.DeviceType.USER_IMAGE
    s.preferredAudioInput = Device.Descriptor.DeviceType.UNKNOWN
    s.matchCanvasAspectMode = false
    s.zIndex = 3
    s.name = "logo"
    s.transparency = 0.7
    s
}

val config = BroadcastConfiguration.with { c ->
    c.mixer.slots = listOf(cameraSlot, streamSlot, logoSlot)
    c.video.targetFramerate = 60
    c.video.setSize(1280, 720)
    c
}
```

## Adding Slots

Once you've created a `BroadcastSession` with your configuration, you can add slots to and remove slots from the mixer. Here, we add to the mixer a large background slot for an image.

## iOS

```
// Background. We will use most of the defaults for this slot.
var backgroundSlot = IVSMixerSlotConfiguration()
backgroundSlot.preferredVideoInput = .userImage
backgroundSlot.preferredAudioInput = .unknown
backgroundSlot.matchCanvasAspectMode = false
try backgroundSlot.setName("background")

session.mixer.addSlot(backgroundSlot)
```

## Android

```
// Background. We will use most of the defaults for this slot.
val backgroundSlot = BroadcastConfiguration.Mixer.Slot.with { s ->
    s.preferredVideoInput = Device.Descriptor.DeviceType.USER_IMAGE
    s.preferredAudioInput = Device.Descriptor.DeviceType.UNKNOWN
    s.matchCanvasAspectMode = false
    s.name = "background"
    s
}

session.mixer.addSlot(backgroundSlot)
```

## Removing Slots

To remove a slot, call `BroadcastSession.Mixer.removeSlot` with the name of the slot you want to remove. Any devices bound to that slot are automatically unbound, so you must rebind them to different slots if you want to continue using them.

## Animations with Transitions

The mixer transition method replaces a slot's configuration with a new configuration. This replacement can be animated over time by setting a duration higher than 0, in seconds.

### Which Properties Can Be Animated?

Not all properties in the slot structure can be animated. Any properties based on Float types can be animated; other properties take effect at either the start or the end of the animation.

Name	Can Be Animated?	Impact Point
aspect	No	End
fillColor	Yes	Interpolated
gain	Yes	Interpolated
matchCanvasAspectMode	No	Start
matchCanvasSize	No	Start
name	No	N/A
Note: You cannot change the name of the slot.		
position	Yes	Interpolated
preferredAudioInput	No	End
preferredVideoInput	No	End
size	Yes	Interpolated
transparency	Yes	Interpolated
zIndex	Yes	Unknown
Note: The <code>zIndex</code> moves 2D planes through 3D space, so the transition happens when the two planes cross at some point in the middle of the animation. This could be computed, but it depends on the starting and ending <code>zIndex</code> values. For a smoother transition, combine this with <code>transparency</code> .		

## Simple Examples

Below are examples of a full-screen camera takeover using the configuration defined above in [Configuring a Broadcast Session for Mixing](#). This is animated over 0.5 seconds.



## iOS

```
// Bottom Left
var bigCameraSlot = cameraSlot
bigCameraSlot.size = CGSize(width: 1280, height: 720)
bigCameraSlot.position = CGPoint(x: 0, y: 0)

session.mixer.transition("camera", bigCameraSlot, 0.5) {
    println("animation completed!")
}
```

## Android

```
// Bottom Left
val bigCameraSlot = cameraSlot.changing { s ->
    s.setSize(1280, 720)
    s.position = BroadcastConfiguration.Vec2(0, 0)
    s
}

session.mixer.transition("camera", bigCameraSlot, 0.5) {
    print("animation completed!")
}
```

## Mirroring the Broadcast

To mirror an attached image device in the broadcast in this direction ...	Use a negative value for the ...
Horizontally	Width of the slot
Vertically	Height of the slot
Both horizontally and vertically	Both the width and height of the slot

The position will need to be adjusted by the same value, to put the slot in the correct position when mirrored.

Below are examples for mirroring the broadcast horizontally and vertically.

## iOS

### Horizontal mirroring:

```
var cameraSlot = IVSMixerSlotConfiguration
cameraSlot.size = CGSize(width: -320, height: 720)
// Add 320 to position x since our width is -320
cameraSlot.position = CGPoint(x: 320, y: 0)
```

### Vertical mirroring:

```
var cameraSlot = IVSMixerSlotConfiguration
cameraSlot.size = CGSize(width: 320, height: -720)
// Add 720 to position y since our height is -720
cameraSlot.position = CGPoint(x: 0, y: 720)
```

## Android

### Horizontal mirroring:

```
cameraSlot = BroadcastConfiguration.Mixer.Slot.with {
    it.size = BroadcastConfiguration.Vec2(-320f, 180f)
    // Add 320f to position x since our width is -320f
    it.position = BroadcastConfiguration.Vec2(320f, 0f)
    return@with it
}
```

### Vertical mirroring:

```
cameraSlot = BroadcastConfiguration.Mixer.Slot.with {
    it.size = BroadcastConfiguration.Vec2(320f, -180f)
    // Add 180f to position y since our height is -180f
    it.position = BroadcastConfiguration.Vec2(0f, 180f)
    return@with it
}
```

**Note:** This mirroring is different than the `setMirrored` method on `ImagePreviewView` (Android) and `IVSImagePreviewView` (iOS). That method affects only the local preview view on the device and does not impact the broadcast.

# IVS Broadcast SDK: Custom Image Sources (Low-Latency Streaming)

This guide assumes you are already familiar with how to set up a broadcast session ([Android](#), [iOS](#)) and how to [use the mixer API](#).

Custom image-input sources allow an application to provide its own image input to the broadcast SDK, instead of being limited to the preset cameras or screen share. A custom image source can be as simple as a semi-transparent watermark or static "be right back" scene, or it can allow the app to do additional custom processing like adding beauty filters to the camera.

You can have multiple custom image sources, like a watermark plus a camera with beauty filters. When you use a custom image-input source for custom control of the camera (such as using beauty-filter libraries that require camera access), the broadcast SDK is no longer responsible for managing the camera. Instead, the application is responsible for handling the camera's lifecycle correctly. See official platform documentation on how your application should manage the camera.

## Android

After you create a broadcast session, create an image-input source:

```
SurfaceSource surfaceSource = broadcastSession.createImageInputSource();
```

This method returns a `SurfaceSource`, which is an image source backed by a standard Android [Surface](#). It is automatically attached to the broadcast session, so there is no need to use the `attachDevice(...)` method afterward. However, the `SurfaceSource` needs to be bound to a slot; this is covered later below. The `SurfaceSource` can be resized and rotated. You also can create an `ImagePreviewView` to display a preview of its contents.

To retrieve the underlying `Surface`:

```
Surface surface = surfaceSource.getInputSurface();
```

This `Surface` can be used as the output buffer for image producers like `Camera2`, `OpenGL ES`, and other libraries. The simplest use case is directly drawing a static bitmap or color into the `Surface`'s `Canvas`. However, many libraries (such as beauty-filter libraries) provide a method that allows an application to specify an external `Surface` for rendering. You can use such a method to pass this

Surface to the filter library, which allows the library to output processed frames for the broadcast session to stream.

Finally, the `SurfaceSource` must be bound to a `Mixer.Slot` to be streamed by the broadcast session:

```
broadcastSession.getMixer().bind(surfaceSource, "customSlot");
```

The [Android sample code](#) has several examples that use a custom image source in different ways:

- A semi-transparent watermark is added in the `MixerActivity`.
- An MP4 file is looped in the `MixerActivity`.
- The [CameraManager](#) utility class does custom management of the device camera using the `Camera2` method in the `CustomActivity`, which applies a simple sepia filter. This example is especially helpful since it shows how to manage the camera and pass the broadcast session's custom `SurfaceSource` to the camera capture request. If you use other external libraries, follow their documentation on how to configure the library to output to the Android Surface provided by the broadcast session.

## iOS

After you create the broadcast session, create an image-input source:

```
let customSource = broadcastSession.createImageSource(withName: "customSourceName")
```

This method returns an `IVSCustomImageSource`, which is an image source that allows the application to submit `CMSampleBuffers` manually. For supported pixel formats, see the [iOS Broadcast SDK Reference](#); a link to the most current version is in the [Amazon IVS Release Notes](#) for the latest broadcast SDK release. The source is not automatically attached to the broadcast session, so you must attach the image source to the session and bind it to a slot before the source will stream:

```
broadcastSession.attach(customSource, toSlotWithName: "customSourceSlot", onComplete: nil)
```

After the custom source is attached and bound, the application can submit `CMSampleBuffers` directly to the custom source. You may choose to use the `onComplete` callback to start doing so.

Samples submitted to the custom source will be streamed in the broadcast session:

```
customSource.onSampleBuffer(sampleBuffer)
```

For streaming video, use this method in a callback. For example, if you're using the camera, then every time a new sample buffer is received from an `AVCaptureSession`, the application can forward the sample buffer to the custom image source. If desired, the application can apply further processing (like a beauty filter) before submitting the sample to the custom image source.

For a static image, after the first sample, the application needs to resubmit the sample if the custom image source's slot binding is changed or the source is detached and reattached to the broadcast session. For example, if you remove the slot from and then add the slot to the mixer, you must resubmit the sample.

The [iOS sample app](#) has several examples that use a custom image source in different ways:

- A semi-transparent watermark is added in `MixerViewController`.
- An MP4 file is looped in `MixerViewController`.
- A `CIFilter` implementation with a device camera is added in `CustomSourcesViewController`. This allows an application to manage a device camera independently of the Amazon IVS Broadcast SDK. It uses `AVCaptureSession` to capture an image from the device camera, processes the image using a `CIFilter` implementation, and submits `CMSampleBuffers` to `customSource` for live streaming.

# Amazon IVS Player SDK

To use Amazon Interactive Video Service (IVS), you must use the Amazon IVS Player. The Player is a cross-platform suite of SDKs for playback of Amazon IVS streams. It is designed to leverage the Amazon IVS architecture and optimized for Amazon IVS playback.

The only player whose performance we can guarantee is the Amazon IVS player. To achieve low latency, the Amazon IVS player is required.

Key features of the Amazon IVS player are:

- **Low-latency streaming** — Low latency is a critical component in building good interactive user experiences that enrich the audience experience. Latency creeps in incrementally throughout the transmission path between broadcaster and viewer, eroding responsiveness.

End-to-end latency is the delay from when a live stream is captured on camera to when it appears on a viewer's screen. Amazon IVS is designed to deliver low end-to-end latency (under five seconds, depending on the broadcast location and broadcaster settings). *To achieve this low latency, the Amazon IVS player is required.*

- **Cross-platform consistency** — Viewers watch broadcasts on a variety of platforms. From mobile devices to web browsers, the Amazon IVS Player gives all viewers a similar experience. This consistency is possible because every platform uses the same library of player functions. The player library is an integral component of the Amazon IVS architecture. Using one video stack ensures that all video-playback behaviors — including low-latency mode, timed metadata, analytics, error tracking, reporting, and logging — are available in a consistent way on all supported platforms.
- **Adaptive bitrate streaming (ABR)** — The Amazon IVS Player uses ABR algorithms optimized for low-latency environments. The Player measures quality of service and bandwidth availability in real time and adapts video quality and buffer levels, to provide uninterrupted playback. When connection quality suffers, ABR switches to a lower bitrate; when connection quality improves, it switches to a higher bitrate.
- **Timed metadata** — The Amazon IVS Player supports *timed metadata*, which can be used to build interactive elements such as polls and quizzes. Metadata is a set of data that describes and gives information about other data. With "timed" metadata, a timecode accompanies the piece of data about the stream. During playback, the timecode serves as a cue point to trigger action based on the data, such as:

- Sending player statistics for a sports stream
- Sending product details for a live shopping stream
- Sending questions for a live quiz stream
- **Robust error handling** — Handling transient errors well avoids interruptions in the viewing experience. The Amazon IVS Player's robust error handling detects many potential streaming errors, automatically switching to an alternative rendition. Viewers continue watching the broadcast uninterrupted, without having to take any corrective action.
- **Ease of integration** — The Amazon IVS Player API bridges the gap between Amazon IVS customers' applications and the Player library. The API has bindings for all supported platforms, making it easy to integrate the Player into applications while using familiar coding environments and techniques. With full control over UI elements, customers can customize the branding and presentation aspects of their applications.

The Amazon IVS player does not support casting with Airplay. Casting with Chromecast can be implemented outside the player using the default Chromecast receiver apps. However, latency in those apps is higher than in the Amazon IVS player SDK, so the switch will not be seamless. Also see our documentation on the Amazon IVS Broadcast SDK: for [Low-Latency Streaming](#) and for [Real-Time Streaming](#).

## Browser & Platform Requirements

For details on the latest released versions of various browsers, see:

- [Chrome Platform Status](#)
- [Firefox Releases](#)
- [Microsoft Edge Release Schedule](#)
- [Safari Release Notes](#)

While Amazon IVS may work with some older browsers, we do not fix bugs related to older browsers.

The IVS Player Web SDK (including the Video.js and Player JW integrations) is not supported in browser-like environments. This includes Native WebViews and "10-foot devices" (TVs, consoles, set-top boxes) which support web applications. Please contact IVS Support if you're unsure of specific browser support outside of the tables listed below.

## Desktop Browsers

Desktop Browser	Supported Platforms	Supported Versions
Chrome	Windows, macOS	Two major versions (current and most recent prior version)
Firefox	Windows, macOS	Two major versions (current and most recent prior version)
Edge	Windows 8.1 and later	44.0 and later  (In auto-quality mode on <a href="#">Microsoft Edge Legacy</a> , only normal-latency playback is supported, not low-latency playback. Auto-quality mode refers to whether ABR is enabled. For example, on the Web player, see <code>setAutoQualityMode</code> .)
Safari	macOS	Two major versions (current and most recent prior version)  (In auto-quality mode on Safari for macOS 14 and above, IVS Player 1.3.0 and above support low-latency playback. For earlier versions of Safari and IVS Player, only normal-latency playback is supported. See above for "auto-quality mode.")

## Mobile Browsers

Mobile Browser	Supported Versions
Chrome for iOS, Safari for iOS	Two major versions (current and most recent prior version)  (Low-latency playback is not supported. Normal latency playback is supported. This constraint applies to all browsers for iOS.)



Mobile Browser	Supported Versions
	(Timed metadata is supported only in Player 1.3.0 and later.)
Chrome for iPadOS, Safari for iPadOS	Two major versions (current and most recent prior version)  (When "Request Mobile Website" is selected: <ul style="list-style-type: none"> <li>• Low-latency playback is not supported.</li> <li>• Timed metadata is supported only in Player 1.3.0 and later.)</li> </ul>
Chrome for Android	Two major versions (current and most recent prior version)

## Native Platforms

Platform	Supported Versions	Supported Devices
Android	5.0 (Lollipop) and later	Phones and tablets
iOS	12 and later  <b>iOS 12 Support Deprecation:</b> Starting with Player SDK 1.30.0, we will no longer support iOS 12. <i>To continue using the latest Player SDKs after that, you must update your applications to target iOS 13+.</i>	All

IVS supports a minimum of 4 major iOS versions and 6 major Android versions. Our current version support may extend beyond these minimums. Customers will be notified via SDK release notes at least 3 months in advance of a major version no longer being supported.

## Reducing Latency in Third-Party Players

**For Basic and Standard channel types:** For the lowest possible latency, you must use the Amazon IVS player. In third-party players (including iOS Safari), you can reduce latency to about 10 seconds by using the following configuration:

- Set your encoder's (e.g. OBS) keyframe interval to 2 seconds or below.
- Add `?keyframeInterval=2` to the RTMP(S) URL. For example: `rtmps://a1b2c3d4e5f6.global-contribute.live-video.net:443/app/sk_us-west-2_abcd1234efgh5678ijkl?keyframeInterval=2`

**Note:** The keyframe interval specified as part of the RTMP URL must be greater than or equal to the value configured in the encoder; otherwise, you may have playback issues. You can set the value to any integer between 2 and 6 inclusive, but 2 enables the lowest latency.

**For Advanced channel types:** The above guidance does not apply. Advanced channel types generate keyframe intervals automatically for encoding efficiency, with at most 2 seconds between keyframes, regardless of the source encoding keyframe interval setting.

### iOS Safari

In iOS Safari, you can reduce latency to approximately 6-8 seconds by using the IVS player and configuring it to use a service worker. See [Set Up Service Worker](#) in the *Player SDK: Web Guide* for implementation details and a reference sample.

**Note:** Getting the lowest latency requires an IVS stream with the keyframe interval set to 2 seconds.

## Audio-Only Playback

All IVS channel types support audio-only renditions. This can be particularly valuable for mobile applications. For instance, in your mobile app, you can switch the player to the audio-only rendition when the user backgrounds the application to conserve bandwidth.

For ADVANCED-SD and ADVANCED-HD channels, the audio-only rendition is included automatically in the multivariant playlist. For BASIC and STANDARD channels, you must append the `?allow_audio_only=true` query parameter to the playback URL to enable inclusion of the audio-only rendition.

Note: The IVS web player SDK supports audio-only playback only in versions 1.24.0 and later.

## Support

If you encounter a playback error or other playback issue with your stream, determine the unique playback session identifier via the player API.

For this Amazon IVS player:	Use this:
Android	<code>sessionId</code> function
iOS	<code>sessionId</code> property of <code>IVSPlayer</code>
Web	<code>getSessionId</code> function

Share this playback session identifier with AWS support. With it, they can get information to help troubleshoot your issue.

**Note:** The Player is continually improved. See [Amazon IVS Release Notes](#) for available versions and fixed issues. If appropriate, before contacting support, update your version of the Player and see if that resolves your issue.

## Versioning

The Amazon IVS Player SDKs use [semantic versioning](#).

For this discussion, suppose:

- The latest release is 4.1.3.
- The latest release of the prior major version is 3.2.4.
- The latest release of version 1.x is 1.5.6.

Backward-compatible new features are added as minor releases of the latest version. In this case, the next set of new features will be added as version 4.2.0.

Backward-compatible, minor bug fixes are added as patch releases of the latest version. Here, the next set of minor bug fixes will be added as version 4.1.4.

Backward-compatible, major bug fixes are handled differently; these are added to several versions:

- Patch release of the latest version. Here, this is version 4.1.4.
- Patch release of the prior minor version. Here, this is version 3.2.5.
- Patch release of the latest version 1.x release. Here, this is version 1.5.7.

Major bug fixes are defined by the Amazon IVS product team. Typical examples are critical security updates and selected other fixes necessary for customers.

**Note:** In the examples above, released versions increment without skipping any numbers (e.g., from 4.1.3 to 4.1.4). In reality, one or more patch numbers may remain internal and not be released, so the released version could increment from 4.1.3 to, say, 4.1.6.

## Amazon IVS Player SDK: Web Guide

The Amazon Interactive Video Service (IVS) Player SDK for Web can be integrated with [player frameworks](#) like Video.js or used standalone on top of an HTML `<video>` element.

**Latest version of Web player:** 1.29.0 ([Release Notes](#))

**Reference documentation:** For information on the most important methods available in the Amazon IVS Web player, see the reference documentation at <https://aws.github.io/amazon-ivs-player-docs/1.29.0/web/>.

## Getting Started

We provide support through a `script` tag as well as through an npm module.

### Demos

The following live demo shows how to use the Web player with a `script` tag from our Content Delivery Network: [Amazon IVS Player Sample](#). The demo includes setting up event listeners.

Also see <https://github.com/aws-samples/amazon-ivs-player-web-sample> for a selection of additional Web player demos.

### Setup With Script Tag

To set up the Amazon IVS player using the `script` tag:

1. Include the following tag (for the latest version of the player).

```
<script src="https://player.live-video.net/1.29.0/amazon-ivs-player.min.js"></script>
```

2. Once `amazon-ivs-player.min.js` is loaded, it adds an `IVSPlayer` variable to the global context. This is the library you will use to create a player instance. First, check `isPlayerSupported` to determine if the browser supports the IVS player:

```
if (IVSPlayer.isPlayerSupported) { ... }
```

Then, to create a player instance, call the `create` function on the `IVSPlayer` object.

```
const player = IVSPlayer.create();
```

The Amazon IVS Player SDK for Web uses web workers to optimize video playback.

3. Load and play a stream using the `load` and `play` functions on the player instance:

```
player.load("PLAYBACK_URL");  
player.play();
```

where `PLAYBACK_URL` is the URL returned from the Amazon IVS API when a stream key is requested.

## Sample Code

In this example, replace `PLAYBACK_URL` with the URL of the source stream you want to load. The example uses the latest version of the Amazon IVS player.

```
<script src="https://player.live-video.net/1.29.0/amazon-ivs-player.min.js"></script>  
<video id="video-player" playsinline></video>  
<script>  
  if (IVSPlayer.isPlayerSupported) {  
    const player = IVSPlayer.create();  
    player.attachHTMLVideoElement(document.getElementById('video-player'));  
    player.load("PLAYBACK_URL");  
    player.play();  
  }  
</script>
```

In the `<video>` tag, `playsinline` is required for inline playback on iOS Safari. See <https://webkit.org/blog/6784/new-video-policies-for-ios/>.

## Setup With NPM

For guidance, including an example Webpack configuration file, see the following repository: <https://github.com/aws-samples/amazon-ivs-player-web-sample>.

**Note:** When hosting player static assets from your own domain, you must set the "Content-Type" response header for the WebAssembly binary (`amazon-ivs-wasmworker.min.wasm`) to "application/wasm." You also should gzip your assets to reduce bytes downloaded over the wire and improve the player's time to start playback.

## TypeScript

If you're using TypeScript, the npm package includes types you may want to import and use. For information on these types, see the [Amazon IVS Player SDK: Web Reference](#).

## Set Up Service Worker

To lower latency further when playing via browsers that only support native playback (primarily iOS Safari), a service worker can be set up and configured. For more context, see [Reducing Latency in Third-Party Players](#).

To set up the Amazon IVS player to use a service worker:

1. Create a file to load the IVS service worker off the CDN. This is required as service workers must be hosted on the same domain as the page that pulls them in.

Create a file named `amazon-ivs-service-worker-loader.js` or similar and add the following line:

```
importScripts('https://player.live-video.net/1.29.0/amazon-ivs-service-worker.min.js');
```

2. When creating a player instance, pass in the following `serviceWorker` config referencing the `amazon-ivs-service-worker-loader.js` file:

```
const player = IVSPlayerPackage.create({
  serviceWorker: {
    url: 'amazon-ivs-service-worker-loader.js'
  }
});
```

```
});
```

3. On the video element, set the `crossOrigin` attribute to `anonymous`. This is required to allow the service worker to make changes to the manifest.

**Note:** To test the service worker locally, the page either needs to be served off `localhost` or `https`.

For a live demo, see the service worker example in the following repository:

<https://github.com/aws-samples/amazon-ivs-player-web-sample>

## Audio-Only Playback

Audio-only quality must be manually selected with the `setQuality()` method. Note that the player does not support a `true` value for the second argument, `adaptive`, so by default, this argument is `false`.

To set the quality to audio-only before playback begins, call `setQuality()` inside the `READY` event:

```
player.addEventListener(PlayerState.READY, () => {
  const qualities = player.getQualities();
  const audioOnly = qualities.find(q => q.name === 'audio_only');
  if (audioOnly) {
    player.setQuality(audioOnly);
  }
});
```

Setting the quality within `READY` works for both `autoplay` and `non-autoplay` modes.

## Framework Integrations

The Amazon IVS Player SDK for Web is designed to be easy to integrate with your framework of choice. We offer an official `Video.js` integration (“tech,” in `Video.js` jargon).

The following is a brief comparison of the Web players we offer:

Player Type	Description	UI	Plugins
Amazon IVS Player SDK for Web	A lightweight and customizable option for developers who want more control.	No	No

Player Type	Description	UI	Plugins
<a href="#">Amazon IVS Player Tech for Video.js</a>	A full-featured option, which may be appropriate if you already use Video.js and want a turnkey solution.	Yes <a href="#">(Video.js Skins)</a>	Yes <a href="#">(Video.js Plugins)</a>
<a href="#">Amazon IVS Player Provider for JW Player</a>	A full-featured option, which may be appropriate if you already use JW Player and want a turnkey solution.	Yes	N/A

## Working With Content Security Policy

The Amazon IVS Web player SDK is configured to work on pages that use Content Security Policy (CSP). A few key CSP directives must be in place. Here, we describe a minimal set of directives that are necessary. Additional directives and sources are likely necessary, depending on your specific setup.

The following directives are the minimum required for CSP:

```
worker-src blob;;
media-src blob;;
connect-src *.live-video.net;
script-src 'wasm-unsafe-eval';
```

**Note:** Older versions of browsers may not recognize one or more of those above CSP rules (such as `wasm-unsafe-eval`) and instead could require a very lenient CSP policy (`unsafe-eval`). However, that works against the whole point of CSP to limit dangerous JavaScript from running on a page. Instead, as a workaround, we recommend that you host the library assets on the same origin as your page.

## Known Issues and Workarounds

- When playing recorded content (also known as VOD) on an iOS mobile browser (e.g. Safari or Chrome), seeking backwards will mute the player.

**Workaround:** Call `player.setMuted(false)` after seeking.



- When playing recorded content on an iOS mobile browser, seeking backwards works intermittently when directly selecting the desired position.

**Workaround:** Drag the seek bar to the desired position.

- When playing recorded content on an iOS mobile browser, `player.seekTo()` calls do not consistently work.

**Workaround:** Set `currentTime` on the video HTML element after the `loadeddata` event. For example:

```
videoEl.addEventListener('loadeddata', () => {
  videoEl.currentTime = 30; // seek 30s from the beginning
});
```

- When playing a live stream or recorded content on an iOS mobile browser, captions may not be rendered in different sizes and may be re-rendered multiple times.

**Workaround:** None.

- When playing a live stream or recorded content on an iOS mobile browser, `player.getQualities()` calls do not return the list of available qualities.

**Workaround:** None. The player supports only auto-quality mode on iOS browsers.

- When native HTML5 controls are enabled, calls to `setQuality()` are ignored.

**Workaround:** Disable HTML5 controls before calling `player.setQuality()`.

- When playing a muted live stream on an iOS mobile browser, player instability (e.g., black or frozen screen, buffering) may be seen when resuming an inactive player tab (e.g., tab switches or device lock/unlock).

**Workaround:** Use the JavaScript [Page Visibility API](#) to detect page visibility changes and then take action on the player accordingly. For example:

```
//if client platform is iOS
if (!!navigator.platform && /iPad|iPhone|iPod/.test(navigator.platform)) {
  document.addEventListener("visibilitychange", () => {
    if (document.visibilityState === "hidden" && player.isMuted()) {
      player.pause()
    }
    if (document.visibilityState === "visible" &&
        player.getState() !== PlayerState.PLAYING) {
      player.play()
    }
  });
}
```

```
    }  
  })  
}
```

## Amazon IVS Player SDK: Android Guide

The Amazon Interactive Video Player (IVS) Android player SDK provides the interfaces required to use the Amazon IVS player on Android.

We guarantee playback performance only for Android mobile devices (phones and tablets). We do not support Android TV, Fire TV, IoT devices, and emulators.

The `com.amazonaws.ivs.player` package implements the interface described in this document. The following operations are supported:

- Set up (initialize) a player.
- Manage playback.
- Manage quality.
- Receive events.
- Receive errors.

**Latest version of Android player:** 1.29.0 ([Release Notes](#))

**Reference documentation:** For information on the most important methods available in the Amazon IVS Android player, see the reference documentation at <https://aws.github.io/amazon-ivs-player-docs/1.29.0/android/>.

**Sample code:** See the Android sample repository on GitHub: <https://github.com/aws-samples/amazon-ivs-player-android-sample>.

**Platform requirements:** Android 5.0 (Lollipop) or greater is required for development.

A **React Native wrapper** for the Amazon IVS Player SDK is available. For the code and documentation, see <https://github.com/aws/amazon-ivs-react-native-player>.

# Getting Started

## Install the Library

To add the Amazon IVS Android player library to your Android development environment, add the library to your module's `build.gradle` file, as shown here (for the latest version of the Amazon IVS player).

```
repositories {  
    mavenCentral()  
}  
  
dependencies {  
    implementation 'com.amazonaws:ivs-player:1.29.0'  
}
```

Alternately, to install the SDK manually, download the latest version from this location:

<https://search.maven.org/artifact/com.amazonaws/ivs-player>

## Create the Player and Set Up Event Listener

The player interface is `com.amazonaws.ivs.player.Player`. Initialize it as shown below:

```
// Create a player instance  
// <this> refers to the current Android Activity  
player = Player.Factory.create(this);  
  
// Set up to receive playback events and errors  
player.addListener(this);
```

Alternately, initialize by using `PlayerView`:

```
// Create a player instance  
// <this> refers to the current Android Activity  
PlayerView playerView = new PlayerView(this);  
Player player = playerView.getPlayer();  
// Set up to receive playback events and errors  
player.addListener(this);
```

**Note:** The listener callback methods are executed in the main thread of your Android application.

## Set the Surface View for Video

If not using `PlayerView` add a `SurfaceView` to your Android UI layout for displaying a video. This `Surface` must be available before you can play any video streams. You can access the underlying surface through the `SurfaceHolder` interface, which is retrieved by calling `getHolder()`. (See [SurfaceView](#) in the Android developer reference). Use the `SurfaceHolder.Callback` to receive events about surface changes (see [SurfaceHolder.Callback](#)).

```
surfaceView = (SurfaceView) findViewById(R.id.surfaceView);
surfaceView.getHolder().addCallback(this);

@Override
public void surfaceCreated(SurfaceHolder holder) {
    this.surface = holder.getSurface();
    if (player != null) {
        player.setSurface(this.surface);
    }
}

@Override
public void surfaceDestroyed(SurfaceHolder holder) {
    this.surface = null;
    if (player != null) {
        player.setSurface(null);
    }
}
```

## Play a Stream

Because the stream is loaded asynchronously, the player must be in a `READY` state before your application can call the `play` method to begin playback. Use the `Player.Listener` interface to determine when the player is in the correct state.

See the following sample code:

```
player.load(Uri.parse(url));

@Override
public void onStateChanged(Player.State state) {
```

```
switch (state) {
    case BUFFERING:
        // player is buffering
        break;
    case READY:
        player.play();
        break;
    case IDLE:
        break;
    case PLAYING:
        // playback started
        break;
}
```

## Release the Player

The `player.release()` method *must* be called when the player is no longer in use, to free the resources used by the library. Typically this is done in the `onDestroy` callback of the Activity or Fragment containing the player.

```
@Override
protected void onDestroy() {
    super.onDestroy();
    player.removeListener(this);
    player.release();
}
```

After the `player.release()` method is called the player can no longer be used.

## Permissions

The Android player SDK requires the following permission:

```
<uses-permission android:name="android.permission.INTERNET" />
```

In addition, these optional permissions can improve the playback experience:

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
```

## Thread Safety

The player API is not thread safe. All calls made to a player instance should be from the same thread.

## SDK Size

The Amazon IVS player SDKs are designed to be as lightweight as possible. For current information about SDK size, see the [Release Notes](#).

**Important:** When evaluating size impact, the size of the AAB/APK produced by Android Studio is not representative of the size of your app downloaded to a user's device. The Google Play Store performs optimizations to reduce the size of your app. We recommend that you use [Android App Bundles](#) to serve optimized apps for each device configuration.

## Known Issues and Workarounds

- The Android player SDK has a runtime dependency on OkHttp version 4.x. Using OkHttp version 3.x may cause instability or crashes due to an API signature mismatch and OkHttp backwards compatibility issues. Specifically, the player depends on OkHttp version 4.2.2, but it should be compatible with any 4.x version.

**Workaround:** Use a 4.x version of OkHttp or remove OkHttp from your application.

- When using an Android 11 (API level 30) emulator, you may experience video-layout issues (specifically, zooming of the stream).

**Workaround:** Play back on the real device instead.

## Amazon IVS Player SDK: iOS Guide

The Amazon Interactive Video Service (IVS) iOS player provides the interfaces required to use the Amazon IVS player on iOS.

**Latest version of iOS player:** 1.29.0 ([Release Notes](#))

**Reference documentation:** For information on the most important methods available in the Amazon IVS iOS player, see the reference documentation at <https://aws.github.io/amazon-ivs-player-docs/1.29.0/ios/>.

**Sample code:** See the iOS sample repository on GitHub: <https://github.com/aws-samples/amazon-ivs-player-ios-sample>.

**Platform requirements:** Xcode 11 or greater is required for development. The SDK supports deployment targets of iOS 12 and above, for both physical devices and the iOS Simulator.

A **React Native wrapper** for the Amazon IVS Player SDK is available. For the code and documentation, see <https://github.com/aws/amazon-ivs-react-native-player>.

## Getting Started

We recommend that you integrate the player SDK via CocoaPods. (Alternately, you can manually add the framework to your project.)

### Recommended: Integrate the Player SDK (CocoaPods)

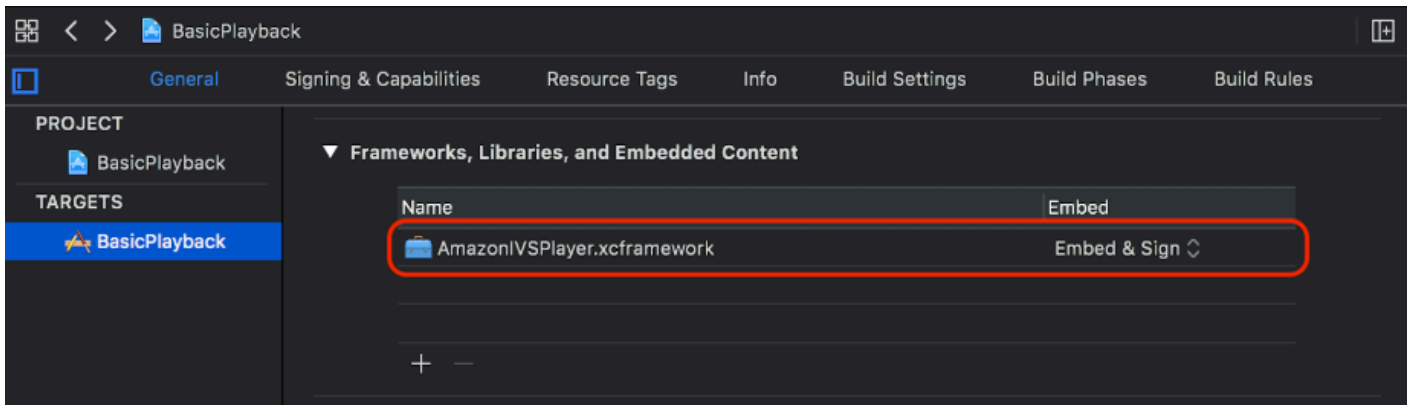
Releases are published via CocoaPods under the name `AmazonIVSPlayer`. Add this dependency to your Podfile:

```
pod 'AmazonIVSPlayer'
```

Run `pod install` and the SDK will be available in your `.xcworkspace`.

### Alternate Approach: Install the Framework Manually

1. Download the latest version from <https://player.live-video.net/1.29.0/AmazonIVSPlayer.xcframework.zip>.
2. Extract the contents of the archive. `AmazonIVSPlayer.xcframework` contains the SDK for both device and simulator.
3. Embed `AmazonIVSPlayer.xcframework` by dragging it into the **Frameworks, Libraries, and Embedded Content** section of the **General** tab for your application target:



## Create Player

The player object is `IVSPlayer`. It can be initialized as shown below:

### Swift

```
import AmazonIVSPlayer

let player = IVSPlayer()
```

### Objective-C

```
#import <AmazonIVSPlayer/AmazonIVSPlayer.h>

IVSPlayer *player = [[IVSPlayer alloc] init];
```

## Set Up Delegate

Delegate callbacks provide information on playback state, events, and errors. All callbacks are invoked on the main queue.

### Swift

```
// Self must conform to IVSPlayer.Delegate
player.delegate = self
```

### Objective-C

```
// Self must conform to IVSPlayer.Delegate
```



```
player.delegate = self
```

## Display Video

The player displays video in a custom layer, `IVSPlayerLayer`. The SDK also provides `IVSPlayerView`, a `UIView` subclass backed by this layer. Use whichever is more convenient for your application's UI.

In both cases, display the video from a player instance by using the `player` property.

### Swift

```
// When using IVSPlayerView:  
playerView.player = player  
  
// When using IVSPlayerLayer:  
playerLayer.player = player
```

### Objective-C

```
// When using IVSPlayerView:  
playerView.player = player;  
  
// When using IVSPlayerLayer:  
playerLayer.player = player;
```

## Load a Stream

The player loads the stream asynchronously. Its state indicates when it is ready to play.

### Swift

```
player.load(url)
```

### Objective-C

```
[player load:url];
```

## Play a Stream

When the player is ready, use `play` to begin playback. Use the delegate interface or key-value observing on the `state` property to observe the state change. Here is an example of the delegate-based approach:

### Swift

```
func player(_ player: IVSPlayer, didChangeState state: IVSPlayer.State) {
    if state == .ready {
        player.play()
    }
}
```

### Objective-C

```
- (void)player:(IVSPlayer *)player didChangeState:(IVSPlayerState)state {
    if (state == IVSPlayerStateReady) {
        [player play];
    }
}
```

## Pause On App Backgrounding

The player does not support playback while the app is in the background, but it does not need to be fully torn down. Pausing is sufficient; see the examples below.

### Swift

```
override func viewDidLoad() {
    super.viewDidLoad()

    NotificationCenter.default.addObserver(self,
        selector: #selector(applicationDidEnterBackground(_:)),
        name: UIApplication.didEnterBackgroundNotification,
        object: nil)
}

@objc func applicationDidEnterBackground(_ notification: NSNotification) {
    playerView?.player?.pause()
}
```

```
}
```

## Objective-C

```
- (void)viewDidLoad {
    [super viewDidLoad];

    NotificationCenter *defaultCenter = NotificationCenter.defaultCenter;
    [defaultCenter addObserver:self
                       selector:@selector(applicationDidEnterBackground:)
                       name:UIApplicationDidEnterBackgroundNotification
                       object:nil];
}

- (void)applicationDidEnterBackground:(NSNotification *)notification {
    [playerView.player pause];
}
```

## Thread Safety

The player API is not thread safe. You should create and use a player instance from the application main thread.

## Putting It All Together

The following simple, view-controller snippet loads and plays a URL in a player view. Note that the `playerView` property is initialized from an XIB/Storyboard, and its class is set to `IVSPlayerView` in Interface Builder [using the Custom Class section of the Identity Inspector](#).

## Swift

```
import AmazonIVSPlayer

class MyViewController: UIViewController {
    ...
    // Connected in Interface Builder
    @IBOutlet var playerView: IVSPlayerView!

    override func viewDidLoad() {
        super.viewDidLoad()
    }
}
```

```

NotificationCenter.default.addObserver(self,
    selector: #selector(applicationDidEnterBackground(_:)),
    name: UIApplication.didEnterBackgroundNotification,
    object: nil)
}

@objc func applicationDidEnterBackground(_ notification: NSNotification) {
    playerView?.player?.pause()
}
...
// Assumes this view controller is already loaded.
// For example, this could be called by a button tap.
func playVideo(url videoURL: URL) {
    let player = IVSPlayer()
    player.delegate = self
    playerView.player = player
    player.load(videoURL)
}
}

extension MyViewController: IVSPlayer.Delegate {
    func player(_ player: IVSPlayer, didChangeState state: IVSPlayer.State) {
        if state == .ready {
            player.play()
        }
    }
}
}

```

## Objective-C

```

// MyViewController.h

@class IVSPlayerView;

@interface MyViewController: UIViewController
...
// Connected in Interface Builder
@property (nonatomic) IBOutlet IVSPlayerView *playerView;
...
@end

// MyViewController.m

```

```
#import <AmazonIVSPlayer/AmazonIVSPlayer.h>

@implementation MyViewController <IVSPlayerDelegate>
...

- (void)viewDidLoad {
    [super viewDidLoad];

    NotificationCenter *defaultCenter = NotificationCenter.defaultCenter;
    [defaultCenter addObserver:self
                       selector:@selector(applicationDidEnterBackground:)
                       name:UIApplicationDidEnterBackgroundNotification
                       object:nil];
}

- (void)applicationDidEnterBackground:(NSNotification *)notification {
    [playerView.player pause];
}

// Assumes this view controller is already loaded.
// For example, this could be called by a button tap.
- (void)playVideoWithURL:(NSURL *)videoURL {
    IVSPlayer *player = [[IVSPlayer alloc] init];
    player.delegate = self;
    playerView.player = player;
    [player load:videoURL];
}

- (void)player:(IVSPlayer *)player didChangeState:(IVSPlayerState)state {
    if (state == IVSPlayerStateReady) {
        [player play];
    }
}

...
@end
```

## SDK Size

The Amazon IVS player SDKs are designed to be as lightweight as possible. For current information about SDK size, see the [Release Notes](#).

**Important:** When evaluating size impact, the size of the IPA produced by Xcode is not representative of the size of your app downloaded to a user's device. The App Store performs optimizations to reduce the size of your app.

## Known Issues and Workarounds

- The player may crash when testing against the arm64e architecture. This only applies when targeting arm64e specifically, and does not apply to App Store builds.

**Workaround:** Do not use arm64e.

## Amazon IVS Player SDK: Video.js Integration

This document describes the most important functions available in the Amazon Interactive Video Service (IVS) Video.js player.

**Latest version of Video.js player integration:** 1.29.0 ([Release Notes](#))

### Getting Started

Amazon IVS support for Video.js is implemented through a Video.js [tech](#). We provide support through script tags as well as through an npm module. Amazon IVS supports Video.js versions 7.6.6 and later 7\*, and 8\*.

Note that when instantiating the player, the Video.js [sources option](#) is not supported. Instead, instantiate the player normally and call the Video.js `src()` function. If autoplay is enabled, the stream will start playing; otherwise, use `play()` to start playback.

### Demo

The following live demo shows how to use the Video.js integration with script tags from our Content Delivery Network: [Amazon IVS Player Video.js integration](#).

### Setup With Script Tag

To set up the Amazon IVS tech using the script tag:

1. Include the following tag (for the latest version of the player integration).

```
<script src="https://player.live-video.net/1.29.0/amazon-ivs-videojs-tech.min.js"></script>
```

## 2. Register the tech using the `registerIVSTech` function:

```
registerIVSTech(videojs);
```

where `videojs` is the object provided by Video.js.

## 3. When creating an instance of the player, add AmazonIVS as your first tech in the `techOrder` option.

When instantiating the player, the Video.js [sources option](#) is not supported. Instead, to set the source, instantiate the player normally, then call the Video.js `src()` function on it. If autoplay is enabled, the stream will start playing; otherwise, use `play()` to start playback.

## Sample Code

In this example, `PLAYBACK_URL` is the source stream you want to load. The example uses the latest version of the Amazon IVS Player.

```
<!doctype html>
<html lang="en">
<head>
  <link href="https://cdnjs.cloudflare.com/ajax/libs/video.js/7.14.3/video-js.css"
    rel="stylesheet">
  <script src="https://cdnjs.cloudflare.com/ajax/libs/video.js/7.14.3/
video.min.js"></script>
  <script src="https://player.live-video.net/1.29.0/amazon-ivs-videojs-
tech.min.js"></script>
</head>

<body>
  <div class="video-container">
    <video id="amazon-ivs-videojs" class="video-js vjs-4-3 vjs-big-play-centered"
controls autoplay playsinline></video>
  </div>
  <style>
    body {
      margin: 0;
    }
  </style>
</body>
```

```
    .video-container {
      width: 640px;
      height: 480px;
      margin: 15px;
    }
  </style>
  <script>
    (function play() {
      // Get playback URL from Amazon IVS API
      var PLAYBACK_URL = '';

      // Register Amazon IVS as playback technology for Video.js
      registerIVSTech(videojs);

      // Initialize player
      var player = videojs('amazon-ivs-videojs', {
        techOrder: ["AmazonIVS"]
      }, () => {
        console.log('Player is ready to use!');
        // Play stream
        player.src(PLAYBACK_URL);
      });
    })();
  </script>
</body>
</html>
```

## Setup With NPM

To use Amazon IVS player through npm:

1. Install the [video.js](#) npm package or ensure that your project has some other access to the Video.js library.
2. Install the `amazon-ivs-player` npm package:

```
npm install amazon-ivs-player
```

3. When you're ready to register the Amazon IVS tech, import the `registerIVSTech` function:

```
import { registerIVSTech } from 'amazon-ivs-player';
```



#### 4. Register the tech using the `registerIVSTech` function:

```
registerIVSTech(videojs, options);
```

where:

- `videojs` is the object provided by Video.js.
- `options` is the options for the Amazon IVS tech layer. Supported options are:
  - `wasmWorker`: URL where the `amazon-ivs-wasmworker.min.js` file is hosted.
  - `wasmBinary`: URL where the `amazon-ivs-wasmworker.min.wasm` file is hosted.

The worker files are in your `node_modules/` folder under `amazon-ivs-player/dist/`. You need to host them, to use the IVS player.

#### 5. When creating an instance of the player, add AmazonIVS as your first tech in the `techOrder` option:

```
const player = videojs('videojs-player', {  
  techOrder: ["AmazonIVS"]  
});
```

## TypeScript

If you're using TypeScript, our npm package includes the following types you may want to import and use.

- `VideoJSEvents`, which describes the returned structure from `getIVSEvents()`.
- `VideoJSIVSTech`, which describes the interface to a player instance that uses the AmazonIVS tech. This can be [intersected](#) with the `VideoJsPlayer` type exposed by the [@types/video.js](#) npm package.
- `TechOptions`, which describes the interface defining the configuration options you can send to `registerIVSTech()`.

For more information on these types, see the [Amazon IVS Player SDK: Web Reference](#).

## Events

To listen to standard Video.js events, use the [on](#) function of the Video.js player.

To listen to events that are specific to Amazon IVS, add and remove event listeners on the Amazon IVS Web player:

```
player.getIVSPlayer().addEventListener(event, callback);
player.getIVSPlayer().removeEventListener(event, callback);
```

where `callback` is a callback you define, and `event` is one of: `PlayerEventType` or `PlayerState`. For more information about events, see the [Amazon IVS Player SDK: Web Reference](#).

## Errors

For general Video.js errors, listen to the generic `error` event on the player:

```
player.on("error", callback);
```

For errors specific to Amazon IVS, listen on the Amazon IVS player for its own errors:

```
let playerEvent = player.getIVSEvents().PlayerEventType;
player.getIVSPlayer().addEventListener(playerEvent.ERROR, callback);
```

The callback will receive an object with the following fields:

Field	Description
<code>type</code>	The error type. Corresponds to <code>ErrorType</code> events. For more information, see <a href="#">Amazon IVS Player SDK: Web Reference</a> .
<code>code</code>	The error code.
<code>source</code>	Source of the error.
<code>message</code>	Human readable error message.

## Plugins

We provide a plugin that creates a UI toggle for available qualities. To use this plugin, it must be loaded by including the `amazon-ivs-quality-plugin.min.js` file if you are using our tech through the following script tag (for the latest version of the IVS Player):

```
<script src="https://player.live-video.net/1.29.0/amazon-ivs-quality-plugin.min.js"></script>
```

If you are using npm, import the `registerIVSQualityPlugin` from the `amazon-ivs-player` module:

```
import { registerIVSQualityPlugin } from 'amazon-ivs-player';
```

Then, once you create an instance of your Video.js player, the following calls are required to register and enable it:

```
registerIVSQualityPlugin(videojs); // where videojs is the video.js variable  
player.enableIVSQualityPlugin(); // where player is the instance of the videojs player
```

This will create a UI menu button which allows you to select a quality for the stream.

## Plugins and TypeScript

If you're using TypeScript, our npm package includes the `VideoJSQualityPlugin` type that you may want to import and use with our plugin. Plugins essentially are mixins, so this type interface is to be used as an [intersection type](#) with the `VideoJSIVSTech` typescript interface.

## Content Security Policy

The Amazon IVS Video.js API is configured to work on pages that use Content Security Policy (CSP). See the section on "Working with Content Security Policy" in the [Amazon IVS Player SDK: Web Guide](#).

## Functions

### Playback

The Amazon IVS Video.js API supports the necessary interfaces for internal use by the Video.js framework. The client application is not likely to need to use these methods directly, since Video.js does the necessary integration and presents a standard interface. However, if needed, one way to access internal Video.js and Amazon IVS player methods is to use the Video.js player object to get the needed object handle to the tech.

To access the API, retrieve the instance of your Video.js player as you would normally:

```
let player = videojs("videoTagId"); //replace videoTagId with your <video> tag's id
```

Then you can call functions on that instance.

The following are the subset of Video.js functions that the Amazon IVS tech layer overrides. For the full list of Video.js functions, see the [video.js API documentation](#).

Function	Description and Amazon IVS-Specific Information
<a href="#">currentTime</a>	Gets or sets the time (in seconds from the beginning).  Amazon IVS: We do not recommend setting current time for live streams.
<a href="#">dispose</a>	Deletes the player instance  Amazon IVS: This also deletes the Amazon IVS tech backend.
<a href="#">duration</a>	Returns the duration of the video, in seconds.  Amazon IVS: For live streams, this returns Infinity.
<a href="#">load</a>	Starts loading the <code>src()</code> data.  Amazon IVS: This is a no-op.
<a href="#">play</a>	Plays the stream that was set up via the <code>src</code> call.  Amazon IVS: If a live stream was paused, this plays the live stream from the latest frames, instead of continuing from where it was paused.
<a href="#">playbackRate</a>	Gets or sets the video-playback rate. 1.0 means normal speed; 0.5, half normal speed; 2.0, two times normal speed; and so on.  Amazon IVS: On a live stream, a get returns 1, and a set is ignored.
<a href="#">seekable</a>	Returns the <code>TimeRanges</code> of the media that can be seeked to.

Function	Description and Amazon IVS-Specific Information
	Amazon IVS: For live streams, calling <code>end(0)</code> on the return value ( <code>TimeRange</code> ) returns Infinity.

## Amazon IVS Specific

The Amazon IVS Video.js tech has additional functions for accessing behaviors specific to Amazon IVS features:

Function	Description
<a href="#"><u>getIVSPlayer</u></a>	Returns the underlying Amazon IVS player instance. The full Amazon IVS Player Web API is available through this instance. We recommend using the basic Video.js playback API as much as possible, and using this function only to access Amazon IVS-specific features. The most common functions you are likely to need to access on the Amazon IVS player instance are <code>setQuality()</code> and <code>addEventListener()</code> / <code>removeEventListener()</code> .
<a href="#"><u>getIVSEvents</u></a>	Returns an object that holds Amazon IVS-specific enums. This is used for listening to Amazon IVS-specific errors. For more information, see <a href="#"><u>Events</u></a> and <a href="#"><u>Errors</u></a> .

## currentTime

Gets or sets the time (in seconds from the beginning).

Amazon IVS: We do not recommend setting current time for live streams.

## Signatures

```
currentTime
currentTime(time)
```

## Parameter

Parameter	Type	Description
time	number	If time is absent, gets the current time. If time is present, sets video playback to that time.

## Return Value

Type	Description
number	The current time, in seconds from the beginning.

## dispose

Deletes the player instance.

Amazon IVS: This also deletes the Amazon IVS tech backend.

## Signature

```
dispose()
```

## Parameters

None

## Return Value

None

## duration

Returns the duration of the video, in seconds.

Amazon IVS: For live streams, this returns Infinity.

## Signature

```
duration()
```

## Parameters

None

## Return Value

Type	Description
number	The duration of the stream, in seconds. For live streams, this value is Infinity.

## getIVSEvents

Returns an object that holds Amazon IVS-specific enums. This is used for listening to Amazon IVS-specific errors and events. For more information, see:

- [Events](#) and [Errors](#) in this document.
- [Amazon IVS Player SDK: Web Reference](#) for more information about events, error types, and error sources.

## Signature

```
getIVSEvents()
```

## Parameters

None

## Return Value

Type	Description
object	An object with <code>PlayerEventType</code> , <code>PlayerState</code> , and <code>ErrorType</code> keys, which map to their associated enums.

## getIVSPlayer

Returns the underlying Amazon IVS player instance. The full Amazon IVS Player Web API is available through this instance. We recommend using the basic Video.js playback API as much as possible, and using this function only to access Amazon IVS-specific features. The most common functions you are likely to need to access on the Amazon IVS player instance are `setQuality()` and `addEventListener()` / `removeEventListener()`.

### Signature

```
getIVSPlayer()
```

### Parameters

None

### Return Value

Type	Description
MediaPlayer	The created instance of the player.

## load

Starts loading the `src()` data.

Amazon IVS: This is a no-op.

### Signature

```
load()
```

### Parameters

None



## Return Value

None

## play

Plays the stream that was set up via the `src` call.

Amazon IVS: If a live stream was paused, this plays the live stream from the latest frames, instead of continuing from where it was paused.

## Signature

```
play()
```

## Parameters

None

## Return Value

None

## playbackRate

Gets or sets the video-playback rate. 1.0 means normal speed; 0.5, half normal speed; 2.0, two times normal speed; and so on.

Amazon IVS: On a live stream, a get returns 1, and a set is ignored.

## Signatures

```
playbackRate  
playbackRate(rate)
```

## Parameter

Parameter	Type	Description
rate	number	The playback rate. Valid values: in the range [0.25, 2.0].

## Return Value

Type	Description
number	The playback rate.

## seekable

Returns the TimeRanges of the media that can be seeked to.

Amazon IVS: For live streams, calling `end(0)` on the return value (TimeRange) returns Infinity.

## Signature

```
seekable()
```

## Parameter

None

## Return Value

Type	Description
TimeRange	TimeRange of the media that is available for seeking to.

## Amazon IVS Player SDK: JW Player Integration

This document describes the most important functions available in the Amazon Interactive Video Service (IVS) JW Player integration.

**Latest version of JW Player integration:** 1.29.0 ([Release Notes](#))

## Getting Started

Amazon IVS support for JW Player is implemented through a Provider. Amazon IVS Provider is supported only on JW Player's web player. The Provider is loaded through a script tag, and any

streams requiring Amazon IVS Provider playback must be tagged with type: 'ivs' in the playlist. Amazon IVS supports JW Player version 8.18.4 and later.

## Setup

In these instructions, `JW_PLAYER_DIV` is the name of the `<div>` of your JW Player instance and `IVS_STREAM` is your IVS playback URL. To set up the Amazon IVS Provider and enable playback:

1. Include the following `script` tag (for the latest version of the player integration; in this case, 1.29.0):

```
<script src="https://player.live-video.net/1.29.0/amazon-ivs-jw-provider.min.js"></script>
```

2. Use the `ivs` type to mark your IVS playlist items. Set the `cast` value in your `setup()` to `null` (since Chromecast is not supported).

```
jwplayer(JW_PLAYER_DIV).setup({  
  playlist: [{  
    file: IVS_STREAM,  
    type: 'ivs',  
  }]  
});
```

3. If you want a reference to the underlying Amazon IVS Player to make Amazon IVS Player API calls or you want references to Amazon IVS-specific enums for callback handling, add a listener to the `'providerPlayer'` event:

```
jwplayer(JW_PLAYER_DIV).on('providerPlayer', function (player) {  
  // player object has 'ivsPlayer' and 'ivsEvents' properties  
  // ...callback code...  
});
```

## Sample Code

In this example, `JW_PLAYER_LIB` is the URL to your JW Player library script and `IVS_STREAM` is your IVS playback URL.

```
<!DOCTYPE html>  
<html lang="en">
```

```
<head>
  <script src=JW_PLAYER_LIB></script>
  <script src="https://player.live-video.net/1.29.0/amazon-ivs-jw-provider.min.js"></
script>
</head>
<body>
  <div id='player'></div>
  <script>
    // set default values for ivsPlayer and ivsEvents
    var ivsPlayer = {};
    var ivsEvents = {};

    // define our player setup
    const ivsConfig = {
      playlist: [{
        file: IVS_STREAM,
        type: 'ivs',
      }]
    };

    jwplayer('player')
      .setup(ivsConfig)
      .on('providerPlayer', function (player) {
        console.log('Amazon IVS Player: ', player.ivsPlayer);
        console.log('Amazon IVS Player Events: ', player.ivsEvents);

        // store the reference to the Amazon IVS Player
        ivsPlayer = player.ivsPlayer;
        // store the reference to the Amazon IVS Player Events
        ivsEvents = player.ivsEvents;
      });
  </script>
</body>
</html>
```

## Events

To listen to standard JW Player events, use the [on](#) function of the JW Player.

To listen to events that are specific to Amazon IVS, or to add and remove event listeners on the Amazon IVS Web player, you must listen to the 'providerPlayer' event to get a reference to the Amazon IVS Player and then add event listening onto it. For example:

```
// store a default value for ivsPlayer
var ivsPlayer = {};

// store references to the Amazon IVS Player and Amazon IVS Events:
jwplayer(JW_PLAYER_DIV).on('providerPlayer', function (player) {
    ivsPlayer = player.ivsPlayer;
});

// set up event listening
ivsPlayer.addEventListener(event, callback);
ivsPlayer.removeEventListener(event, callback);
```

where `callback` is a callback that you define, and `event` is one of: `PlayerEventType`, `PlayerState`, or `ErrorType`. For more information about events, see the [Amazon IVS Player SDK: Web Reference](#).

The 'providerPlayer' event is emitted by JW Player, and the callback you register with it will receive an object with the following fields:

Field	Description
<code>ivsPlayer</code>	Returns the underlying Amazon IVS player instance. The full Amazon IVS Player Web API is available through this instance. We recommend using the basic JW Player playback API as much as possible, and using this function only to access Amazon IVS-specific features. The most common functions you are likely to need to access on the Amazon IVS player instance are <code>addEventListener()</code> and <code>removeEventListener()</code> .
<code>ivsEvents</code>	Returns an object with <code>PlayerEventType</code> , <code>PlayerState</code> , and <code>ErrorType</code> fields, which map to their associated Amazon IVS-specific enums. For more information, see <a href="#">Amazon IVS Player SDK: Web Reference</a> .

## Errors

For general JW Player errors, use the [on](#) function of the JW Player to listen to error events.

For errors specific to Amazon IVS, listen on the Amazon IVS player for its own errors:

```
// set default values for ivsPlayer and ivsEvents
```

```
var ivsPlayer = {};  
var ivsEvents = {};  
  
// store references to the Amazon IVS Player and Amazon IVS Events  
jwplayer(JW_PLAYER_DIV).on('providerPlayer', function (player) {  
    ivsPlayer = player.ivsPlayer;  
    ivsEvents = player.ivsEvents;  
});  
  
// set up event listening:  
let playerEvent = ivsEvents.PlayerEventType;  
ivsPlayer.addEventListener(playerEvent.ERROR, callback);
```

The callback will receive an object with the following fields:

Field	Description
type	The error type. Corresponds to <code>ErrorType</code> events. For more information, see <a href="#">Amazon IVS Player SDK: Web Reference</a> .
code	The error code.
source	Source of the error.
message	Human readable error message.

## Content Security Policy

The Amazon IVS Provider API is configured to work on pages that use Content Security Policy (CSP). See the section on “Working with Content Security Policy” in the [Amazon IVS Player SDK: Web Guide](#).

## Limitations

The Provider does not support casting. If you enabled casting in the JW Player dashboard, you can disable it by setting `cast` to `null` when calling `setup()`. This hides the casting button.

# Embedding Metadata within a Video Stream

Amazon Interactive Video Service (IVS) timed metadata provides a way to embed metadata in an Amazon IVS stream. It ensures that all your viewers receive the metadata at the same time in the video stream, regardless of stream latency or geographic location.

## What is Timed Metadata?

*Timed* metadata is metadata with timestamps. It can be inserted into a stream programmatically, using the IVS API or the IVS broadcast SDK. When Amazon IVS processes a stream, the timed metadata is synchronized with the audio and video frames. During playback, all viewers of the stream get the metadata at the same time relative to the stream. The timecode serves as a cue point, which can be used to trigger an action based on the data, such as the following:

- Updating player statistics for a sports stream.
- Sending product details for a live shopping stream.
- Sending questions for a live quiz stream.

Amazon IVS timed metadata uses ID3 tags embedded in the video segments. As a result, they are available in the recorded video.

## Setting Up IAM Permissions

**Prerequisite:** Before proceeding, you should have stepped through [???](#) (including creating an IAM user and setting up permissions).

Next, you must give your IAM user permission to use timed metadata. Follow these steps:

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**, then choose the desired user (the user name you specified when you created an AWS account).
3. In the user **Summary** window, on the **Permissions** tab, choose **Add inline policy** (on the right side).
4. On the **JSON** tab, paste in this blob:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ivs:PutMetadata"
      ],
      "Resource": "arn:aws:ivs:*:*:channel/*"
    }
  ]
}
```

5. Still in the **Create Policy** window, choose **Review Policy**. Give the policy a **Name**, then choose **Create Policy**.
6. You're returned to the user **Summary** window, showing your new policy name.

## Inserting Timed Metadata

You can insert timed metadata only into an active stream on a specified channel.

### Using the AWS CLI

For testing, the easiest way to add timed metadata is with the AWS CLI. Using the AWS CLI requires that you first download and configure the CLI on your machine. You may have already done that when you stepped through [Getting Started with IVS](#); if not, do it now. For details, see the [AWS Command Line Interface User Guide](#).

Once you have the CLI:

1. Run the `put-metadata` command and pass in the channel ARN and your metadata:

```
aws ivs put-metadata --channel-arn <your-channel-arn> --metadata <your-metadata>
```

For example:

```
aws ivs put-metadata --channel-arn arn:aws:ivs:us-west-2:465369119046:channel/
GbiYJna5hFoC --metadata '{"question": "What does IVS stand for?", "correctIndex":
```



```
0, "answers": ["interactive video service", "interesting video service", "ingenious video service"]}]'
```

2. Amazon IVS checks whether the stream is live. If the stream is not live, you get an error; otherwise, the CLI returns without an error and the metadata (text blob) is inserted into the stream. This happens as soon as possible. There is no guarantee as to when this occurs; however, all viewers see the metadata at the same point in the stream.

## Using the Amazon IVS API

To programmatically insert timed metadata, use the [PutMetadata](#) API endpoint.

Here is an example HTTP request:

```
POST /PutMetadata HTTP/1.1
{
  "channelArn": "my_channel",
  "metadata": "{\"question\": \"What does IVS stand for?\", \"correctIndex\": 0, \"answers\": [\"interactive video service\", \"interesting video service\", \"ingenious video service\"]}"
}
```

## Using the IVS Broadcast SDK

You can insert timed metadata inband using the IVS broadcast SDK. This may be useful to synchronize the metadata with the audio and video content.

- Android — In the `BroadcastSession` class, use `sendTimedMetadata`.
- iOS — In the `IVSBroadcastSession` class, use `sendTimedMetadata`.

## Consuming Timed Metadata

Use the Amazon IVS Player to consume timed metadata embedded in a video stream. See [Amazon IVS Player SDK](#) and the rest of the Player documentation.

Below are example snippets that print any metadata received to the console using the Amazon IVS Player SDK. An event is triggered whenever playback reaches a segment with embedded metadata. (The event is `TEXT_METADATA_CUE` for Web, `onCue()` for Android, and `player(_:didOutputCue:)` for iOS.) You can use this event to initiate functionality within your

client application, such as updating an interactive widget. This event is triggered for both live and recorded content.

### Amazon IVS Player SDK for Web:

```
const player = IVSPlayer.create();
player.addEventListener(IVSPlayer.PlayerEventType.TEXT_METADATA_CUE,
  function (cue) {
    console.log('Timed metadata: ', cue.text);
  });
```

### Amazon IVS Player SDK for Android:

```
@Override
public void onCue(@NonNull Cue cue) {
    if(cue instanceof TextMetadataCue) {
        Log.i("Timed Metadata: ", ((TextMetadataCue)cue).text);
    }
}
```

### Amazon IVS Player SDK for iOS:

```
func player(_ player: IVSPlayer, didOutputCue cue: IVSCue) {
    if let textMetadataCue = cue as? IVSTextMetadataCue {
        print("Timed Metadata: \(textMetadataCue.text)")
    }
}
```

**Note:** Timed metadata is supported for iOS Safari and iOS Chrome in Player 1.3.0 and later.

## Sample Demo: Quiz App

Code samples of an interactive quiz app are available on GitHub. We use JSON via timed metadata to populate a quiz UI to display questions and answers. The answers are selectable and reveal whether the selection is correct.

Amazon IVS Player SDK Platform	Repo of Samples
Web	<a href="https://github.com/aws-samples/amazon-ivs-basic-web-sample">https://github.com/aws-samples/amazon-ivs-basic-web-sample</a>

Amazon IVS Player SDK Platform	Repo of Samples
	Within this repo, see the <a href="#">quiz demo</a> (and <a href="#">live demo</a> ).
Android	<a href="https://github.com/aws-samples/amazon-ivs-player-android-sample">https://github.com/aws-samples/amazon-ivs-player-android-sample</a> Within this repo, see the <a href="#">quiz demo</a> .
iOS	<a href="https://github.com/aws-samples/amazon-ivs-player-ios-sample">https://github.com/aws-samples/amazon-ivs-player-ios-sample</a> Within this repo, see the <a href="#">quiz demo</a> .

## Viewing Timed Metadata

If desired, you can view the timed metadata embedded in your live stream, in the console:

1. Open the [Amazon IVS console](#).
2. In the top left, choose the hamburger icon to open the navigation pane, then choose **Live channels**.
3. Choose the channel whose stream you want to view, to go to a details page for that channel.

The live stream is playing in the **Live stream** section of the page.

4. At the bottom of the window, choose **Timed Metadata**.

While the player is playing, as each timed-metadata event is received, its value and time received are displayed.

## For More Information

See [Using Amazon Interactive Video Service Timed Metadata](#), the first of a two-part blog series on using Amazon IVS timed metadata.

# Setting Up Private Channels

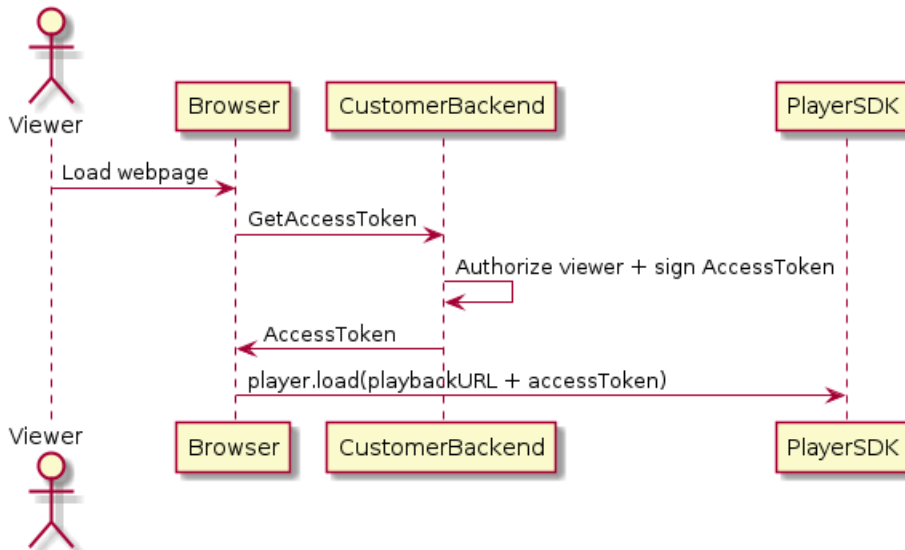
Amazon Interactive Video Service (IVS) offers customers the ability to create private channels, allowing customers to restrict their streams by channel or viewer. Customers control access to video playback by enabling *playback authorization* on channels and generating signed JSON Web Tokens (JWTs) for authorized playback requests.

Requiring playback authorization on a channel is optional. When a viewer tries to watch a stream, if the channel has authorization enabled, Amazon IVS verifies that the viewer has a valid playback token in the request. A playback token is a JWT that the Amazon IVS customer signs (with a playback authorization key) and includes with every playback request for a channel that has playback authorization enabled.

## Topics

- [Workflow for Private Channels](#)
- [Create or Import a Playback Key](#)
- [Enable Playback Authorization on Channels](#)
- [Generate and Sign Playback Tokens](#)
- [List Playback Keys](#)
- [Delete Playback Keys](#)
- [Get Information about Playback Keys](#)
- [Revoke Viewer Sessions](#)

## Workflow for Private Channels



1. When a viewer tries to load the webpage for a private stream, the browser requests an access token. (The customer provides the browser code to do this.)
2. The customer's backend app receives the access-token request and determines whether that viewer should be authorized to view the stream. If yes, the backend generates a JWT, uses the customer's private key to sign it, and returns the signed JWT in a playback request to the browser.
3. The browser loads the stream, using a request to the Amazon IVS player (or other player) SDK. The request contains the stream playback URL and the signed JWT.
4. Amazon IVS uses the customer's public key to verify that the JWT was signed using the correct private key.
5. If the JWT is verified, Amazon IVS plays the private stream for the viewer.

Customers are responsible for creating:

- The browser code to request access tokens.
- The backend server app that generates and signs JWTs.
- A playback authorization key pair. This has two parts: a public key that AWS retains and a private key that you download. With the private key, you sign the JWTs that authorize access to your private channel.

The method described above — using a network request from the browser to fetch tokens — is not the only way to implement playback authorization. Alternately, customers could send the signed playback tokens in the initial webpage, to reduce the number of network round trips that a viewer needs to make.

In the sections below, we describe how to make a channel private (enable playback authorization), generate and sign playback tokens, and work with playback key pairs.

**Note:** In the console instructions below, if the left navigation menu is not displaying, you can open it by choosing the hamburger icon in the top left.

## Create or Import a Playback Key

Amazon IVS allows a maximum of three key pairs that can be used to sign and verify playback tokens. Amazon IVS does not offer any key rotations.

**Once imported, playback keys cannot be updated.** Instead, you must delete the existing playback key and import a new key.

You need to generate an [ECDSA public/private key pair](#) to sign the JWTs and upload the public key to Amazon IVS as a playback-key resource. Then Amazon IVS can verify the signature in playback requests.

### Creating a New Key Pair

There are various ways to create a key pair; below, we give two examples.

To create a new key pair in the console, follow these steps. Note this process enables you to download only the private key.

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the left navigation menu, choose **Playback security > Playback keys**.
3. Choose **Create playback key**. A **Create playback key** dialog appears.
4. Follow the prompts and choose **Create**.
5. Amazon IVS generates a new key pair. The public key is imported as a playback key resource and the private key is immediately made available for download.

Amazon IVS generates the key on the client side and does not store the private key. **Be sure you save the key; you cannot retrieve it later.:**

To create a new P384 EC key pair with OpenSSL (you may have to install [OpenSSL](#) first), follow these steps. This process enables you to access both the private and public keys. You need the public key only if you want to test verification of your tokens.

```
openssl ecparam -name secp384r1 -genkey -noout -out priv.pem
openssl ec -in priv.pem -pubout -out public.pem
```

Now import your new public key, using the instructions below.

## Importing an Existing Public Key

If you already have a key pair, you can import the public key into IVS. The private key is not needed by our system but is employed by you to sign tokens.

To import an existing public key with the console:

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the left navigation menu, choose **Playback security > Playback keys**.
3. Choose **Import**. An **Import playback key** dialog appears.
4. Follow the prompts and choose **Import**.
5. Amazon IVS imports your public key and generates a playback key resource.

To import an existing public key with the CLI:

```
aws ivs import-playback-key-pair --public-key-material "`cat public.pem`" --region
<aws-region>
```

You can omit `--region <aws-region>` if the region is in your local AWS configuration file.

Here is an example response:

```
{
  "keyPair": {
    "arn": "arn:aws:ivs:us-west-2:693991300569:playback-key/f99cde61-
c2b0-4df3-8941-ca7d38acca1a",
    "fingerprint": "98:0d:1a:a0:19:96:1e:ea:0a:0a:2c:9a:42:19:2b:e7",
    "tags": {}
  }
}
```

## API Request

```
POST /ImportPlaybackKeyPair HTTP/1.1
{
  "publicKeyMaterial": "<pem file contents>"
}
```

## Enable Playback Authorization on Channels

A channel's authorization requirement can be configured when the channel is created or later (using an update endpoint). Note that the steps are the same whether you want to enable or disable playback authorization.

### Console Instructions

To enable authorization when creating a channel:

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the **Get started** box (top right), choose **Create channel**.
3. On the **Channel create** page, choose **Custom configuration**.
4. In the **Playback authentication** section, turn on **Enable token-authentication requirement for video playback**.
5. Follow the rest of the prompts to create a channel. (See [Getting Started with IVS](#).)

To enable authorization by updating an existing channel:

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the left navigation menu, choose **Channels**.
3. Choose the checkbox for the channel you want to update, then choose **Edit**.
4. In the **Playback authentication** section, turn on **Enable token-authentication requirement for video playback**.
5. Click **Save changes**.

### CLI Instructions

To enable authorization when creating a channel:



```
aws ivs create-channel --authorized --region <aws-region>
```

You can omit `--region <aws-region>` if the region is in your local AWS configuration file.

Here is an example response. Note that `authorized` is `true`.

```
{
  "streamKey": {
    "channelArn": "arn:aws:ivs:us-west-2:123456789:channel/fbc789c1-2c56-4ce6-a30a-d99275dc4481",
    "value": "sk_us-west-2_abcd1234efgh5678ijkl",
    "arn": "arn:aws:ivs:us-west-2:123456789:stream-key/62f15f1b-fe31-4127-b252-0666ac7f55a7",
    "tags": {}
  },
  "channel": {
    "name": "test-channel",
    "tags": {},
    "authorized": true,
    "latencyMode": "LOW",
    "ingestEndpoint": "jds34ksdg3las.global-contribute.live-video.net",
    "playbackUrl": "https://b37c565f6d79.us-west-2.playback.live-video.net/api/video/v1/aws.ivs.us-west-2.123456789.channel.oU40KS4LA1Dz.m3u8",
    "arn": "arn:aws:ivs:us-west-2:123456789:channel/fbc789c1-2c56-4ce6-a30a-d99275dc4481"
  }
}
```

To enable authorization by updating an existing channel:

```
aws ivs update-channel --arn
arn:aws:ivs:us-west-2:693991300569:channel/742da049-fe9f-4f23-928e-c6753760a189
--authorized
```

This is just an example; you must specify your own channel ARN after `--arn`. As when creating a channel, `authorized` is `true` in the update response.

## API Request (Create and Update)

```
POST /CreateChannel HTTP/1.1
{
```

```
"name": "<your channel name>",
"authorized": true
}
```

```
POST /UpdateChannel HTTP/1.1
{
  "arn": "<channel arn>",
  "authorized": true
}
```

## Generate and Sign Playback Tokens

For details on working with JWTs and the supported libraries for signing tokens, visit [jwt.io](https://jwt.io). On the [jwt.io](https://jwt.io) interface, you must enter your private key to sign tokens. The public key is needed only if you want to verify tokens.

### Token Schema

All JWTs have three fields: header, payload, and signature.

- The **header** specifies:
  - `alg` is the signing algorithm. This is ES384, an ECDSA signature algorithm that uses the SHA-384 hash algorithm.
  - `typ` is the token type, JWT.

```
{
  "alg": "ES384",
  "typ": "JWT"
}
```

- The **payload** contains data specific to Amazon IVS:
  - `channel-arn` is a reference for the video-playback request.
  - `access-control-allow-origin` is an optional field that can be used to restrict playback to a specified [origin](#); i.e., to make a stream viewable from only a specified website. For example, you may want to prevent people from embedding the player on other websites. By default, playback is allowed on all origins. (Note that this restricts only the browser client; it does not restrict playback from a non-browser client.) This field may contain multiple origins, separated by commas. Wildcard domains are allowed: each origin may begin its hostname

with `*` (example: `https://*.amazon.com`). If `strict-origin-enforcement` is `true`, at most 5 domains may be specified; otherwise, there is no maximum.

- `strict-origin-enforcement` is an optional field that can be used to strengthen the origin restriction specified in the `access-control-allow-origin` field. By default, the `access-control-allow-origin` restriction applies only to the multivariant playlist. If `strict-origin-enforcement` is enabled, the server will enforce a requirement that the requesting origin matches the token for all playback requests (including multivariant playlist, variant playlist, and segments). This means that all clients (including non-browser clients) will have to provide a valid `origin-request` header with each request. Use the `setOrigin` method to set the header in the IVS iOS and Android player SDKs. It is set automatically in web browsers except iOS Safari. For iOS Safari, you need to add `crossorigin="anonymous"` to the video element, to ensure that the origin request header is sent. Example: `<video crossorigin="anonymous"></video>`.
- `single-use-uuid` is an optional field which contains a valid [universally unique identifier \(UUID\)](#) that you generate as part of authoring the token. If you add this field and a UUID value, the associated token that you generate is invalidated once it is used to fetch a multivariant playlist and watch a stream. Single-use auth tokens make it more difficult for malicious users to share a stream on your private channels with other viewers. Note that when using the `single-use-uuid` claim, the maximum value for the `exp` claim is 10 minutes in the future.
- `viewer-id` is an optional field which contains an ID used for tracking and referring to the viewer to whom the token is granted. This field is required to enable the ability to revoke the viewing session of the viewer in the future. The maximum length is 40 characters, and the value must qualify as a string. Do not use this field for personally identifying, confidential, or sensitive information. Note that when using `viewer-id`, the maximum value for `exp` is 10 minutes in the future.
- `viewer-session-version` is an optional field which contains a version to associate with this viewer session. When revoking viewer sessions, this value can be used to filter which viewer sessions are revoked. For example, specifying a Unix timestamp here would enable revocation of all sessions started before the specified time. The value must be a 64-bit signed integer (Int64). This field is meant to be provided (optionally) alongside `viewer-id`; it does nothing on its own. The default value is 0.
- `exp` is a Unix UTC timestamp for when the token expires. This does not indicate the length of time that the stream can be viewed. The token is validated when the viewer initializes playback, not throughout the stream. Enter this value as an integer type value.

Note that a Unix timestamp is a numeric value representing the number of seconds from 1970-01-01T00:00:00Z UTC until the specified UTC date/time, ignoring leap seconds. Different languages measure Unix timestamps in different units; e.g., JavaScript's `Date.now()` returns the time in milliseconds. (See `exp` in the [JWT RFC section 4.1.4.](#))

```
{
  "aws:channel-arn": "<channel_arn>",
  "aws:access-control-allow-origin": "<your-origin>",
  "aws:strict-origin-enforcement": true,
  "aws:single-use-uuid": "<UUID>",
  "aws:viewer-id": "<viewer_id>",
  "aws:viewer-session-version": "<viewer_session_version>",
  "exp": <unix timestamp>
}
```

- To create the **signature**, use the private key with the algorithm specified in the header (ES384) to sign the encoded header and encoded payload.

```
ECDSASHA384(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  <private-key>
)
```

## Instructions

1. Generate the token's signature with the ES384 signing algorithm and a private key that is associated with one of your playback-key resources (see the ECDSASHA384 example above).
2. Assemble the token.

```
base64UrlEncode(header) + "." +
base64UrlEncode(payload) + "." +
base64UrlEncode(signature)
```

3. Append the signed token to the playback URL as a query parameter.

```
https://b37c565f6d790a14a0e78afaa6808a80.us-west-2.playback.live-video.net/
api/video/v1/aws.ivs.us-west-2.123456789.
```

```
channel.fbc789c1-2c56-4ce6-a30a-d99275dc4481.m3u8?token=<token>
```

## List Playback Keys

Amazon IVS customers can get a list of all of their playback-key resources at any time.

### Console Instructions

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the left navigation menu, choose **Playback security > Playback keys**.

All playback-key resources associated with your account are displayed. Deleted keys are not displayed, and there is no history of past keys.

### CLI Instructions

```
aws ivs list-playback-key-pairs --region <aws-region>
```

You can omit `--region <aws-region>` if the region is in your local AWS configuration file.

Example response:

```
{
  "keyPairs": [
    {
      "arn": "arn:aws:ivs:us-west-2:991729659840:playback-key/3db9fc15-df57-4c02-b5a6-d4ee3448b8ad",
      "fingerprint": "81:f3:8c:88:78:61:4e:bc:58:07:a3:ca:63:f5:72:08",
      "tags": {}
    },
    {
      "arn": "arn:aws:ivs:us-west-2:991729659840:playback-key/3ff88c71-b18e-415f-948b-18bbde605a97",
      "fingerprint": "a2:b5:b3:0b:be:8e:73:00:0e:ad:e9:bb:02:c9:81:9a",
      "tags": {}
    }
  ]
}
```

## API Request

For usage information, see [ListPlaybackKeyPairs](#) in the *IVS Low-Latency Streaming API Reference*.

```
POST /ListPlaybackKeyPairs HTTP/1.1
{
  "maxResults": number,
  "nextToken": "string"
}
```

## Delete Playback Keys

Amazon IVS customers can delete playback keys from their accounts. Deleted keys will remove the resource from the customer's account; playback tokens signed with deleted keys will not pass verification.

## Console Instructions

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the left navigation menu, choose **Playback security > Playback keys**.
3. Choose the key(s) you want to delete.
4. Choose **Delete**. A **Delete playback key** dialog appears.
5. Choose **Delete playback key**.

## CLI Instructions

You can delete playback keys via the AWS CLI, if you have the key's ARN. Amazon IVS does not support batch deletes via the CLI.

```
aws ivs delete-playback-key-pair --arn arn:aws:ivs:us-west-2:991729659840:playback-key/3db9fc15-df57-4c02-b5a6-d4ee3448b8ad --region <aws-region>
```

You can omit `--region <aws-region>` if the region is in your local AWS configuration file.

On success, there is no response. You can run the `get` command (below) to verify that the key was deleted.

Here is an example error response:

```
An error occurred (ResourceNotFoundException) when calling the
DeletePlaybackKeyPair operation: ResourceNotFoundException:
```

## API Request

```
POST /DeletePlaybackKeyPair HTTP/1.1
{
  "arn": "<playback key arn>"
}
```

## Get Information about Playback Keys

Amazon IVS customers can get information about their playback key resources. It is important to note that the associated private key will not be available, even in the case that the playback key was created by Amazon IVS via the console.

## Console Instructions

1. Open the [Amazon IVS console](#). Choose your channel's region if you are not already on it.
2. In the left navigation menu, choose **Playback security** > **Playback keys**.
3. Choose the key you want to get more details about and choose **View details**.

## CLI Instructions

```
aws ivs get-playback-key-pair --arn arn:aws:ivs:us-west-2:991729659840:playback-
key/3db9fc15-df57-4c02-b5a6-d4ee3448b8ad --region <aws-region>
```

You can omit `--region <aws-region>` if the region is in your local AWS configuration file.

Example response:

```
{
  "keyPair": {
    "arn": "arn:aws:ivs:us-west-2:991729659840:playback-key/3ff88c71-
b18e-415f-948b-18bbde605a97",
    "fingerprint": "a2:b5:b3:0b:be:8e:73:00:0e:ad:e9:bb:02:c9:81:9a",
```

```
    "tags": {}  
  }  
}
```

## API Request

```
POST /GetPlaybackKeyPair HTTP/1.1  
{  
  "arn": "<playback key arn>"  
}
```

## Revoke Viewer Sessions

Amazon IVS customers can revoke the viewer session associated with an auth token, to prevent and stop playback using that token. An example use case is transitioning a public stream to a private stream in which only a subset of the public stream viewers can continue watching.

For information on the `viewer-id` field mentioned in the instructions below, see the "Token Schema" under [the section called "Generate and Sign Playback Tokens"](#).

## CLI Instructions

You can revoke the viewer session via the AWS CLI, if you have the channel ARN and the viewer ID.

```
aws ivs start-viewer-session-revocation --channel-arn arn:aws:ivs:us-  
west-2:991729659840:channel/abcdABCDefgh --viewer-id UDbh1u6M8nr0oarrzuKe --region  
<aws-region>
```

An optional input, `--viewer-session-versions-less-than-or-equal-to <version>` lets you specify a filter for which versions of the viewer session to revoke at once.

You can omit `--region <aws-region>` if the region is in your local AWS configuration file.

On success, there is no response.

Here is an example error response:

```
An error occurred (ValidationException) when calling the StartViewerSessionRevocation  
operation: ValidationException:
```



## API Request

```
POST /StartViewerSessionRevocation HTTP/1.1
{
  "channelArn": <channel ARN>,
  "viewerId": <viewer ID>,
  "viewerSessionVersionsLessThanOrEqualTo": <version>
}
```

There also is a `BatchStartViewerSessionRevocation` endpoint. See the [IVS Low-Latency Streaming API Reference](#).

# Auto-Record to Amazon S3 (Low-Latency Streaming)

This section provides information about the auto-record-to-S3 feature of Amazon IVS low-latency streaming. We discuss data storage for recorded Amazon IVS streams. We explain the storage contents and metadata file schema. We also discuss playback of your recorded content.

For details on ...	See ...
Setting up and stopping video recording	<a href="#">Create a Channel with Optional Recording</a> in <i>Getting Started with Amazon IVS</i>
The API	<a href="#">IVS API Reference</a>
Costs	<a href="#">Amazon IVS Costs</a>

## S3 Prefix

The S3 prefix is a unique directory structure for each live stream that is recorded. All media and metadata files for the live stream are written within this directory. For channels with recording enabled, the S3 prefix is generated when a live session starts and will be provided in the CloudWatch event at the start and end of a recording.

The S3 prefix has the following format:

```
/ivs/v1/<aws_account_id>/<channel_id>/<year>/<month>/<day>/<hours>/<minutes>/<recording_id>
```

Where:

- `aws_account_id` is the ID of your AWS account (generated when you created an AWS account), from which the channel is created.
- `channel_id` is the resource ID part of the channel ARN (the last part of the Amazon Resource Name). See ARN in the [Glossary](#).
- `<year>/<month>/<day>/<hours>/<minutes>` is a UTC timestamp when recording starts.
- `recording_id` is a unique ID generated for each recording session.

For example:

```
ivs/v1/123456789012/AsXego4U6tnj/2020/6/23/20/12/j8Z9091ndcVs
```

## Recording Contents

When recording starts, video segments and metadata files are written to the S3 bucket that is configured for the channel. These contents are available for post-processing or playback as on-demand video.

Note that after a live stream starts and the Recording Start EventBridge event is emitted, it takes a little time before the manifest files and video segments are written. We recommend that you play back or process recorded streams only after the Recording End event is sent. (See [Using Amazon EventBridge with IVS](#).)

The following is a sample directory structure and contents of a recording of a live Amazon IVS session:

```
ivs/v1/123456789012/AsXego4U6tnj/2020/6/23/20/12/j8Z9091ndcVs/  
  events  
    recording-started.json  
    recording-ended.json  
  media  
    hls  
    thumbnails
```

The events folder contains the metadata files corresponding to the recording event. JSON metadata files are generated when recording starts, ends successfully, or ends with failures:

- events/recording-started.json
- events/recording-ended.json
- events/recording-failed.json

A given events folder will contain recording-started.json and either recording-ended.json or recording-failed.json.

These contain metadata related to the recorded session and its output formats. JSON details are given below.

The `media` folder contains all supported media contents, in two subfolders:

- `hls` contains all media and manifest files generated during the live session and is playable with the Amazon IVS player. There are two types of HLS manifests in this folder, the standard master manifest `master.m3u8` and the byte-range enabled manifest `byte-range-multivariant.m3u8`. Therefore, each rendition folder has both `playlist.m3u8` and a `byte-range-variant.m3u8` files. (See [Byte-Range Playlists](#) below.)
- `thumbnails` contains thumbnail images generated during the live session. Thumbnails are generated and written to the bucket every minute. (To change this behavior, override the `thumbnailConfiguration` property on a recording configuration.)

**Important:** The contents within the `media` folder are dynamically generated and determined by the characteristics of the first received video segments; the folder contents may not represent the ultimate characteristics (e.g., rendition quality). *Do not make any assumptions about the static path.* To discover the HLS renditions available and its path, use the JSON metadata files described below.

## Byte-Range Playlists

The auto-record-to-S3 feature supports [byte-range playlist](#) generation, in addition to standard HLS playlists. Byte-range playlists conform to version 4 of the HLS specification. This allows for more fine-grained content clipping: in a byte-range playlist, each segment in a rendition index file references a subrange of bytes of a video chunk, providing more granularity than the standard 10-second media file size. With a byte-range playlist, the segment duration is the same as the keyframe interval configured for the stream.

## Thumbnails

The `thumbnailConfiguration` property on a recording configuration allows you to enable or disable the recording of thumbnails for a live session and modify the interval at which thumbnails are generated for the live session. Thumbnail intervals may range from 1 second to 60 seconds; by default, thumbnail recording is enabled, at an interval of 60 seconds. For details, see the [Amazon IVS API Reference](#).

Thumbnail configuration also may include the `storage` field (SEQUENTIAL and/or LATEST) and a resolution (LOWEST\_RESOLUTION, SD, HD, or FULL\_HD). Below are the resolutions for each option:

160 <= LOWEST\_RESOLUTION <= 360

360 < SD <= 480

480 < HD <= 720

720 < FULL\_HD <= 1080

## Merge Fragmented Streams

The `recordingReconnectWindowSeconds` property on a recording configuration allows you to specify a window of time (in seconds) during which, if your stream is interrupted and a new stream is started, Amazon IVS tries to record to the same S3 prefix as the previous stream. In other words, if a broadcast disconnects and then reconnects within the specified interval, the multiple streams are considered a single broadcast and merged together.

**IVS Recording State Change events in Amazon EventBridge:** Recording End events and *recording-ended* JSON metadata files are delayed by at least `recordingReconnectWindowSeconds`, as Amazon IVS waits to ensure a new stream is not started.

For instructions on setting up the merge-streams functionality, see [Step 4: Create a Channel with Optional Recording](#) in *Getting Started with Amazon IVS*.

## Eligibility

For multiple streams to record to the same S3 prefix, certain conditions must be met for all the streams:

- Video width and height must be the same.
- Frame rate must be the same.
- The bitrate difference of subsequent streams must be less than or equal to 50% of the bitrate of the original stream.
- Video and audio codecs must be the same.

### Notes:

- At most 20 streams are merged, after which a new S3 prefix is created.
- After 48 hours, a new S3 prefix is created. For example, if the first broadcast lasts for 48 hours and another broadcast is started within the `recordingReconnectWindowSeconds` interval, the next broadcast *is not* merged into the first S3 prefix.

- Each stream must start 10 seconds or more after the previous stream.

## Known Issue

If `recordingReconnectWindowSeconds` is enabled and the Web Broadcast SDK is used, recording to the same S3 prefix may not work, as the Web Broadcast SDK dynamically changes bitrates and qualities.

## JSON Metadata Files

When a recording state-change event occurs, a corresponding Amazon CloudWatch metric is generated and a metadata file is written within the S3 prefix. (See [Monitoring Amazon IVS Low-Latency Streaming](#).)

This metadata is in JSON format. It comprises the following information.

Field	Type	Required	Description
<code>channel_arn</code>	string	Yes	ARN of the channel broadcasting the live stream.
<code>media</code>	object	Yes	Object that contains the enumerated objects of media content available for this recording. Valid values: "hls", "thumbnails" .
<code>hls</code>	object	Yes	Enumerated field that describes the Apple HLS format output.
<code>duration_ms</code>	integer	Condition al	Duration of the recorded HLS content in milliseconds. This is available only when <code>recording_status</code> is "RECORDING_ENDED" or "RECORDING_ENDED_WITH_FAILURE" . If a failure occurred before any recording was done, this is 0.

Field	Type	Required	Description
path	string	Yes	Relative path from the S3 prefix where HLS content is stored.
playlist	string	Yes	Name of the HLS master playlist file.
byte_range_playlist	string	Yes	Name of the HLS byte-range multivariant playlist.
renditions	object	Yes	Array of renditions (HLS variant) of metadata objects. There always is at least one rendition.
path	string	Yes	Relative path from the S3 prefix where HLS content is stored for this rendition.
playlist	string	Yes	Name of the media playlist file for this rendition.
byte_range_playlist	string	Yes	Name of the byte-range playlist for this rendition.
resolution_height	int	Conditional	Pixel resolution height of the encoded video. This is available only when the rendition contains a video track.
resolution_width	int	Conditional	Pixel resolution width of the encoded video. This is available only when the rendition contains a video track.

Field	Type	Required	Description
<code>thumbnails</code>	object	Conditional	Enumerated field that describes thumbnails output. This is available only when the thumbnail configuration's <code>recordingMode</code> is <code>INTERVAL</code> .
<code>path</code>	string	Conditional	Relative path from the S3 prefix where thumbnail content is stored. This is available only when the thumbnail configuration's <code>recordingMode</code> is <code>INTERVAL</code> .
<code>resolution_height</code>	int	Yes	The height of the thumbnail. Default: resolution of the source rendition. This value is affected by user input in the related recording configuration; specifically, the <code>thumbnailConfiguration.resolution</code> value.
<code>resolution_width</code>	int	Yes	The width of the thumbnail. Default: resolution of the source rendition. This value is affected by user input in the related recording configuration; specifically, the <code>thumbnailConfiguration.resolution</code> value.
<code>latest_thumbnail</code>	object	Yes	Enumerated field that describes latest thumbnail output. This is available only when the thumbnail configuration's storage includes <code>LATEST</code> .



Field	Type	Required	Description
<code>resolution_height</code>	int	Yes	The height of the thumbnail . Default will be the resolution of the source rendition. This value is affected by user input in the related recording configuration; specifically, the <code>thumbnailConfiguration.resolution</code> value.
<code>resolution_width</code>	int	Yes	The width of the thumbnail . Default will be the resolution of the source rendition. This value is affected by user input in the related recording configuration; specifically, the <code>thumbnailConfiguration.resolution</code> value.
<code>recording_ended_at</code>	string	Conditional	RFC 3339 UTC timestamp when the recording ended. This is available only when <code>recording_status</code> is "RECORDING_ENDED" or "RECORDING_ENDED_WITH_FAILURE" .  <code>recording_started_at</code> and <code>recording_ended_at</code> are timestamps when these events are generated and may not exactly match the HLS video-segment timestamps. To accurately determine the duration of a recording, use the <code>duration_ms</code> field.

Field	Type	Required	Description
recording_started_at	string	Yes	RFC 3339 UTC timestamp when the recording started.  See the note above for recording_ended_at .
recording_status	string	Yes	Status of the recording. Valid values: "RECORDING_STARTED" , "RECORDING_ENDED" , "RECORDING_ENDED_WITH_FAILURE" .
recording_status_message	string	Conditional	Descriptive information on the status. This is available only when recording_status is "RECORDING_ENDED" or "RECORDING_ENDED_WITH_FAILURE" .
version	string	Yes	The version of the metadata schema.

## Example: recording\_started.json

```
{
  "version": "v1",
  "channel_arn": "arn:aws:ivs:us-west-2:123456789012:channel/AsXego4U6tnj",
  "recording_started_at": "2020-06-12T12:53:26Z",
  "recording_status": "RECORDING_STARTED",
  "media": {
    "hls": {
      "path": "media/hls",
      "playlist": "master.m3u8",
      "byte_range_playlist": "byte-range-multivariant.m3u8",
      "renditions": [
        {
          "path": "480p30",
```

```
    "playlist": "playlist.m3u8",
    "byte_range_playlist": "byte-range-variant.m3u8",
    "resolution_height": 480,
    "resolution_width": 852
  },
  {
    "path": "360p30",
    "playlist": "playlist.m3u8",
    "byte_range_playlist": "byte-range-variant.m3u8",
    "resolution_height": 360,
    "resolution_width": 640
  },
  {
    "path": "160p30",
    "playlist": "playlist.m3u8",
    "byte_range_playlist": "byte-range-variant.m3u8",
    "resolution_height": 160,
    "resolution_width": 284
  },
  {
    "path": "720p60",
    "playlist": "playlist.m3u8",
    "byte_range_playlist": "byte-range-variant.m3u8",
    "resolution_height": 720,
    "resolution_width": 1280
  }
]
},
"thumbnails": {
  "path": "media/thumbnails",
  "resolution_height": 480,
  "resolution_width": 852
},
"latest_thumbnail": {
  "path": "media/latest_thumbnail/thumb.jpg",
  "resolution_height": 480,
  "resolution_width": 852
}
}
```

## Example: recording\_ended.json

```
{
  "version": "v1",
  "channel_arn": "arn:aws:ivs:us-west-2:123456789012:channel/AsXego4U6tnj",
  "recording_ended_at": "2020-06-14T12:53:20Z",
  "recording_started_at": "2020-06-12T12:53:26Z",
  "recording_status": "RECORDING_ENDED",
  "media": {
    "hls": {
      "duration_ms": 172794489,
      "path": "media/hls",
      "playlist": "master.m3u8",
      "byte_range_playlist": "byte-range-multivariant.m3u8",
      "renditions": [
        {
          "path": "480p30",
          "playlist": "playlist.m3u8",
          "byte_range_playlist": "byte-range-variant.m3u8",
          "resolution_height": 480,
          "resolution_width": 852
        },
        {
          "path": "360p30",
          "playlist": "playlist.m3u8",
          "byte_range_playlist": "byte-range-variant.m3u8",
          "resolution_height": 360,
          "resolution_width": 640
        },
        {
          "path": "160p30",
          "playlist": "playlist.m3u8",
          "byte_range_playlist": "byte-range-variant.m3u8",
          "resolution_height": 160,
          "resolution_width": 284
        },
        {
          "path": "720p60",
          "playlist": "playlist.m3u8",
          "byte_range_playlist": "byte-range-variant.m3u8",
          "resolution_height": 720,
          "resolution_width": 1280
        }
      ]
    }
  }
}
```

```
    ]
  },
  "thumbnails": {
    "path": "media/thumbnails",
    "resolution_height": 480,
    "resolution_width": 852
  },
  "latest_thumbnail": {
    "path": "media/latest_thumbnail/thumb.jpg",
    "resolution_height": 480,
    "resolution_width": 852
  }
}
```

## Example: recording\_failed.json

```
{
  "version": "v1",
  "channel_arn": "arn:aws:ivs:us-west-2:123456789012:channel/AsXego4U6tnj",
  "recording_ended_at": "2020-06-14T12:53:20Z",
  "recording_started_at": "2020-06-12T12:53:26Z",
  "recording_status": "RECORDING_ENDED_WITH_FAILURE",
  "recording_status_message": "InternalServerException",
  "media": {
    "hls": {
      "duration_ms": 172794489,
      "path": "media/hls",
      "playlist": "master.m3u8",
      "renditions": [
        {
          "path": "480p30",
          "playlist": "playlist.m3u8",
          "resolution_height": 480,
          "resolution_width": 852
        },
        {
          "path": "720p60",
          "playlist": "playlist.m3u8",
          "resolution_height": 720,
          "resolution_width": 1280
        }
      ]
    }
  }
}
```

```
    },
    "thumbnails": {
      "path": "media/thumbnails",
      "resolution_height": 480,
      "resolution_width": 852
    },
    "latest_thumbnail": {
      "path": "media/latest_thumbnail/thumb.jpg",
      "resolution_height": 480,
      "resolution_width": 852
    }
  }
}
```

## Discovering the Renditions of a Recording

When you stream content to an Amazon IVS channel, `auto-record-to-s3` uses the source video to generate multiple renditions. Using [Adaptive Bitrate Streaming](#) (ABR), the Amazon IVS Player automatically switches the renditions (bitrates) as needed to optimize playback for varying network conditions.

Each rendition generated during live streaming is recorded in a unique path within the S3 recording prefix. The resolution detail, path, and playlist file names are stored in a [JSON metadata file](#) during the start and stop of the recording. If the recording configuration's `renditionSelection` value is `ALL`, all renditions are selected for recording. If `renditionSelection` is `CUSTOM`, the user must select one or more of the following options: `LOWEST_RESOLUTION`, `SD`, `HD`, and `FULL_HD`. Below are the resolutions for each option:

`160 <= LOWEST_RESOLUTION <= 360`

`360 < SD <= 480`

`480 < HD <= 720`

`720 < FULL_HD <= 1080`

**Important:** *Do not* make any assumptions about the static rendition path or the list of generated renditions, as these are subject to change. *Do not* assume that a specific rendition will always be available for an Amazon IVS recording. To determine the available renditions, resolutions, and paths, refer to the metadata files.

The `event/recording_started.json` or `event/recording_ended.json` file within the recording prefix contains the paths and names of media files within the recording prefix. All path elements are relative to the previous path in the hierarchy. Elements under `media > hls` describe HLS assets, with master playlist name and path defined at this level.

Here is a Python code snippet that shows how to generate a master playlist path using the S3 recording prefix and metadata file:

```
def get_master_playlist(metadata_json, s3_recording_prefix):
    return s3_recording_prefix + '/' + metadata_json['media']['hls']['path'] + '/' +
        metadata_json['media']['hls']['playlist']
```

Elements under `media > hls > renditions` describe the list of renditions recorded. The `resolution_height` and `resolution_width` properties can be used to identify the video resolution. The `path` and `playlist` elements can be used to derive the rendition playlist path. Use these fields to determine which rendition to use for any post processing.

To discover the highest available rendition playlist for a recording, you can subscribe to "IVS Recording State Change" EventBridge events. (See [Using Amazon EventBridge with IVS.](#)) Below is a sample Python script that illustrates using a lambda function subscribed to those events.

```
import json
import boto3
s3 = boto3.resource('s3')

def get_highest_rendition_playlist(bucket_name, prefix_name):
    object_path = "{}/events/recording-started.json".format(prefix_name)
    object = s3.Object(bucket_name, object_path)
    body = str(object.get()['Body'].read().decode('utf-8'))
    metadata = json.loads(body)
    media_path = metadata["media"]["hls"]["path"]
    renditions = metadata["media"]["hls"]["renditions"]

    highest_rendition = None
    highest_rendition_size = 0

    for rendition in renditions:
        current_rendition_size = rendition["resolution_height"]
        if (current_rendition_size > highest_rendition_size):
            highest_rendition_size = current_rendition_size
            highest_rendition = rendition
```

```
highest_rendition_playlist = media_path + '/' + highest_rendition['path'] + '/' +
highest_rendition['playlist']
return highest_rendition_playlist

def lambda_handler(event, context):
    prefix_name = event["detail"]["recording_s3_key_prefix"]
    bucket_name = event["detail"]["recording_s3_bucket_name"]
    rendition_playlist = get_highest_rendition_playlist(bucket_name, prefix_name)
    print("Highest rendition playlist: {}/{}".format(prefix_name, rendition_playlist))

    return {
        'statusCode': 200,
        'body': rendition_playlist
    }
```

## Playback of Recorded Content from Private Buckets

Objects recorded with the Auto-Record to Amazon S3 feature are private by default; hence, these objects are inaccessible for playback using the direct S3 URL. If you try to open the HLS master manifest (m3u8 file) for playback using the Amazon IVS player or another player, you will get an error (e.g., "You do not have permission to access the requested resource"). Instead, you can play back these files with the Amazon CloudFront CDN (Content Delivery Network).

### Amazon CloudFront Distribution

CloudFront distributions can be configured to serve content from private buckets. Typically this is preferable to having openly accessible buckets where reads bypass the controls offered by CloudFront. Your distribution can be set up to service from a private bucket by creating an origin access control (OAC), which is a special CloudFront user that has read permissions on the private origin bucket. You can create the OAC after you create your distribution, through the CloudFront console or API. See [Creating a new origin access control](#).

### Playback from Amazon CloudFront

Once you have set up your distribution using an OAC to gain access to your private bucket, your video files should be available for consumption through the CloudFront URL. Your CloudFront URL is the **Distribution domain name** on the **Details** tab in the AWS CloudFront console. It should be something like this:



a1b23cdef4ghij.cloudfront.net.

To stream your recorded video through your distribution, find the object key for your `master.m3u8` file. It should be something like this:

```
ivs/v1/012345678912/a0bCDeFGH1IjK/2021/4/20/12/03/aBcdEFghIjkl/media/hls/master.m3u8
```

Append the object key to the end of your CloudFront URL. Your final URL will be something like this:

```
https://a1b23cdef4ghij.cloudfront.net/ivs/v1/012345678912/a0bCDeFGH1IjK/2021/4/20/12/03/aBcdEFghIjkl/media/hls/master.m3u8
```

To play back from a web browser, make sure to configure CORS in both CloudFront and S3 bucket. For CloudFront configuration, follow the instructions in [Creating origin request policies](#) to attach a **CORS-S3 Origin** request policy and **SimpleCORS** response header policy to the CloudFront distribution. See the example configuration console page below:

## Cache key and origin requests

We recommend using a cache policy and origin request policy to control the cache key and origin requests.

Cache policy and origin request policy (recommended)

Legacy cache settings

### Cache policy

Choose an existing cache policy or create a new one.

CachingOptimized

Recommended for S3

Policy with caching enabled. Supports Gzip and Brotli compression.

[Create cache policy](#) [View policy](#)

### Origin request policy - optional

Choose an existing origin request policy or create a new one.

CORS-S3Origin

Policy for S3 origin with CORS

[Create origin request policy](#) [View policy](#)

### Response headers policy - optional

Choose an existing response headers policy or create a new one.

SimpleCORS

Allows all origins for simple CORS requests

[Create response headers policy](#) [View policy](#)

► **Additional settings**

For S3 CORS configuration, see [CORS configuration](#) to create appropriate rules for your S3 bucket.

Now you can play back your recorded video as if you were playing directly from a bucket.

For more information, see [Restricting access to an Amazon S3 origin](#).

# Using Amazon EventBridge with IVS Low-Latency Streaming

You can use Amazon EventBridge to monitor your Amazon Interactive Video Service (IVS) streams.

Amazon IVS sends change events about the status of your streams to Amazon EventBridge. All events that are delivered are valid. However, events are sent on a best-effort basis, which means there is no guarantee that:

- Events are delivered — A designated event can occur (e.g., a stream starts) but it is possible that Amazon IVS will not send a corresponding change event to EventBridge. Amazon IVS tries to deliver events for several hours before giving up.
- Events that are delivered will arrive in a specified timeframe — You may receive events up to a few hours old.
- Events are delivered in order — Events may be out of order, especially if they are sent within a short time of each other. For example, you could see Stream Down before Stream Up.

While it's rare for events to be missing, late, or out of sequence, you should handle these possibilities if you write business-critical programs that depend on the order or existence of notification events.

You can create EventBridge rules for any of the following events.

Event Type	Event	Sent When ...
IVS Stream State Change	Session Created	A channel stream key was used successfully and a stream session was created. This event fires when a stream is initiated, before video is processed or delivered to viewers. This event can help you determine if a stream was initiated but failed to go live; e.g., due to misconfiguration or limit breach.

Event Type	Event	Sent When ...
IVS Stream State Change	Session Ended	<p>The encoder disconnected and Amazon IVS is no longer receiving video. This event can help you determine when the encoder stopped sending media.</p> <p>Note: When the encoder disconnects, the Session Ended event may come before the Stream End event. This is because there may be a short period of time after the Session Ended event when Amazon IVS is still processing video.</p>
IVS Stream State Change	Stream Start	<p>A stream is being processed and segments are available for the viewer to watch. This event indicates that the video stream is being processed and can be watched by viewers. This event can help you determine if a stream went live successfully.</p>
IVS Stream State Change	Stream End	<p>A stream stops being processed and no longer produces video segments for the viewer. This event can help you determine when the stream ended and no new video segments can be consumed by the viewers. (Also see the note in Session Ended.)</p>
IVS Stream State Change	Stream Failure	<p>A stream is not being processed and is not available because processing capacity was exceeded.</p>

Event Type	Event	Sent When ...
IVS Stream Health Change	Starvation Start	A stream is not receiving data from the streamer; the stream is said to be in "starvation."
IVS Stream Health Change	Starvation End	A starving stream begins receiving data from the streamer and the stream is healthy again.
IVS Limit Breach	Ingest Bitrate	The incoming stream's bitrate exceeds the Amazon IVS limit.
IVS Limit Breach	Ingest Resolution	The incoming stream's resolution exceeds the Amazon IVS limit.
IVS Limit Breach	Concurrent Broadcasts	The total number of channels streaming at the same time exceeds the Amazon IVS limit.
IVS Limit Breach	Concurrent Viewers	The total number of viewers watching your channels at the same time exceeds the Amazon IVS limit.

Event Type	Event	Sent When ...
IVS Recording State Change	Recording Start	<p>A stream starts being processed, and the recording prefix is created and validated. Segments will be written to the storage location configured for the channel.</p> <p>Note that after a live stream starts and the Recording Start event is emitted, it takes a little time before the manifest files and video segments are written to the S3 bucket that is configured for the channel. We recommend that you play back or process recorded streams only after the Recording End event is sent.</p>
IVS Recording State Change	Recording End	A stream ends and recording stops for this channel.
IVS Recording State Change	Recording Start Failure	A stream starts but recording fails to start due to errors (for example, the S3 bucket does not exist or is not in the correct region). This live stream is not recorded.
IVS Recording State Change	Recording End Failure	Recording ends with failure, due to errors encountered during recording (e.g., if the attempt to write a master playlist fails). Some objects may still be written to the configured storage location.

**Note on stream IDs:** The `stream_id` field (in many events) is a unique stream identifier assigned each time a channel goes live. For a given channel, each live stream has a new `stream_id`. Hence,

each channel ARN can have many corresponding stream IDs. Stream IDs allow customers to distinguish different stream sessions on the same channel.

**Note on latency of some events:** Encoder-configuration settings, especially the IDR/keyframe interval, affect the timing of stream startup and the latency of related events (Stream Start and Recording Start). A shorter keyframe interval decreases this latency. See ["Reducing Latency"](#) in *Amazon IVS Streaming Configuration* for information on setting IDR/Keyframe.

## Creating Amazon EventBridge Rules for Amazon IVS

You can create a rule that triggers on an event emitted by Amazon IVS. Follow the steps in [Create a rule in Amazon EventBridge](#) in the *Amazon EventBridge User Guide*. When selecting a service, choose **Interactive Video Service (IVS)**.

### Examples: Stream State Change

**Stream Start:** This event is sent when a stream is being processed and segments are available for the viewer.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Stream State Change",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
  ],
  "detail": {
    "event_name": "Stream Start",
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
  }
}
```

**Stream End:** This event is sent when a stream stops being processed and no longer produces video segments for the viewer.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Stream State Change",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
  ],
  "detail": {
    "event_name": "Stream End",
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
  }
}
```

**Stream Failure:** This event is sent when a stream is not being processed and is not available because processing capacity was exceeded.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Stream State Change",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
  ],
  "detail": {
    "event_name": "Stream Failure",
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn",
    "reason": "Transcode capacity exceeded. Please try again."
  }
}
```



## Examples: Stream Health Change

**Starvation Start:** This event is sent when a stream is not receiving data from the streamer; the stream is said to be in “starvation.”

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Stream Health Change",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
  ],
  "detail": {
    "event_name": "Starvation Start",
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
  }
}
```

**Starvation End:** This event is sent when a starving stream begins receiving data from the streamer and the stream is healthy again.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Stream Health Change",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
  ],
  "detail": {
    "event_name": "Starvation End",
    "channel_name": "Your Channel",
  }
}
```

```

    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
  }
}

```

## Examples: Limit Breach

All limit-breach events include the name of the limit that is breached, the value of the limit, and the number by which the limit was exceeded (value at breach subtracted by the limit).

**Ingest Bitrate:** This event is sent when the incoming stream's bitrate exceeds the Amazon IVS limit.

```

{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Limit Breach",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
  ],
  "detail": {
    "limit_name": "Ingest Bitrate",
    "limit_value": 1234,
    "exceeded_by": 3,
    "limit_unit": "bits per second",
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
  }
}

```

**Ingest Resolution:** This event is sent when the incoming stream's resolution (total pixels or pixels per edge) exceeds the Amazon IVS limits.

Maximum total pixels exceeded:

```

{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",

```

```

"detail-type": "IVS Limit Breach",
"source": "aws.ivs",
"account": "aws_account_id",
"time": "2017-06-12T10:23:43Z",
"region": "us-east-1",
"resources": [
  "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"
],
"detail": {
  "limit_name": "Ingest Resolution",
  "limit_value": 495000,
  "exceeded_by": 426600,
  "limit_unit": "total pixels",
  "channel_name": "Your Channel",
  "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
}
}

```

### Maximum pixels per edge exceeded:

```

{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Limit Breach",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ivs:us-east-1:aws_account_id:channel/12345678-1a23-4567-
a1bc-1a2b34567890"TBID
  ],
  "detail": {
    "limit_name": "Ingest Resolution",
    "limit_value": 855,
    "exceeded_by": 45,
    "limit_unit": "pixels per edge",
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn"
  }
}

```

**Concurrent Broadcasts:** This event is sent when the total number of channels streaming at the same time exceeds the Amazon IVS limit.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Limit Breach",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [],
  "detail": {
    "limit_name": "Concurrent Broadcasts",
    "limit_value": 2,
    "exceeded_by": 3,
    "limit_unit": "active streams"
  }
}
```

**Concurrent Viewers:** This event is sent when the total number of viewers watching your channels at the same time exceeds the Amazon IVS limit.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "IVS Limit Breach",
  "source": "aws.ivs",
  "account": "aws_account_id",
  "time": "2017-06-12T10:23:43Z",
  "region": "us-east-1",
  "resources": [],
  "detail": {
    "limit_name": "Concurrent Viewers",
    "limit_value": 10,
    "exceeded_by": 11,
    "limit_unit": "viewers"
  }
}
```

## Examples: Recording State Change

For all recording state change events, the top-level path where all objects for this live stream are stored is `recording_s3_key_prefix`. In the case of failures, the reason for the failure is in `recording_status_reason`. The `recording_duration_ms` field is the number of milliseconds of recording duration.

**Recording Start:** This event is sent when a stream starts being processed and segments are being written to the storage location configured for the channel.

```
{
  "version": "0",
  "id": "12345678-1a23-4567-a1bc-1a2b34567890",
  "detail-type": "IVS Recording State Change",
  "source": "aws.ivs",
  "account": "123456789012",
  "time": "2020-06-23T20:12:36Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ivs:us-west-2:123456789012:channel/AbCdef1G2hij"
  ],
  "detail": {
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn",
    "recording_status": "Recording Start",
    "recording_status_reason": "",
    "recording_s3_bucket_name": "r2s3-dev-channel-1-recordings",
    "recording_s3_key_prefix": "ivs/v1/123456789012/AbCdef1G2hij/2020/6/23/20/12/j8Z9091ndcVs",
    "recording_duration_ms": 0,
    "recording_session_id": "a6RfV23ES97iyfoQ"
  }
}
```

**Recording End:** This event is sent when a stream ends and recording stops for this channel.

```
{
  "version": "0",
  "id": "12345678-1a23-4567-a1bc-1a2b34567890",
  "detail-type": "IVS Recording State Change",
  "source": "aws.ivs",
  "account": "123456789012",
```

```

"time": "2020-06-24T07:51:32Z",
"region": "us-west-2",
"resources": [
  "arn:aws:ivs:us-west-2:123456789012:channel/AbCdef1G2hij"
],
"detail": {
  "channel_name": "Your Channel",
  "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn",
  "recording_status": "Recording End",
  "recording_status_reason": "",
  "recording_s3_bucket_name": "r2s3-dev-channel-1-recordings",
  "recording_s3_key_prefix": "ivs/v1/123456789012/AbCdef1G2hij/2020/6/23/20/12/
j8Z9091ndcVs",
  "recording_duration_ms": 99370264,
  "recording_session_id": "a6RfV23ES97iyfoQ",
  "recording_session_stream_ids": ["st-254sopYUvi6F78ghp09vn0A",
"st-1A2b3c4D5e6F78ghij9Klmn"]
}
}

```

**Recording Start Failure:** This event is sent when a stream starts but recording fails to start due to errors (for example, the S3 bucket does not exist or is not in the correct region). This live stream is not recorded.

```

{
  "version": "0",
  "id": "12345678-1a23-4567-a1bc-1a2b34567890",
  "detail-type": "IVS Recording State Change",
  "source": "aws.ivs",
  "account": "123456789012",
  "time": "2020-06-23T20:12:36Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ivs:us-west-2:123456789012:channel/AbCdef1G2hij"
  ],
  "detail": {
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn",
    "recording_status": "Recording Start Failure",
    "recording_status_reason": "ValidationException",
    "recording_s3_bucket_name": "r2s3-dev-channel-1-recordings",
    "recording_s3_key_prefix": "",
    "recording_duration_ms": 0,
  }
}

```

```
    "recording_session_id": "a6RfV23ES97iyfoQ"
  }
}
```

**Recording End Failure:** This event is sent when recording ends with failure, due to errors encountered during recording. Some objects may still be written to the configured storage location.

```
{
  "version": "0",
  "id": "12345678-1a23-4567-a1bc-1a2b34567890",
  "detail-type": "IVS Recording State Change",
  "source": "aws.ivs",
  "account": "123456789012",
  "time": "2020-06-24T07:51:32Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ivs:us-west-2:123456a7-ab1c-2d34-e5f6-1a2b3c4d5678"
  ],
  "detail": {
    "channel_name": "Your Channel",
    "stream_id": "st-1A2b3c4D5e6F78ghij9Klmn",
    "recording_status": "Recording End Failure",
    "recording_status_reason": "InternalServerException",
    "recording_s3_bucket_name": "r2s3-dev-channel-1-recordings",
    "recording_s3_key_prefix": "ivs/v1/123456789012/AbCdef1G2hij/2020/6/23/20/12/j8Z9091ndcVs",
    "recording_duration_ms": 0,
    "recording_session_id": "a6RfV23ES97iyfoQ"
  }
}
```

# Logging Amazon IVS API Calls with AWS CloudTrail

Amazon Interactive Video Service (IVS) is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or AWS service in Amazon IVS. CloudTrail captures all API calls for Amazon IVS as events. The calls captured include API calls from the Amazon IVS console and from your applications.

If you create a *trail*, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including Amazon IVS events. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to Amazon IVS, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

## Amazon IVS Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon IVS, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see [Viewing Events with CloudTrail Event History](#).

For an ongoing record of events in your AWS account, including events for Amazon IVS, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the CloudTrail console, the trail applies to all AWS regions. The trail logs events from all Regions in the AWS partitions and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to analyze and act on the event data collected in CloudTrail logs. For more information, see these items in the *CloudTrail User Guide*:

- [Creating a Trail For Your AWS Account](#) (overview)
- [CloudTrail Supported Services and Integrations](#)
- [Configuring Amazon SNS Notifications for CloudTrail](#)
- [Receiving CloudTrail Log Files from Multiple Regions](#)
- [Receiving CloudTrail Log Files from Multiple Accounts](#)

All Amazon IVS actions are logged by CloudTrail and documented in the [IVS Low-Latency Streaming API Reference](#), [IVS Real-Time Streaming API Reference](#), and [IVS Chat API Reference](#). For



example, calls to the `CreateChannel`, `ListChannels`, and `DeleteChannel` endpoints generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine whether the request was made:

- With root or AWS Identity and Access Management (IAM) user credentials
- With temporary security credentials for a role or federated user.
- By another AWS service.

For more information, see the [CloudTrail userIdentity Element](#).

## Understanding Amazon IVS Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on.

CloudTrail log files contain one or more log entries. CloudTrail log files are not an ordered stack trace of the public API calls, so they do not appear in any specific order.

The following example shows a CloudTrail log entry for the `CreateChannel` endpoint.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "ABCDEFGHIJK1L2EXAMPLE:account_name",
    "arn": "arn:aws:sts::123456789012:assumed-role/
First_Streamer/1234567890123456789",
    "accountId": "123456789012",
    "accessKeyId": "ABCDEFGHIJKL1EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "ABCDEFGHIJK1L2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "First_Streamer"
      }
    }
  },
```

```
    "webIdFederationData": {},
    "attributes": {
      "mfaAuthenticated": "false",
      "creationDate": "2020-04-02T20:57:43Z"
    }
  },
  "eventTime": "2020-04-02T20:57:46Z",
  "eventSource": "ivs.amazonaws.com",
  "eventName": "CreateChannel",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "10.10.10.10",
  "userAgent": "console.amazonaws.com",
  "requestParameters": {
    "name": "default"
  },
  "responseElements": {
    "channel": {
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/1EXAMPLE",
      "authorized": false,
      "ingestEndpoint": "EXAMPLE.global-contribute.live-video.net",
      "latencyMode": "LOW",
      "name": "default",
      "playbackUrl": "https://EXAMPLE.m3u8",
      "tags": {}
    },
    "streamKey": {
      "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/2EXAMPLE",
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/1EXAMPLE",
      "tags": {}
    }
  },
  "requestID": "12a34bc5-EXAMPLE",
  "eventID": "a1b2c3de-EXAMPLE",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

# Amazon IVS Security

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The [shared responsibility model](#) describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** — AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the [AWS compliance programs](#).
- **Security in the cloud** — Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Amazon IVS. The following topics show you how to configure Amazon IVS to meet your security and compliance objectives.

## Topics

- [Data Protection](#)
- [Identity and Access Management](#)
- [Managed Policies for Amazon IVS](#)
- [Using Service-Linked Roles for Amazon IVS](#)
- [Logging and Monitoring](#)
- [Incident Response](#)
- [Resilience](#)
- [Infrastructure Security](#)

## Data Protection

For data sent to Amazon Interactive Video Service (IVS), the following data protections are in place:

- Amazon IVS encrypts data in transit via HTTPS API endpoints, RTMPS ingest, and HTTPS playback. No configuration is required for the API endpoints.
  - For ingest, streamers can secure their content by using RTMPS. This is available by default. See [Getting Started with IVS](#).
  - IVS channels can be configured to allow insecure RTMP ingest, though we recommend using RTMPS unless you have specific and verified use cases that require RTMP.
  - For transcoding/transmuxing, data may be transmitted unencrypted on internal Amazon networks.
  - For playback, data is served over HTTPS.
- Live-video content is not stored and is ephemeral. It simply travels through the system and is cached (on internal systems) while being viewed.
- For the auto-record-to-S3 feature, video content is written to Amazon S3. For more information, see [data protection in Amazon S3](#).
- All stored, customer-input metadata is in AWS-managed services using server-side encryption.
- To improve quality of service, Amazon IVS stores customer (end user) metadata (for example, buffer rates for a particular region). This metadata cannot be used to personally identify your end users.
- Public encryption keys (which you manage) can be used with the `ImportPlaybackKeyPair` API endpoint. See the [IVS Low-Latency Streaming API Reference](#). *Do not share these encryption keys.*

Amazon IVS does not require that you supply any customer (end user) data. There are no fields in channels, inputs, or input security groups where there is an expectation that you will provide customer (end user) data.

Do not put sensitive identifying information such as your customer (end user) account numbers into free-form fields such as a Name field. This includes when you work with the Amazon IVS console or API, AWS CLI, or AWS SDKs. Any piece of data that you enter into Amazon IVS might be included in diagnostic logs.

Streams are not end-to-end encrypted; a stream may be transmitted unencrypted internally in the IVS network, for processing.

# Identity and Access Management

AWS Identity and Access Management (IAM) is an AWS service that helps an account administrator securely control access to AWS resources. Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies. IAM account administrators control who can be authenticated (signed in) and authorized (have permissions) to use Amazon IVS resources. IAM is a feature of your AWS account offered at no additional charge.

**Important:** For comprehensive information, see the [AWS IAM product page](#), [IAM User Guide](#), and [Signing AWS API Requests](#). Throughout this section, we also provide links to specific sections of the *IAM User Guide*. You should be familiar with this material before proceeding.

## Audience

How you use IAM differs, depending on the work you do in Amazon IVS:

- **Service user** – If you use the Amazon IVS service to do your job, your administrator provides you with the credentials and permissions that you need. As you use more Amazon IVS features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Amazon IVS, see [Troubleshooting](#).
- **Service administrator** – If you're in charge of Amazon IVS resources at your company, you probably have full access to Amazon IVS. It's your job to determine which Amazon IVS features and resources your employees should access. You must then submit requests to your IAM administrator, to change the permissions of your service users. Review the information on this page to understand basic IAM concepts. To learn more about how your company can use IAM with Amazon IVS, see [How Amazon IVS Works with IAM](#).
- **IAM administrator** – If you're an IAM administrator, you can write policies to manage access to Amazon IVS. To view example Amazon IVS identity-based policies that you can use in IAM, see [Identity-Based Policy Examples](#).

## How Amazon IVS Works with IAM

Before you can make Amazon IVS API requests, you must create one or more IAM *identities* (users, groups, and roles) and IAM *policies*, then attach policies to identities. It takes up to a few minutes for the permissions to propagate; until then, API requests are rejected.

For a high-level view of how Amazon IVS works with IAM, see [AWS Services That Work with IAM](#) in the *IAM User Guide*.

## Identities

You can create IAM identities to provide authentication for people and processes in your AWS account. IAM groups are collections of IAM users that you can manage as a unit. See [Identities \(Users, Groups, and Roles\)](#) in the *IAM User Guide*.

## Policies

See these sections in the *IAM User Guide*:

- [Access Management](#) — All about policies.
- [Actions, Resources, and Condition Keys for Amazon IVS](#)
- [AWS Global Condition Context Keys](#)
- [IAM JSON Policy Elements Reference](#) — All the elements that you can use in a JSON policy.

By default, IAM users and roles don't have permission to create or modify Amazon IVS resources (even to change their own passwords). They also cannot perform tasks using the AWS console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources that they need.

IAM policies define permissions for an action regardless of the method that is used to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Policies are JSON permissions-policy documents made up of *elements*. Amazon IVS supports three elements:

- **Actions** — Policy actions for Amazon IVS use the `ivs` prefix before the action. For example, to grant someone permission to create an Amazon IVS channel with the Amazon IVS `CreateChannel` API method, you include the `ivs:CreateChannel` action in the policy for that person. Policy statements must include either an `Action` or `NotAction` element.
- **Resources** — The Amazon IVS channel resource has the following [ARN](#) format:

```
arn:aws:ivs:${Region}:${Account}:channel/${channelId}
```

For example, to specify the VgNkEJg0VX9N channel in your statement, use this ARN:

```
"Resource": "arn:aws:ivs:us-west-2:123456789012:channel/VgNkEJg0VX9N"
```

Some Amazon IVS actions, such as those for creating resources, cannot be performed on a specific resource. In those cases, you must use the wildcard (\*):

```
"Resource": "*" 
```

- **Conditions** — Amazon IVS supports some global condition keys: `aws:RequestTag`, `aws:TagKeys`, and `aws:ResourceTag`.

You can use variables as placeholders in a policy. For example, you can grant an IAM user permission to access a resource only if it is tagged with the user's IAM username. See [Variables and Tags](#) in the *IAM User Guide*.

Amazon IVS provides AWS managed policies that can be used to grant a preconfigured set of permissions to identities (read only or full access). You can choose to use managed policies instead of the identity-based policies shown below. For details, see [Managed Policies for Amazon IVS](#).

## Authorization Based on Amazon IVS Tags

You can attach tags to Amazon IVS resources or pass tags in a request to Amazon IVS. To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys. For more information about tagging Amazon IVS resources, see “Tagging” in the [IVS Low-Latency Streaming API Reference](#), [IVS Real-Time Streaming API Reference](#), and [IVS Chat API Reference](#).

For an example, see [View Amazon IVS Channels Based on Tags](#).

## Roles

See [IAM Roles](#) and [Temporary Security Credentials](#) in the *IAM User Guide*.

An IAM *role* is an entity within your AWS account that has specific permissions.

Amazon IVS supports using *temporary security credentials*. You can use temporary credentials to sign in with federation, assume an IAM role, or assume a cross-account role. You obtain temporary

security credentials by calling [AWS Security Token Service](#) API operations such as `AssumeRole` or `GetFederationToken`.

## Privileged and Unprivileged Access

API resources have privileged access. Unprivileged playback access can be set up through private channels; see [Setting Up Private Channels](#).

## Best Practices for Policies

See [IAM Best Practices](#) in the *IAM User Guide*.

Identity-based policies are very powerful. They determine whether someone can create, access, or delete Amazon IVS resources in your account. These actions can incur costs for your AWS account. Follow these recommendations:

- **Grant least privilege** — When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant more permissions as needed. Doing so is more secure than starting with permissions that are too lenient, then trying to tighten them later. Specifically, reserve `ivs:*` for admin access; do not use it in applications.
- **Enable multi-factor authentication (MFA) for sensitive operations** — For extra security, require IAM users to use MFA to access sensitive resources or API operations.
- **Use policy conditions for extra security** — To the extent practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses from which a request must come. You also can write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA.

## Identity-Based Policy Examples

### Use the Amazon IVS Console

To access the Amazon IVS console, you must have a minimum set of permissions which allow you to list and view details about the Amazon IVS resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console will not function as intended for identities with that policy. To ensure access to the Amazon IVS console, attach the following policy to the identities (see [Adding and Removing IAM Permissions](#) in the *IAM User Guide*).



The parts of the following policy provide access to:

- All Amazon IVS API endpoints
- Your Amazon IVS [service quotas](#)
- Amazon S3 endpoints needed for IVS auto-record-to-S3 functionality (low-latency-streaming) and IVS composite-recording functionality (real-time streaming).
- Auto-record-to-S3 service-linked-role creation
- Amazon Cloudwatch to get metrics for your live-stream session

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "ivs:*",
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": [
        "servicequotas:ListServiceQuotas"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": [
        "s3:CreateBucket",
        "s3:DeleteBucketPolicy",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicy",
        "s3:ListAllMyBuckets",
        "s3:PutBucketPolicy"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": [
        "iam:AttachRolePolicy",
        "iam:CreateServiceLinkedRole",
        "iam:PutRolePolicy"
      ]
    }
  ]
}
```

```

    ],
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/ivs.amazonaws.com/
AWSServiceRoleForIVSRecordToS3*"
  },
  {
    "Action": [
      "cloudwatch:GetMetricData"
    ],
    "Effect": "Allow",
    "Resource": "*"
  },
  {
    "Action": [
      "lambda:AddPermission",
      "lambda:ListFunctions"
    ],
    "Effect": "Allow",
    "Resource": "*"
  }
]
}

```

## Allow Users to View Their Own Permissions

This example shows a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the AWS console or programmatically using the AWS CLI or AWS API.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": [

```

```

        "arn:aws:iam:*:*:user/${aws:username}"
    ]
},
{
    "Sid": "NavigateInConsole",
    "Effect": "Allow",
    "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
    ],
    "Resource": "*"
}
]
}

```

## Access an Amazon IVS Channel

Here, you want to grant an IAM user in your AWS account access to one of your Amazon IVS channels, VgNkJgOVX9N. You also want to allow the user to stop the stream (`ivs:StopStream`), add metadata (`ivs:PutMetadata`), and update the channel (`ivs:UpdateChannel`). The policy also grants permissions required by the Amazon IVS console: `ivs:ListChannels`, `ivs:ListStreams`, `ivs:GetChannel`, and `ivs:GetStream`.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListChannelsInConsole",
            "Effect": "Allow",
            "Action": [
                "ivs:ListChannels",
                "ivs:ListStreams"
            ],
            "Resource": "arn:aws:ivs:*:*:channel/*"
        }
    ],
}

```

```

    {
      "Sid": "ViewSpecificChannelInfo",
      "Effect": "Allow",
      "Action": [
        "ivs:GetChannel",
        "ivs:GetStream"
      ],
      "Resource": "arn:aws:ivs:*:*:channel/VgNkJg0VX9N"
    },
    {
      "Sid": "ManageChannel",
      "Effect": "Allow",
      "Action": [
        "ivs:StopStream",
        "ivs:PutMetadata",
        "ivs:UpdateChannel"
      ],
      "Resource": "arn:aws:ivs:*:*:channel/VgNkJg0VX9N"
    }
  ]
}

```

## View Amazon IVS Channels Based on Tags

You can use conditions in your identity-based policy to control access to Amazon IVS resources based on tags. This example shows a policy that allows viewing a channel. This policy also grants the permissions necessary to complete this action on the Amazon IVS console.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListWidgetsInConsole",
      "Effect": "Allow",
      "Action": "ivs:ListChannels",
      "Resource": "arn:aws:ivs:*:*:channel/*"
    },
    {
      "Sid": "ViewChannelIfOwner",
      "Effect": "Allow",
      "Action": "ivs:GetChannel",
      "Resource": "arn:aws:ivs:*:*:channel/*",
      "Condition": {

```

```
        "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
    }
}
]
```

You can attach this policy to the IAM users in your account. However, permission is granted only if the channel is tagged with that user's username as an owner. If a user named richard-roe tries to view an Amazon IVS channel, the channel must be tagged `Owner=richard-roe` or `owner=richard-roe`; otherwise he is denied access. (The condition tag key `Owner` matches both `Owner` and `owner` because condition-key names are not case sensitive.)

## Troubleshooting

Use the following information to help diagnose and fix common issues that you might encounter when working with Amazon IVS and IAM.

- **I am not authorized to perform an action in Amazon IVS.**

The following example error occurs when the mateojackson IAM user tries to use the AWS console to view details about a channel but does not have `ivs:GetChannel` permission.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
ivs:GetChannel on resource: arn:aws:ivs:us-west-2:123456789012:channel/VgNkEJg0VX9N
```

In this case, Mateo asks his administrator to update his policies to allow him to access the `arn:aws:ivs:us-west-2:123456789012:channel/VgNkEJg0VX9N` resource using the `ivs:GetChannel` action.

- **I want to view my access keys.**

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair. Access keys have two parts:

- An access key ID (for example, `AKIAIOSFODNN7EXAMPLE`)
- A secret access key (for example, `wJa1rXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY`)

As with a username and password, you must use both the access key ID and the secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

**Important: Do not give your access keys to a third party, even to help [find your canonical user ID](#). Doing so might give someone permanent access to your account.**

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only when you create it. If you lose your secret access key, you must add new access keys to your IAM user.

You can have at most two access keys. If you already have two, you must delete one key pair before creating a new one. See [Managing Access Keys for IAM Users](#) in the *IAM User Guide*.

- **I'm an administrator and want to allow others to access Amazon IVS.**

To allow others to access Amazon IVS, you must create an IAM entity (user or role) for the person or application that needs access. The person or application will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants the correct permissions in Amazon IVS.

To get started, see [Creating Your First IAM Delegated User and Group](#) in the *IAM User Guide*.

- **I want to allow people outside my AWS account to access my Amazon IVS resources.**

You can create a role that users in other accounts or people outside your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources. For related information, see these sections of the *IAM User Guide*:

To learn ...	See ...
How to provide access to your resources across AWS accounts that you own	<a href="#">Providing Access to an IAM User in Another AWS Account That You Own</a>
How to provide access to your resources to third-party AWS accounts	<a href="#">Providing Access to AWS Accounts Owned by Third Parties</a>
How to provide access through <i>identity federation</i>	<a href="#">Providing Access to Externally Authenticated Users (Identity Federation)</a>

To learn ...	See ...
The difference between using roles and resource-based policies for cross-account access	<a href="#">How IAM Roles Differ from Resource-based Policies</a>

## Managed Policies for Amazon IVS

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining [customer managed policies](#) that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see [AWS managed policies](#) in the *IAM User Guide*.

### IVSReadOnlyAccess

Use the [IVSReadOnlyAccess](#) AWS managed policy to give your application developers access to all non-mutating IVS API endpoints (for both low-latency and real-time streaming).

## IVSFullAccess

Use the [IVSFullAccess](#) AWS managed policy to give your users access to all IVS and IVS Chat API endpoints (for both low-latency and real-time streaming). This policy includes additional permissions for dependent services, to allow full access to the IVS console.

## Policy Updates

View details about updates to AWS managed policies for Amazon IVS since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Amazon IVS Low-Latency Streaming [Document History](#) page.

Change	Description	Date
<a href="#">IVSReadOnlyAccess</a> – Change	<p>IVS added new actions to grant the following permissions in support of Server-Side Composition, Real-Time Composite Recording, and Tokenless Playback Restrictions:</p> <ul style="list-style-type: none"> <li>• GetComposition</li> <li>• ListCompositions</li> <li>• GetEncoderConfiguration</li> <li>• ListEncoderConfigurations</li> <li>• GetPlaybackRestrictionPolicy</li> <li>• ListPlaybackRestrictionPolicies</li> <li>• GetStorageConfiguration</li> <li>• ListStorageConfigurations</li> </ul>	February 16, 2024
<a href="#">IVSFullAccess</a> – New policy	IVS added a new policy to allow full access to IVS (both	December 5, 2023



Change	Description	Date
	low-latency and real-time streaming) and IVS Chat.	
<a href="#">IVSReadOnlyAccess</a> – New policy	IVS added a new policy to allow read-only access to IVS (both low-latency and real-time streaming).	December 5, 2023
Amazon IVS started tracking changes	Amazon IVS started tracking changes for its AWS managed policies.	December 5, 2023

## Using Service-Linked Roles for Amazon IVS

Amazon IVS uses IAM [service-linked roles](#). A service-linked role is a unique type of IAM role that is linked directly to an AWS service. Service-linked roles are predefined by Amazon IVS and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up Amazon IVS easier because you don't have to manually add the necessary permissions. Amazon IVS defines the permissions of its service-linked roles, and only Amazon IVS can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete an IVS service-linked role only after first deleting the related IVS resources. This prevents you from inadvertently removing permission for IVS to access the AWS resources associated with the service-linked role.

For information about other services that support service-linked roles, see [AWS Services That Work with IAM](#) and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

## Service-Linked Role Permissions for Amazon IVS

Amazon IVS uses the service-linked role named **AWSServiceRoleForIVSRecordToS3** to access Amazon S3 buckets on behalf of your Amazon IVS Channels.

The `AWSServiceRoleForIVSRecordToS3` service-linked role trusts the following services to assume the role:

- `ivs.amazonaws.com`

The role permissions policy allows Amazon IVS to complete the following actions on the specified resources:

- Action: `s3:PutObject` on your Amazon S3 buckets

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see [Service-Linked Role Permissions](#) in the *IAM User Guide*.

## Creating a Service-Linked Role for Amazon IVS

You don't need to manually create the service-linked role for IVS. Amazon IVS creates it for you, when you create a recording-configuration resource in the Amazon IVS Console, the AWS CLI, or the AWS API. The service-linked role is named `AWSServiceRoleForIVSRecordToS3`.

### Important

This service-linked role can appear in your account if you completed an action in another service that uses the features supported by this role. To learn more, see [A New Role Appeared in My IAM Account](#).

If you delete this service-linked role and then need to create it again, you can use the same process to recreate the role in your account. When you create a recording-configuration resource, Amazon IVS creates the service-linked role for you again.

## Editing a Service-Linked Role for Amazon IVS

Amazon IVS does not allow you to edit the `AWSServiceRoleForIVSRecordToS3` service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see [Editing a Service-Linked Role](#) in the *IAM User Guide*.

## Deleting a Service-Linked Role for Amazon IVS

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

### Note

If the Amazon IVS service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

**To delete Amazon IVS resources used by the `AWSServiceRoleForIVSRecordToS3` service-linked role:**

Use the Amazon IVS Console, the AWS CLI, or the AWS API to remove the recording-configuration association from all channels and delete all recording-configuration resources in the region.

**To manually delete the service-linked role using IAM:**

Use the IAM console, the AWS CLI, or the AWS API to delete the `AWSServiceRoleForIVSRecordToS3` service-linked role. For more information, see [Deleting a Service-Linked Role](#) in the *IAM User Guide*.

## Supported Regions for Amazon IVS Service-Linked Roles

Amazon IVS supports using service-linked roles in all of the regions where the service is available. For more information, see [Amazon IVS Service Endpoints](#).

## Logging and Monitoring

To log performance and/or operations, use Amazon CloudTrail. See [Logging Amazon IVS API Calls with AWS CloudTrail](#).

## Incident Response

To detect or alert for incidents, you can monitor your stream's health via Amazon EventBridge events. See [Using Amazon EventBridge with Amazon IVS: for Low-Latency Streaming](#) and for [Real-Time Streaming](#).

Use the [AWS Health Dashboard](#) for information on the overall health of Amazon IVS (by region).

## Resilience

IVS APIs use the AWS global infrastructure and is built around AWS Regions and Availability Zones. AWS Regions provide multiple Availability Zones, which are:

- Physically separated and isolated.
- Connected with low-latency, high-throughput, highly-redundant networking.
- More available, fault tolerant, and scalable than traditional single or multiple data-center infrastructures.

For more information on the APIs, see the [IVS Low-Latency Streaming API Reference](#), [IVS Real-Time Streaming API Reference](#), and [IVS Chat API Reference](#). For more information on AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

## Amazon IVS Video Data Plane

Video ingestion and distribution run over Amazon IVS's Content Delivery Network (CDN). The CDN is specialized and highly tuned for low-latency video. This enables Amazon IVS to provide customers with end-to-end, high quality video served to a global audience with minimal delay. The video CDN has global Points-of-Presence (PoPs), allowing broadcasters and viewers to be geographically dispersed.

Regardless of the AWS region where you chose to configure your Amazon IVS resources:

- Streamers automatically ingest video to a PoP geographically close to their location.
- Viewers stream video via the global video CDN.

Once ingested, video streams are processed and transcoded in one of several Amazon IVS datacenters. Amazon IVS does not provide automated failover for ingestion or transcoding failures. Instead, streamers should configure their encoders or broadcasting clients to automatically re-ingest on any broadcasting failures.

## Infrastructure Security

As a managed service, Amazon IVS is protected by the AWS global network security procedures. These are described in [Best Practices for Security, Identity, & Compliance](#).

### API Calls

You use AWS published API calls to access Amazon IVS through the network. Clients must support Transport Layer Security (TLS) 1.2 or later. We recommend TLS 1.3 or later (due to vulnerabilities in earlier versions). Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Also, API requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the [AWS Security Token Service](#) to generate temporary security credentials to sign requests.

You can call these API operations from any network location, but Amazon IVS does support resource-based access policies, which can include restrictions based on the source IP address. You can also use Amazon IVS policies to control access from specific Amazon Virtual Private Cloud (Amazon VPC) endpoints or specific VPCs. Effectively, this isolates network access to a given Amazon IVS resource from only the specific VPC within the AWS network.

Also, all API requests are signed sigv4.

For API details, see the [IVS Low-Latency Streaming API Reference](#), [IVS Real-Time Streaming API Reference](#), and [IVS Chat API Reference](#).

### Streaming and Playback

Playback happens over HTTPS from the edge to the viewer, and the “contribution edge” (ingest endpoint) supports RTMPS (RTMP over TLS) or RTMP if the channel is configured to allow insecure ingest. Amazon IVS streaming requires TLS version 1.2 or later. Streams are not end-to-end encrypted; a stream may be transmitted unencrypted internally in the IVS network, for processing.

## Service Quotas (Low-Latency Streaming)

The following are service quotas and limits for Amazon Interactive Video Service (IVS) endpoints, resources, and other operations. Service quotas (also known as limits) are the maximum number of service resources or operations for your AWS account. That is, these limits are per AWS account, unless noted otherwise in the table. Also see [AWS Service Quotas](#).

You use an endpoint to connect programmatically to an AWS service. Also see [AWS Service Endpoints](#).

All quotas are enforced per region.

**Important:** All accounts have limits on the number of concurrent views and concurrent streams. (A *view* is a unique viewing session which is actively downloading or playing video. For a more detailed definition, see the [Glossary](#).) *Ensure that your limits are adequate and request an increase if needed, especially if you are planning a large streaming event.*

### Service Quota Increases

For quotas that are adjustable, you can request a rate increase through the [AWS console](#). Use the console to view information about service quotas too.

API call rate quotas are not adjustable.

### API Call Rate Quotas

Endpoint Type	Endpoint	Default
Channel	BatchGetChannel	5 TPS
Channel	CreateChannel	5 TPS
Channel	DeleteChannel	5 TPS
Channel	GetChannel	5 TPS
Channel	ListChannels	5 TPS
Channel	UpdateChannel	5 TPS

Endpoint Type	Endpoint	Default
Playback restriction policy	CreatePlaybackRestrictionPolicy	5 TPS
Playback restriction policy	DeletePlaybackRestrictionPolicy	5 TPS
Playback restriction policy	GetPlaybackRestrictionPolicy	5 TPS
Playback restriction policy	ListPlaybackRestrictionPolicies	5 TPS
Playback restriction policy	UpdatePlaybackRestrictionPolicy	5 TPS
Private channel	DeletePlaybackKeyPair	3 TPS
Private channel	GetPlaybackKeyPair	3 TPS
Private channel	ImportPlaybackKeyPair	3 TPS
Private channel	ListPlaybackKeyPairs	3 TPS
Private channel	BatchStartViewerSessionRevocation	2 TPS
Private channel	StartViewerSessionRevocation	10 TPS
Recording configuration	CreateRecordingConfiguration	3 TPS
Recording configuration	DeleteRecordingConfiguration	3 TPS
Recording configuration	GetRecordingConfiguration	3 TPS
Recording configuration	ListRecordingConfigurations	3 TPS
Stream	GetStream	5 TPS
Stream	GetStreamSession	5 TPS
Stream	ListStreams	5 TPS
Stream	ListStreamSessions	5 TPS

Endpoint Type	Endpoint	Default
Stream	PutMetadata	5 TPS per channel 155 TPS per account
Stream	StopStream	5 TPS
Stream key	BatchGetStreamKey	5 TPS
Stream key	CreateStreamKey	5 TPS
Stream key	DeleteStreamKey	5 TPS
Stream key	GetStreamKey	5 TPS
Stream key	ListStreamKeys	5 TPS
Tags	ListTagsForResource	10 TPS
Tags	TagResource	10 TPS
Tags	UntagResource	10 TPS

## Other Quotas

Resource or Feature	Default	Adjustable	Description
Channels	5,000	Yes	Maximum number of channels, per AWS Region.
Concurrent streams	100	Yes	Maximum number of channels that can be streamed simultaneously, per AWS Region. If you exceed



Resource or Feature	Default	Adjustable	Description
			this threshold, the stream is rejected.
Concurrent views	15,000	Yes	Maximum number of views allowed to play back a live channel, across all channels in an AWS Region. (A <i>view</i> is a unique viewing session which is actively downloading or playing video. See the <b>Important</b> note at the beginning of this page.)
Ingest bitrate (if channel type is BASIC)	1.5 Mbps or 3.5 Mbps	No	<p>Maximum bits per second that can be streamed to a channel whose type is BASIC.</p> <ul style="list-style-type: none"> <li>• If input video quality is 480p or less, the default quota is 1.5 Mbps.</li> <li>• If input video quality is more than 480p but less than 1080p, the default quota is 3.5 Mbps.</li> </ul> <p>Warning: If you exceed this threshold, the stream probably will disconnect immediately. See the <a href="#">Amazon IVS Low-Latency Streaming API Reference</a> for details about channel type.</p>

Resource or Feature	Default	Adjustable	Description
Ingest bitrate (if channel type is STANDARD)	8.5 Mbps	No	Maximum bits per second that can be streamed to a channel whose type is STANDARD (the default). Warning: If you exceed this threshold, the stream probably will disconnect immediately. See the <a href="#">Amazon IVS Low-Latency Streaming API Reference</a> for details about channel type.
Ingest bitrate (if channel type is ADVANCED_HD )	8.5 Mbps	No	Maximum bits per second that can be streamed to a channel whose type is ADVANCED_HD . Warning: If you exceed this threshold , the stream probably will disconnect immediately. See the <a href="#">Amazon IVS Low-Latency Streaming API Reference</a> for details about channel type.
Ingest bitrate (if channel type is ADVANCED_SD )	8.5 Mbps	No	Maximum bits per second that can be streamed to a channel whose type is ADVANCED_SD . Warning: If you exceed this threshold , the stream probably will disconnect immediately. See the <a href="#">Amazon IVS Low-Latency Streaming API Reference</a> for details about channel type.

Resource or Feature	Default	Adjustable	Description
Ingest resolution	1080p (2.1M total pixels, 1920 pixels/edge)	No	Maximum resolution in pixels that can be streamed to a channel (regardless of its type). There are two relevant thresholds: total pixels and pixels per edge. Warning: If you exceed either of these thresholds, the stream probably will disconnect immediately. See the <a href="#">Amazon IVS Low-Latency Streaming API Reference</a> for details about channel type.
Metadata payload	1 KB	No	Maximum size of a <code>PutMetadata</code> request payload (Amazon IVS API).
Playback authorization key pairs	3	No	Maximum number of playback authorization key pairs, per AWS Region.
Playback restriction policies	3	No	Maximum number of playback restriction policies, per AWS Region.
Playback restriction policy countries	200	No	Maximum size of the <code>allowedCountries</code> list in a playback restriction policy; that is, the maximum number of countries per policy.

Resource or Feature	Default	Adjustable	Description
Playback restriction policy origins	5	No	Maximum size of the <code>allowedOrigins</code> list in a playback restriction policy; that is, the maximum number of origins per policy.
Playback restriction policy origin length	256	No	Maximum size (in characters) of an entry in the <code>allowedOrigins</code> list in a playback restriction policy.
Playback token size	2 KB	No	Maximum size of the entire JSON web token (JWT) used to initiate playback.
Recording configurations	20	Yes	Maximum number of recording configurations, per AWS Region.
Stream key	1	No	Maximum number of stream keys, per channel.

## Service Quotas Integration with CloudWatch Usage Metrics

You can use CloudWatch to proactively manage your service quotas, via CloudWatch *usage metrics*. You can use these metrics to visualize your current service usage on CloudWatch graphs and dashboards. Amazon IVS usage metrics correspond to Amazon IVS service quotas.

You can use a CloudWatch metric math function to display the service quotas for those resources on your graphs. You can also configure alarms that alert you when your usage approaches a service quota.

To access usage metrics:

1. Open the Service Quotas console at <https://console.aws.amazon.com/servicequotas/>
2. In the navigation pane, select **AWS services**.

3. From the AWS services list, search for and select **Amazon Interactive Video Service**.
4. In the **Service quotas** list, select the service quota of interest. A new page opens with information about the service quota/metric.

Alternately, you can get to these metrics through the CloudWatch console. Under **AWS Namespaces**, choose **Usage**. Then, from the **Service** list, choose **IVS**. (See [Monitoring Amazon IVS Low-Latency Streaming](#).)

In the **AWS/Usage** namespace, Amazon IVS provides the following metric:

Metric Name	Description
ResourceCount	<p>A count of the specified resources running in your account. The resources are defined by the dimensions associated with the metric.</p> <p>Valid statistic: Maximum (the maximum number of resources used during the 1-minute period).</p>

The following dimensions are used to refine the usage metric:

Dimension	Description
Service	The name of the AWS service containing the resource. Valid value: IVS.
Class	The class of resource being tracked. Valid value: None.
Type	The type of resource being tracked. Valid value: Resource.
Resource	<p>The name of the AWS resource. Valid values: ConcurrentStreams , ConcurrentViews .</p> <p>The ConcurrentStreams and ConcurrentViews usage metrics are copies of the ones in the <b>AWS/IVS</b> namespace (with the None dimension), as described in <a href="#">Monitoring Amazon IVS Low-Latency Streaming</a>.</p>

## Creating a CloudWatch Alarm for Usage Metrics

To create a CloudWatch alarm based on an Amazon IVS usage metric:

1. From the Service Quotas console, select the service quota of interest, as described above.  
Currently, alarms can be created only for ConcurrentStreams and ConcurrentViews.
2. In the **Amazon CloudWatch alarms** section, choose **Create**.
3. From the **Alarm threshold** dropdown list, choose the percentage of your applied quota value that you want to set as the alarm value.
4. For **Alarm name**, enter a name for the alarm.
5. Select **Create**.

# Amazon IVS Streaming Configuration

Amazon Interactive Video Service (IVS) allows developers to easily deliver low-latency video to viewers worldwide. With Amazon IVS, streamers need to handle only stream production, then send the stream to Amazon IVS. Amazon IVS handles video processing (ingesting and transcoding), delivery, and playback to viewers using the Amazon IVS player.

There is a wealth of solutions for live streaming. Whether you have a studio equipped with multiple cameras, visual switchers, graphics compositing, and a variety of audio mixing equipment, or you plan to start your first stream off a smartphone, you need to deal with some of the same concepts and encoding parameters.

This document describes how to configure video encoders to stream to Amazon IVS. The audience for this document is developers who want to build streaming functionality into their applications.

Note that audio-only input is not supported for IVS low-latency streaming.

## Prerequisites

Follow the steps in [Getting Started with IVS](#), to create a channel and set up streaming. In the process, a channel ARN (Amazon Resource Name) and stream key are assigned, along with URLs for ingesting and playing back a stream. You will need to point your streaming application to the ingest URL.

Before reading this document, you should be familiar with:

- Amazon IVS basics: Read [What is IVS Low-Latency Streaming](#) and [Getting Started with IVS](#)
- Amazon IVS API: Understand the [IVS Low-Latency Streaming API Reference](#).

## Reducing Latency

Amazon IVS low-latency streaming is compatible with most streaming applications and requires only minor changes to your streaming-application configuration. For the lowest possible latency, you must use the Amazon IVS player; third-party HLS video players are not supported. See the Amazon IVS Player SDK documentation.

To prepare your streaming application for low-latency streaming, do the following. (Note: not all these options are available on every streaming application.)

- On the video encoder, set `IDR/Keyframe` to a 2-second interval (or 1 second, for even lower end-to-end latency).

`IDR/Keyframe` directly affects the timing of stream startup and the latency of related `EventBridge` events (`Stream Start` and `Recording Start`). If `IDR/Keyframe` is 2 seconds, stream-start latency will be approximately 6-7 seconds. If `IDR/Keyframe` is 1 second, stream-start latency will be approximately 3-4 seconds. Your video will be available for viewers and auto-recording to Amazon S3 only after the initial stream-start latency period.

The shorter, 1-second keyframe interval has some QoS tradeoffs. It can cause the Amazon IVS Player's adaptive bitrate streaming (ABR) to switch resolution more often; the segment size is smaller, so the ABR check happens more often. Buffering may increase due to increased resolution-switching and/or if the viewer's network cannot download the segments fast enough. Evaluate these tradeoffs when deciding between a 1- or 2-second keyframe interval.

Avoid setting `IDR/Keyframe` to values higher than 5 seconds. Not only will the stream-start latency be higher than when using 1 or 2 seconds, but IVS will be unable to guarantee that every segment generated for playback will begin with an `IDR/keyframe`. Segments not beginning with an `IDR/keyframe` may result in decode errors or visual distortions when viewers start playback or change renditions.

- If available, set your encoder to zero-latency tuning within an x264 configuration.
- Ensure that buffer size (VBV) does not exceed the average bitrate (kilobits-per-second) of the stream.

## Avoid Third-Party Streaming/Forwarding Services

We strongly recommend you do not use third-party service to restream or forward content to Amazon IVS. *This will incur extra latency.* For low latency, stream directly to Amazon IVS.

## Encoder Settings

### Stream Ingest: Codecs and Ingest Protocols

**Codecs:** Amazon IVS supports H.264 for video and AAC (LC) for audio.

**Ingest protocols:** Amazon IVS supports the most common ingest protocols used in streaming software and hardware: RTMPS (Real-Time Messaging Protocol over a TLS/SSL connection), RTMP,



and SRT (Secure Reliable Transport). Amazon IVS streaming through RTMPS requires TLS version 1.2 or later.

## RTMPS/RTMP

Your video encoder must connect to Amazon IVS ingest over the RTMPS protocol associated with outbound port 443/TCP. To ensure this, specify an IVS ingest server, which includes the port in the path:

```
rtmps://<IVS-ingest-server>/<IVS-stream-key>
```

For example:

```
rtmps://a1b2c3d4e5f6.global-contribute.live-video.net:443/app/<IVS-stream-key>
```

IVS channels also can be configured to allow insecure RTMP ingest, though we recommend that you use RTMPS unless you have specific and verified use cases that require RTMP. When streaming RTMP, ensure that the protocol is set to `rtmp://` and remove the `:443` port. For example:

```
rtmp://a1b2c3d4e5f6.global-contribute.live-video.net/app/<IVS-stream-key>
```

## SRT

Your video encoder must connect to the ingest endpoint using the SRT protocol at port 9000. To ensure this, specify an ingest endpoint, which includes the port and passphrase in the path:

```
srt://<ingest-endpoint>:<port>?streamid=<stream-key>&passphrase=<passphrase>
```

Use a passphrase only if insecure ingest is not enabled for the channel.

For example:

```
srt://a1b2c3d4e5f6.srt.live-video.net:9000?streamid=sk_us-west-2_abcd1234efgh5678ijkl&passphrase=ZU5A3yrjGakghUNDr0c5NXBhsPrj1mtcKMNBluh7o
```

## Resolution/Bitrate/FPS

The stream's resolution largely determines its bitrate and frame rate (frames-per-second, or FPS). Use the following guidelines; these are our recommendations. Note the resolutions shown below are landscape orientation (horizontal x vertical), so reverse these for portrait orientation.

	<b>Acceptable Quality (SD) 480p (852x480)</b>	<b>Good Quality (HD) 720p (1280x720)</b>	<b>High Quality (Full HD) 1080p (1920x1080)</b>
<b>Bitrate</b>	Up to 1500 Kbps	Up to 4500 Kbps	Up to 8500 Kbps
<b>FPS</b>	30	30 or 60	30 or 60
<b>Keyframe interval</b>	2 seconds	2 seconds	2 seconds

Bitrate, FPS, and resolution are interrelated. The optimal values depend on circumstances and can be complicated to determine. Our best guidance is to start with the values above and experiment if desired. The goal is clear and smooth motion of video components during streaming and good resolution within the available bandwidth. Increasing frame rate and/or resolution increases overall video quality, but this is necessarily limited by bandwidth.

Amazon IVS supports framerate up to 60 FPS (including European PAL 25 and 50 standard frame rates). The higher the framerate, the better the quality -- as long as there is adequate bitrate bandwidth. Depending on the application, a low framerate can be fine; e.g., for a security camera.

## Channel Types

Channel type determines the allowable resolution and bitrate. *If you exceed the allowable input resolution or bitrate, the stream probably will disconnect immediately.*

There are four channel types: STANDARD, ADVANCED\_SD, ADVANCED\_HD, and BASIC. When you create a channel, the default type is STANDARD.

Depending on the channel type, video may be *transcoded* or *transmuxed*:

- Video on STANDARD and ADVANCED channels is transcoded: multiple qualities are generated from the original input, to automatically give viewers the best experience for their devices and network conditions. Transcoding allows higher playback quality across a range of download speeds.
- Video on BASIC channels is transmuxed: Amazon IVS delivers the original input to viewers.

All transcoded channels have *transcode* presets, which determine which renditions are produced. Think of these as ABR ladders. They allow you to trade off available download bandwidth and video quality, to optimize the viewing experience.

- STANDARD channels have one, default transcode preset.
- ADVANCED channels have two, selectable transcode presets:
  - *Constrained bandwidth delivery* uses a lower bitrate than STANDARD for each quality level. Use it if you have low download bandwidth and/or simple video content (e.g., talking heads).
  - *Higher bandwidth delivery* uses a higher bitrate for each quality level. Use it if you have high download bandwidth and/or complex video content (e.g., flashes and quick scene changes). This is the default.

## STANDARD Channels

STANDARD channels are transcoded. The highest video resolution produced is full HD, 1080p. This is the default channel type.

- **Transcode presets:** There is one, default transcode-preset ladder.
- **Audio:** For renditions 360p and below, audio is transcoded. For other renditions, original audio is passed through.

Input Resolution and Maximum Bitrate	Ladder Details
1080p60 at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. 1080p60 at source bitrate</li> <li>2. 720p60 at 3.4 Mbps</li> <li>3. 480p30 at 1.4 Mbps</li> <li>4. 360p30 at 0.63 Mbps</li> <li>5. 160p30 at 0.23 Mbps</li> </ol>
1080p30 at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. 1080p30 at source bitrate</li> <li>2. 720p30 at 2.4 Mbps</li> <li>3. 480p30 at 1.4 Mbps</li> <li>4. 360p30 at 0.63 Mbps</li> <li>5. 160p30 at 0.23 Mbps</li> </ol>

Input Resolution and Maximum Bitrate	Ladder Details
Less than 1080p60 and greater than 720p60, at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. Source passthrough</li> <li>2. 720p60 at 3.4 Mbps</li> <li>3. 480p30 at 1.4 Mbps</li> <li>4. 360p30 at 0.63 Mbps</li> <li>5. 160p30 at 0.23 Mbps</li> </ol>
Less than 1080p30 and greater than 720p30, at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. Source passthrough</li> <li>2. 720p30 at 2.4 Mbps</li> <li>3. 480p30 at 1.4 Mbps</li> <li>4. 360p30 at 0.63 Mbps</li> <li>5. 160p30 at 0.23 Mbps</li> </ol>
720p60 at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. 720p60 at 3.4 Mbps</li> <li>2. 480p30 at 1.4 Mbps</li> <li>3. 360p30 at 0.63 Mbps</li> <li>4. 160p30 at 0.23 Mbps</li> </ol>
720p30 at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. 720p30 at 2.4 Mbps</li> <li>2. 480p30 at 1.4 Mbps</li> <li>3. 360p30 at 0.63 Mbps</li> <li>4. 160p30 at 0.23 Mbps</li> </ol>
Less than 720p30/60 and greater than or equal to 480p30/60, at 8.5 Mbps	<ol style="list-style-type: none"> <li>1. 480p30 at 1.4 Mbps</li> <li>2. 360p30 at 0.63 Mbps</li> <li>3. 160p30 at 0.23 Mbps</li> </ol>

## ADVANCED-HD Channels

ADVANCED-HD channels are transcoded. The highest video resolution produced is HD, 720p.

- **Transcode presets:** There are two, selectable transcode-preset ladders.
- **Audio:** For renditions 360p and below, audio is transcoded. For other renditions, original audio is passed through.

Input Resolution and Maximum Bitrate	Ladder Details
720p60 up to 1080p60, at 8.5 Mbps	<p>Transcode preset: higher bandwidth delivery (default):</p> <ol style="list-style-type: none"> <li>1. 720p60 at 3 Mbps</li> <li>2. 480p30 at 1.3 Mbps</li> <li>3. 360p30 at 0.7 Mbps</li> <li>4. 160p30 at 0.27 Mbps</li> <li>5. Audio-only at 0.08 Mbps</li> </ol> <p>Transcode preset: constrained bandwidth delivery:</p> <ol style="list-style-type: none"> <li>1. 720p60 at 2.2 Mbps</li> <li>2. 480p30 at 0.8 Mbps</li> <li>3. 360p30 at 0.4 Mbps</li> <li>4. 160p30 at 0.22 Mbps</li> <li>5. Audio-only at 0.08 Mbps</li> </ol>
720p30 up to 1080p30, at 8.5 Mbps	<p>Transcode preset: higher bandwidth delivery (default):</p> <ol style="list-style-type: none"> <li>1. 720p30 at 2.3 Mbps</li> <li>2. 480p30 at 1.3 Mbps</li> <li>3. 360p30 at 0.7 Mbps</li> <li>4. 160p30 at 0.27 Mbps</li> <li>5. Audio-only at 0.08 Mbps</li> </ol> <p>Transcode preset: constrained bandwidth delivery:</p> <ol style="list-style-type: none"> <li>1. 720p30 at 1.9 Mbps</li> <li>2. 480p30 at 0.8 Mbps</li> <li>3. 360p30 at 0.4 Mbps</li> <li>4. 160p30 at 0.22 Mbps</li> <li>5. Audio-only at 0.08 Mbps</li> </ol>

Input Resolution and Maximum Bitrate	Ladder Details
<p>Less than 720p30/60 and greater than 480p30/60, at 8.5 Mbps</p>	<p>Transcode preset: higher bandwidth delivery (default):</p> <ol style="list-style-type: none"> <li>1. Source transcoded at 2.3 Mbps</li> <li>2. 480p30 at 1.3 Mbps</li> <li>3. 360p30 at 0.7 Mbps</li> <li>4. 160p30 at 0.27 Mbps</li> <li>5. Audio-only at 0.08 Mbps</li> </ol> <p>Transcode preset: constrained bandwidth delivery:</p> <ol style="list-style-type: none"> <li>1. Source transcoded at 1.9 Mbps</li> <li>2. 480p30 at 0.8 Mbps</li> <li>3. 360p30 at 0.4 Mbps</li> <li>4. 160p30 at 0.22 Mbps</li> <li>5. Audio-only at 0.08 Mbps</li> </ol>
<p>480p30/60 at 8.5 Mbps</p>	<p>Transcode preset: higher bandwidth delivery (default):</p> <ol style="list-style-type: none"> <li>1. 480p30 at 1.3 Mbps</li> <li>2. 360p30 at 0.7 Mbps</li> <li>3. 160p30 at 0.27 Mbps</li> <li>4. Audio-only at 0.08 Mbps</li> </ol> <p>Transcode preset: constrained bandwidth delivery:</p> <ol style="list-style-type: none"> <li>1. 480p30 at 0.8 Mbps</li> <li>2. 360p30 at 0.4 Mbps</li> <li>3. 160p30 at 0.22 Mbps</li> <li>4. Audio-only at 0.08 Mbps</li> </ol>

## ADVANCED-SD Channels

ADVANCED-SD channels are transcoded. Available renditions are capped at input quality, with no up-conversion.

- **Transcode presets:** There are two, selectable transcode-preset ladders.
- **Audio:** Audio is transcoded.

Input Resolution and Maximum Bitrate	Ladder Details
480p30/60 up to 1080p30/60, at 8.5 Mbps	<p>Transcode preset: higher bandwidth delivery (default):</p> <ol style="list-style-type: none"> <li>1. 480p30 at 1.3 Mbps</li> <li>2. 360p30 at 0.7 Mbps</li> <li>3. 160p30 at 0.27 Mbps</li> <li>4. Audio-only at 0.08 Mbps</li> </ol> <p>Transcode preset: constrained bandwidth delivery:</p> <ol style="list-style-type: none"> <li>1. 480p30 at 0.8 Mbps</li> <li>2. 360p30 at 0.4 Mbps</li> <li>3. 160p30 at 0.22 Mbps</li> <li>4. Audio-only at 0.08 Mbps</li> </ol>

## BASIC Channels

BASIC channels are transmuxed. A single rendition is produced.

- **Transcode presets:** NA
- **Audio:** Audio is transcoded.

Input Resolution and Maximum Bitrate	Ladder Details
Greater than 480p30/60 and less than or equal to 1080p30/60, at 3.5 Mbps	Source encoding parameters (no ladder)
480p30/60 at 1.5 Mbps	Source encoding parameters (no ladder)

## Video Settings

We recommend the following settings. They are available to most H.264 video-encoding software or hardware APIs.

- On the video encoder, set IDR/Keyframe to a 2-second interval (or 1 second, for even lower end-to-end latency).
- H.264 level: Main
- Scene change: Off (preferred)
- Chroma subsample: YUV420P
- CABAC: Preferred
- ColorSpace: BT.709 (recommended for maximum compatibility across HDTVs and computer displays). Amazon IVS video transcoding supports ColorSpace pass-through; advanced users can use other ColorSpace video and full-range video.

## Audio Settings

We support the following settings:

- Codec: AAC (LC)
- Bitrate: 96 Kbps to 320 Kbps
- Sample rate: 44.1 Khz or 48 Khz (it is best to match your production audio flow)
- Channels: Maximum 2 - Stereo (1: mono or 2: stereo audio channel support)

## Use CBR, Not VBR

Always use CBR (Constant BitRate), not VBR (Variable BitRate), as the rate-control method for encoders. CBR is better suited for the fixed-bandwidth nature of networks, and it produces more



predictable, stable video playback for client devices. With a consistent bitrate, it is easy for viewers to select a quality level that their connection can handle over time.

Depending on the complexity of the scene, VBR can result in spikes in bitrate, which can cause frame drops before the video reaches Amazon IVS and/or buffering in client players.

*We strongly recommend you only use CBR.* If you use VBR, your streams will be more subject to buffering and playback that is not smooth.

## Use Progressive Signals

**Use progressive signal flows; avoid any interlaced video in production flow and/or encoding.**

Progressive stream signals yield much better playback quality displaying a whole frame at a time, avoiding any motion artifacting that is produced when displaying an interlaced signal.

## Network Requirements

You must have a stable internet connection that can maintain an adequate, constant upload stream. An unstable internet connection could result in stream stuttering and lagging for your viewers.

**Use wired connections.** WiFi and LTE connections can be spotty or suffer from interference or latency due to bad QoS/packet-queue prioritization. Whenever possible, rely on a hardwired connection for streams.

Plan to allocate 50% more bandwidth than the minimum required. The overhead is added to compensate for the bitrate fluctuations in encoding of a video bitstream.

Use a dedicated Internet VLAN to encoding machines. Keeping the encoder on a separate network prevents potentially disruptive effects, including: pollution by traffic, bandwidth bottlenecks and adverse security factors.

## Closed Captioning

IVS supports closed captioning. As a streamer, if you want to offer captions to your audience, you must transmit caption data in an accepted format, either embedded in your stream or alongside your stream, through your video encoder.

Amazon IVS accepts captions in line 21 CEA-708/EIA-608 format (also referred to as 608 over 708). You can transmit captions using one of the following methods:

- CEA-708/EIA-608 embedded in the video elementary stream, as described in ATSC A/72 (SEI user\_data). This format is common among television broadcast encoders.
- CEA-708/EIA-608 transmitted via RTMPS onCaptionInfo script/AMF0 tag. This format is common among Internet broadcast encoders and media servers like Elemental Technologies and Wowza. The Amazon IVS Player SDKs support one language; they do not support multi-track captions playback.

Note: The Amazon IVS Player SDKs support caption data only in the CC1 NTSC field 1. They do not support multi-track captions playback.

When transmitting via RTMPS, the payload must contain an ECMA array with two element pairs:

- A string named type that contains the characters 708.
- A string named data that contains a base64-encoded CEA-708/EIA-608 payload.

For example:

```
00000000 12 00 00 69 00 00 00 00 00 00 00 02 00 0d 6f 6e |...i.....on|
00000010 43 61 70 74 69 6f 6e 49 6e 66 6f 08 00 00 00 02 |CaptionInfo....|
00000020 00 04 74 79 70 65 02 00 03 37 30 38 00 04 64 61 |..type...708..da|
00000030 74 61 02 00 3c 74 51 41 78 52 30 45 35 4e 41 4e |ta..<tQAxR0E5NAN|
00000040 4c 41 50 79 55 72 76 79 55 49 50 79 52 51 50 7a |LAPyUirvyUIPyRQPz|
00000050 49 35 66 7a 73 37 50 7a 76 4c 50 77 67 56 50 7a |I5fzs7PzvLPwgVPz|
00000060 33 36 66 7a 30 34 2f 78 6f 67 50 79 55 4c 2f 38 |36fz04/xogPyUL/8|
00000070 3d 00 00 09 00 00 00 74 |=......t|
```

If you use the Elemental video encoder, set it up as follows:

- Set caption embed to “capture 608 Field 1.”
- Embed captions with **onCaptionInfo** as the RTMPS tag in the output group.

For more information, see the blog post [Adding Closed Captions to an Amazon IVS Live Stream](#).

## Stream with FFmpeg

FFmpeg is a free, open-source project comprising a vast suite of software libraries for handling video, audio, and other multimedia files and streams. It can be used with many operating systems and devices.

See the [FFmpeg website](#) for installation and other information about FFmpeg. Use the latest static build (do not compile).

After installing, choose an audio/video input source for FFmpeg. You can look up what is available, as follows:

```
ffmpeg -list_devices true -f dshow -i dummy.
```

For more information, see [here](#). Depending on what is available and what capture method is targeted, you should be able to capture the video/audio (embedded) directly from your selected device and encode the signals with FFmpeg. For example:

- Webcam — To capture output from the Logitech C920 webcam:

```
ffmpeg -f dshow -video_size 1920x1080 -framerate 30 -i video="HD Pro Webcam C920":audio="Microphone (HD Pro Webcam C920)" -c:v libx264 -b:v 6000K -maxrate 6000K -pix_fmt yuv420p -r 30 -s 1920x1080 -profile:v main -preset veryfast -g 120 -x264opts "nal-hrd=cbr:no-scenecut" -acodec aac -ab 160k -ar 44100 -f flv rtmps://<IVS-ingest-server>/<IVS-stream-key>
```

- Video file — FFmpeg works with many video-file formats and capture cards. Here is an example of streaming based on a MP4 input:

```
ffmpeg -re -i input.mp4 -c:v libx264 -b:v 6000K -maxrate 6000K -pix_fmt yuv420p -s 1920x1080 -profile:v main -preset veryfast -force_key_frames expr:gte(t,n_forced*2) -x264opts "nal-hrd=cbr:no-scenecut" -acodec aac -ab 160k -ar 44100 -f flv rtmps://<IVS-ingest-server>/app/<IVS-stream-key>
```

For more information about what to enter for <IVS-ingest-server> and <IVS-stream-key>, see the information about setting up live-streaming software in [Getting Started with IVS](#). For example:

- Ingest server: `rtmps://jds34ksdg3las.global-contribute.live-video.net/app/`
- Stream key: `sk_us-west-2_abcd1234efgh5678ijkl`

# Stream with the Amazon IVS Broadcast SDK

The Amazon IVS broadcast SDK is for developers who are building Android, iOS, or Web applications with Amazon IVS. See the broadcast SDK documentation in the *Amazon IVS User Guide*, starting [here](#). There are subpages with guides for Android, iOS, and Web streaming. The broadcast SDKs enable you to customize bitrate, frame rate, and resolution.

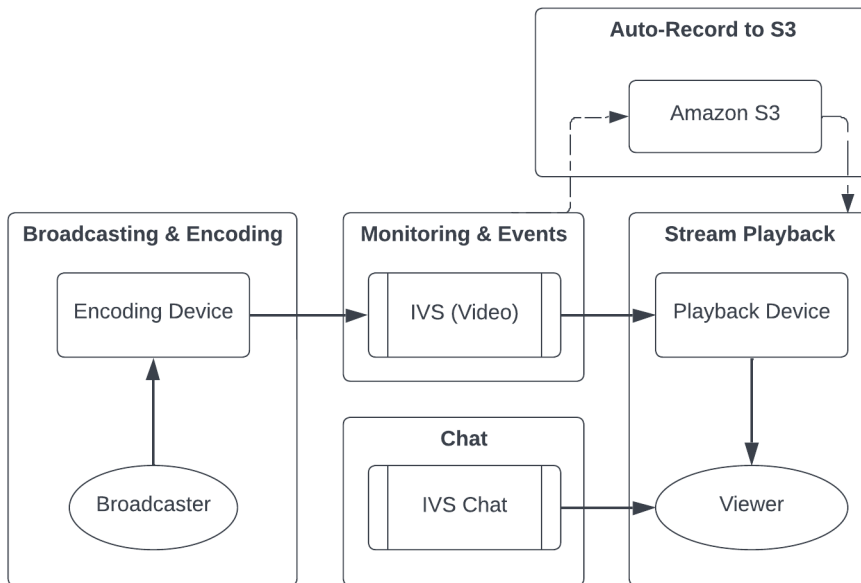
## Testing the Stream

**Always verify that your stream works.**

Navigate to the video stream in the [Amazon IVS console](#), to watch what is being streamed and manage the live stream.

# Troubleshooting FAQs

This document describes best practices and troubleshooting tips for Amazon Interactive Video Service (IVS). Unexpected or unintended behaviors may occur when using IVS. These behaviors can occur at various points in the streaming process, from broadcasting to playback of content:



For information on support and other Amazon IVS resources, see [Resources and Support](#).

## Broadcasting and Encoding

Questions in this section are about broadcasting, encoding, and first-mile conditions of streaming to IVS. These behaviors occur before the content reaches IVS servers.

Topics:

- [the section called “What is stream starvation?”](#)
- [the section called “Why did the stream suddenly stop?”](#)
- [the section called “What happens when I switch networks while streaming?”](#)
- [the section called “How can I have multi-region redundancy with IVS?”](#)
- [the section called “How do I troubleshoot an IVS Web Broadcast SDK session?”](#)
- [the section called “How do I use Google Chrome’s WebRTC-internals metrics to evaluate an IVS Web Broadcast SDK session?”](#)

## What is stream starvation?

"Stream starvation" is a delay or halt in content packet delivery when you are sending content to IVS; that is, when content is being ingested by IVS. If IVS does not get the expected amount of bits on ingest that the encoding device advertised it would send over a certain timeframe, this is considered a starvation event. Often, starvation events are caused by the broadcaster's encoder, local network conditions, and/or in transit over the public internet, between the encoding device and IVS.

From a viewer's perspective, starvation events may appear as video that lags, buffers, or freezes. Stream-starvation events can be brief (less than 5 seconds) or long (several minutes), depending on the nature of the starvation event.

To allow monitoring for starvation events, IVS sends starvation events as Amazon EventBridge events; see [Examples: Stream Health Change](#) in *Using Amazon EventBridge with Amazon IVS*. These are sent when a stream enters or exits a state of starvation. Depending on the use case, you can take an appropriate action, like notifying the broadcaster and viewers of intermittent stream conditions.

For additional starvation monitoring tools, see [Monitoring Amazon IVS Low-Latency Streaming](#), the IVS [ListStreams](#) API endpoint (filtering by health), and the IVS [GetStream](#) endpoint (to analyze an individual stream). Also see [the section called "How do I monitor stream-starvation events?"](#)

## Why did the stream suddenly stop?

The following are the most common reasons why a stream can abruptly stop (i.e., the stream session ends):

- **Missing ingest data** — When the ingest of a stream session completely stops (no data ingested into IVS) for 30 seconds, the IVS ingest server terminates the IVS stream session. The 30-second period allows the broadcaster to reconnect to the ingest server. However, in some cases (such as switching networks), reconnection to the existing stream session may not be possible, as the TLS handshake of RTMPS has been broken. Common root causes for this include network issues (like congestion between the broadcast device and IVS), complete loss of internet on the broadcast device, or the broadcast device not producing content segments (FLV tags).

Often, stream disconnection aligns with a stream-starvation event; the starvation event is triggered when there is a halt in incoming data. If a starvation-start event is sent and then a

stream-end event is sent (without a starvation-end event), this often indicates that the stream was ended due to no data being sent to IVS.

- **IVS StopStream endpoint** — During an IVS stream session, if the [StopStream](#) API call is made, the IVS stream session will end. The StopStream endpoint disconnects the incoming RTMPS stream from the IVS ingest server. Depending on the encoding software/hardware being used, a new stream session may be attempted.
- **Encoder error** — Some software/hardware encoders will disconnect the stream session when an error occurs during the encoding process. From the IVS perspective, these disconnections appear as intentional disconnects by the broadcaster. However, in the encoding logs, it may be determined that the stream was disconnected due to an unintentional error.

## What happens when I switch networks while streaming?

When a broadcaster switches networks (for example, from WiFi to cellular), an ongoing RTMPS connection is disconnected. While the broadcaster's internet connection probably is re-established after 3-4 seconds, the new connection has a new IP address due to the network switch, which generates a new RTMPS connection. During this switch, the previous RTMPS connection is not disconnected cleanly: the encoder does not send IVS a disconnect message. As a result, IVS waits 30 seconds for the previous RTMPS connection to reconnect, which blocks the new RTMPS stream on the new network from connecting to IVS.

To enable faster switching between networks, we recommend that you use the IVS [StopStream](#) endpoint to close the previous stream session when the device switches networks. In this scenario, when the broadcast device connects to the new network, the broadcast device could call the StopStream endpoint to end the now-dormant stream. Following a successful StopStream call, the broadcast device could begin a new stream session on the new network without waiting for 30 seconds.

## How can I have multi-region redundancy with IVS?

Redundancy within IVS can be achieved in several ways; see [Resilience](#) in IVS Security .

IVS is separated into different networking planes; Control and Data.

- The *control plane* is regional (based on AWS regions) and stores information about IVS resources (channels, stream keys, playback key pairs, and recording configurations).

- The *data plane* is not restricted to an AWS region and is the network that carries data from ingest to egress. Even if a channel is created in the us-west-2 region (for example), the video that is streamed to that channel may not go through us-west-2.

Also see [Global Solution, Regional Control](#). Consider these two scenarios:

- If only one control-plane region (e.g., us-east-1) is being used — If a particular AWS control region experiences a degradation or outage, the IVS control plane may experience latency or errors when creating, reading, updating, or deleting any of the following: channels, stream keys, playback key pairs, or recording configurations. Trying to start a new stream during an outage may result in more latency or errors when initiating a stream session. Depending on severity of the degradation, it may be possible to continue broadcasting to a channel with an already ongoing stream.

If [playback authorization](#) is enabled, current viewers probably can continue their playback of ongoing streams, but new viewers may not be able to start viewing if there are issues with playback key-pair authorization. If playback authorization is not enabled, both current and new viewers should be able to view the ongoing stream.

The IVS Auto-Record to S3 feature also may be interrupted in the event of an outage.

The IVS control plane does not automatically fail over to another AWS region in the event of a regional outage.

- If two control-plane regions (e.g., us-east-1 and us-west-2) are being used, and the second region is a failover if the primary region is unavailable — IVS does not natively support regional control-plane failover; thus, if a control-plane region experiences issues, new streams starting or calls to the control plane may experience issues. However, the data plane probably would not be impacted, so ongoing streams for the control plane region would continue without issue. Moving the control plane to a secondary (failover) region would need to be accomplished on the application side. You can write custom implementation logic to handle control-plane failover. We do not have official guidance on how to manage a regional channel failover.

By separating the video data plane and the regional control plane, the IVS architecture adds resilience: ongoing live streams should have little to no interruption in the event of a regional control-plane failure. IVS maintains an SLA of 99.9% uptime and is committed to ensuring the stability of its infrastructure for its customers (see our [SLA](#)).



## How do I troubleshoot an IVS Web Broadcast SDK session?

The [IVS Web Broadcast SDK](#) works slightly differently than a normal IVS RTMPS ingest session. The Web Broadcast SDK leverages the WebRTC protocol to stream to an IVS endpoint. Once the content enters the IVS endpoint, it is processed and remuxed/transcoded into the HLS output for viewing.

Due to the nature of the Web Broadcast SDK, note these tips for troubleshooting encoding behaviors:

- Close any tabs/programs on the broadcasting device that are not required to be open during the broadcasting session. Extraneous tabs/programs can use computing resources (such as CPU, RAM, and networking), which can cause poor performance for the broadcasting application. For tabs/programs that cannot be closed, ensure they are not using unnecessary amounts of computing resources.
- Ensure that the device's upload speed exceeds 200 Kbps. (This is noted in one of the [Known Issues](#) for the Web Broadcast SDK.) To evaluate the upload speed, open the Task Manager of the broadcasting device to analyze the network available when streaming. If the upload speed/bitrate is lower than expected or desired, evaluate other tabs/processes that may be consuming bandwidth. Also, look at other machines on the local network that may be consuming high amounts of bandwidth.
- If there are random spikes in CPU usage, look at the Task Manager of the machine to understand what processes may be consuming CPU. A common service that randomly causes CPU usage is anti-virus software which runs periodic scans on the machine.
- Try to stream via <https://stream.ivs.rocks/> to help isolate environments and ensure that the application logic is not causing the undesirable behavior. This site is operated by IVS and is a solid testing environment to evaluate if any part of the integration with the Web Broadcast SDK is the root cause of the undesirable behavior.
- Try using Google Chrome's WebRTC-internals (see below).

## How do I use Google Chrome's WebRTC-internals metrics to evaluate an IVS Web Broadcast SDK session?

When streaming via the IVS Web Broadcast SDK, various behaviors can occur during encoding and sending of the broadcast. Follow these steps to troubleshoot or gather information about the session on the broadcasting device:

1. In Google Chrome, open the broadcasting webpage.
2. Open a new Chrome tab and go to `chrome://webrtc-internals/` (copy this exactly).
3. In the original broadcasting-webpage tab, start the Web Broadcasting SDK session and let the session run until the behavior is observed.
4. Once the behavior is observed, switch to the `chrome://webrtc-internals/` tab (do not end the broadcast session), and ensure that the correct webpage is displayed:

► Create Dump

Read stats From:

**Note:** computed stats are in []. Experimental stats are marked with an \* at the end and do not show up in the getStats result.

<https://stream.ivs.rocks/> [ rid: 3067, lid: 1, pid: 32946 ]

GetUserMedia Requests

```
https://stream.ivs.rocks/, { iceServers: [], iceTransportPolicy: all, bundlePolicy: max-bundle, rtcMuxPolicy: require, iceCandidatePoolSize: 0 }
```

```
ICE connection state: new
Connection state: new
Signaling state: new
ICE Candidate pair: (not connected)
► ICE candidate grid
```

#### Stats Tables

Filter statistics by type including

- certificate (id=CF9C:62:D5:A8:03:45:55:A5:00:F7:0A:59:1D:AA:23:46:DE:31:45:AE:A2:48:6A:03:66:FC:2B:81:2F:2B:32:AD)
- data-channel (id=D1)
- track (id=DEPRECATED\_TO1)
- track (id=DEPRECATED\_TO2)
- local-candidate (candidateType=host, tcpType=active, id=l8m+mV7dh)
- local-candidate (candidateType=host, id=I9P+Kok6N)
- local-candidate (candidateType=host, id=IUUCiN2O)
- local-candidate (candidateType=host, tcpType=active, id=ISkSIIgsx)
- local-candidate (candidateType=host, id=IUUWMOuTJ)
- local-candidate (candidateType=host, tcpType=active, id=lwKxg6czL)
- outbound-rtp (kind=audio, mid=1, ssrc=1134012001, id=OT01A1134012001)
- outbound-rtp (kind=video, mid=0, ssrc=3966401599, id=OT01V3966401599)
- peer-connection (id=P)
- media-source (kind=audio, id=SA2)
- media-source (kind=video, id=SV1)
- transport (id=T01)
- Stats graphs for track (id=DEPRECATED\_TO1)
- Stats graphs for outbound-rtp (kind=audio, mid=1, ssrc=1134012001, id=OT01A1134012001)
- Stats graphs for outbound-rtp (kind=video, mid=0, ssrc=3966401599, id=OT01V3966401599)
- Stats graphs for peer-connection (id=P)
- Stats graphs for media-source (kind=audio, id=SA2)
- Stats graphs for media-source (kind=video, id=SV1)

5. Open the **Create Dump** expandable section at the very top of the screen.
6. Select **Download the PeerConnection updates and stats data** at the top of the screen (right below **Create Dump**), to download the .txt file from the relevant session.
7. Once downloaded, the file will show an historical view of the WebRTC connection. You can view this in various tools or send it to the AWS Support team for further analysis.

# Monitoring and Events

Questions in this section are about IVS monitoring, metrics, and events.

Topics:

- [the section called “How do I monitor stream-starvation events?”](#)
- [the section called “How do I use Amazon CloudWatch to monitor IVS service quotas?”](#)
- [the section called “How do I diagnose stream instability using IVS Stream Health?”](#)

## How do I monitor stream-starvation events?

We recommend the following methods of monitoring for stream-starvation events:

- [Amazon EventBridge with Amazon IVS](#) — When a stream-starvation event starts or ends, IVS produces an EventBridge stream health change event. Using Amazon EventBridge targets and rules, you can use these stream-starvation event to get alerts when stream starvation is occurring. For details on targets and rules, see the [Amazon EventBridge User Guide](#).
- [Monitoring Amazon IVS Low-Latency Streaming](#) — During a live-stream session, data is recorded and then available via IVS stream-health analytics. This includes information about encoder configuration, ingest metrics, and stream-session events. This is beneficial when monitoring an ongoing stream or retroactively evaluating a stream. You can use the IVS console or API to identify streams that have experienced starvation. Stream-session data is available for 60 days, even after a channel is deleted, so this can be useful for identifying past streams with starvation events.
- [Filtering Streams by Health](#) — With the IVS console or the IVS [ListStreams](#) API endpoint, you can use the health filter to find stream sessions that are in a STARVING state. Also, the IVS CloudWatch metric for ConcurrentStreams includes a Health dimension that you can use to gather a total count of streams that are in a stream-starvation state. See [Monitoring Amazon IVS Low-Latency Streaming](#).
- You can use the IVS [GetStream](#) endpoint to analyze an individual stream.

Also see [the section called “What is stream starvation?”](#)

## How do I use Amazon CloudWatch to monitor IVS service quotas?

You can use Amazon CloudWatch to proactively monitor/manage IVS service quotas. See [IVS Service Quotas](#). This documentation includes information on creating CloudWatch alarms for usage metrics.

We recommend that you set up a proper SNS topic to notify the correct individuals/groups when an alarm is triggered. If the alarm is triggered and the quota is adjustable, you should request a service-quota increase with a new value. See [IVS Service Quotas](#) for information on requesting an increase.

## How do I diagnose stream instability using IVS Stream Health?

We recommend that you evaluate stream instability using the IVS Stream Health dashboard. Instructions are in [Monitoring Amazon IVS Low-Latency Streaming](#).

The dashboard has time-series graphs for video bitrate, frame rate, and audio bitrate; examples are below. Also, you can click **View in CloudWatch** to view the data in Amazon CloudWatch.

Several scenarios are discussed below.

### Low Internet Bandwidth or Internet Congestion

In this case, the stream is relatively unstable, even when bitrates are lowered. Either there is not enough bandwidth between the broadcaster and the ISP or between the ISP and IVS, or something is wrong in the network path to IVS. To resolve this, check that no other network process is using bandwidth, or contact the ISP for network diagnostics.

**IVS Stream Health dashboard:**

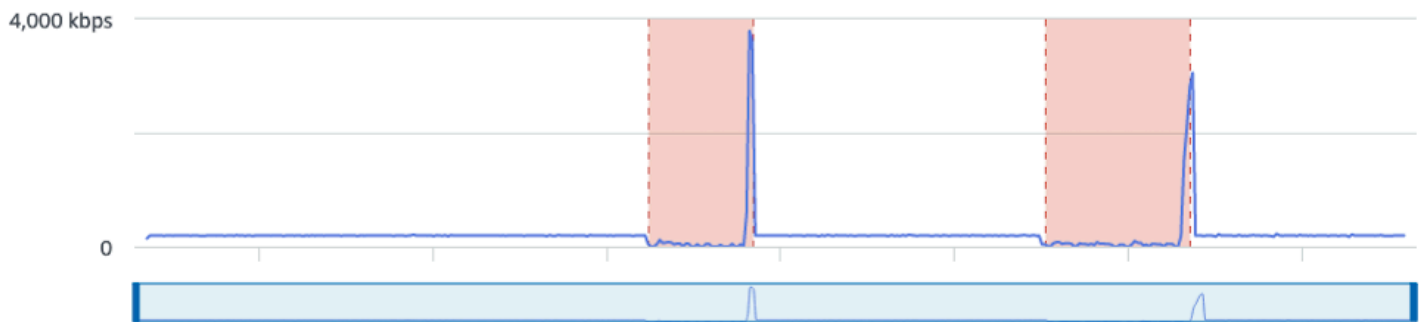
**Video bitrate**



**Frame rate**



**Audio bitrate**



**CloudWatch:**



## Excessive High Bitrate

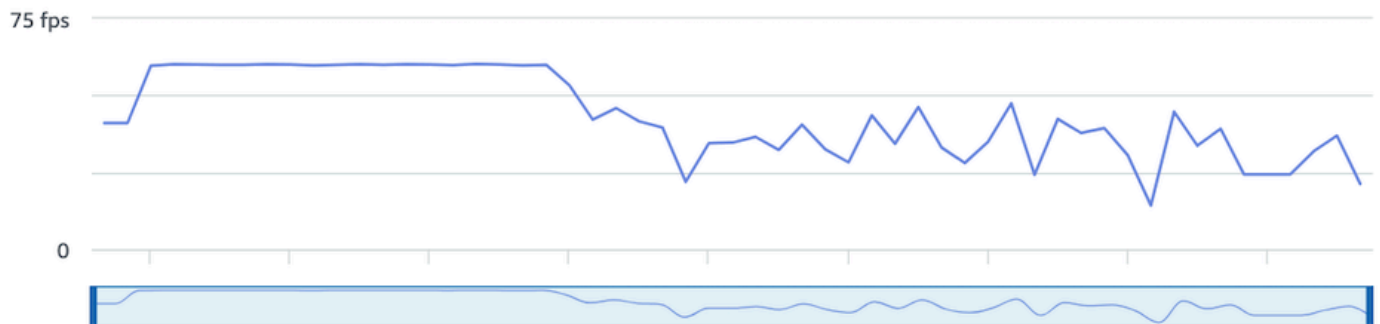
A higher bitrate does not necessarily mean better quality; here, high bitrate is causing instability. In many cases, due to network congestion, high bitrates causes stream instability throughout a broadcast. Adhere to the maximum bitrates listed in [the section called “Resolution/Bitrate/FPS”](#).

### IVS Stream Health dashboard:

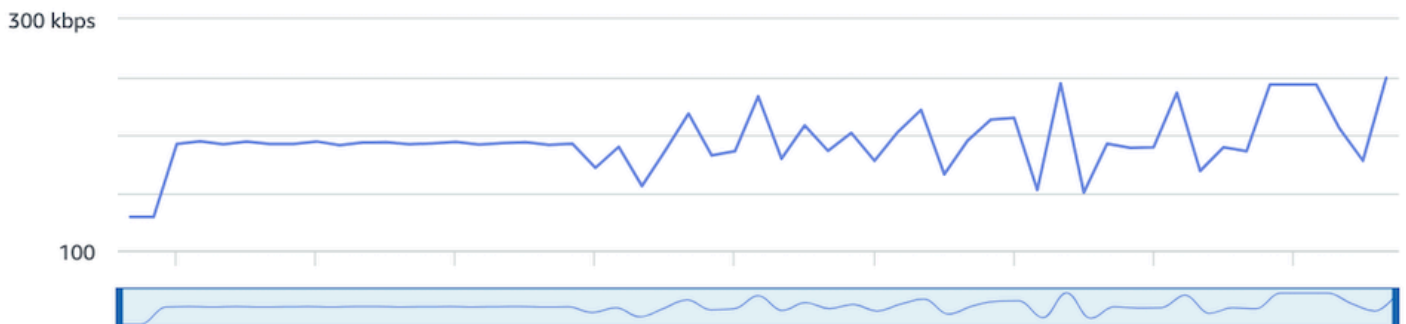
#### Video bitrate



#### Frame rate



#### Audio bitrate



### CloudWatch:

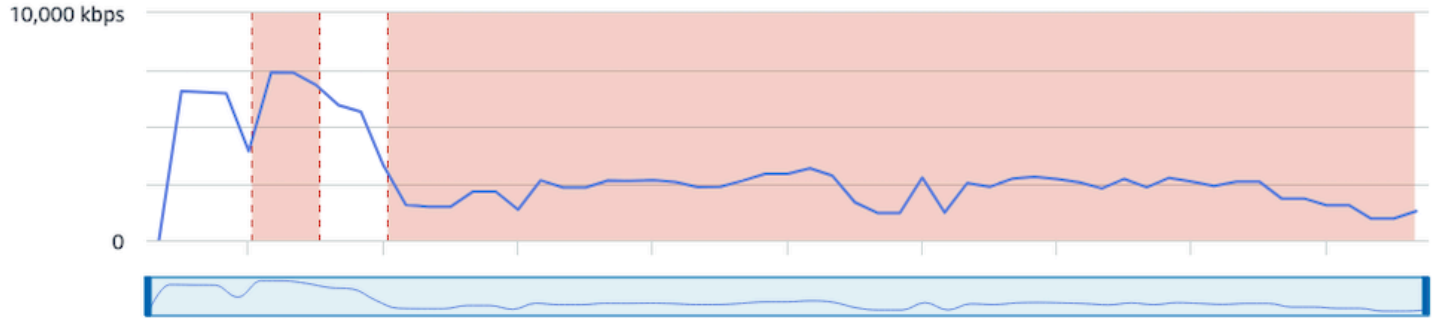


## Network or Hardware Problems

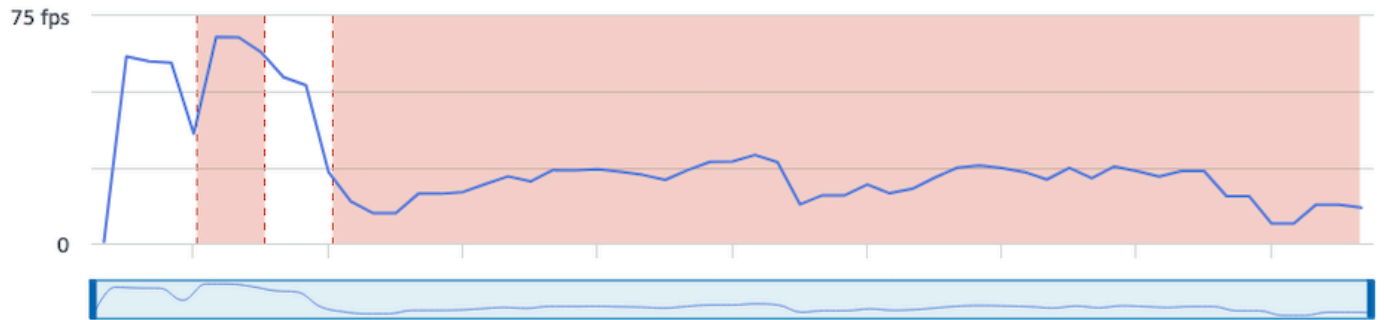
Video encoding takes a lot of computing resources, and sometimes the machine doing the video encoding cannot keep up with the load. In this case, verify that the machine is not overloaded (running too many things at a time) and that the encoder is up to date. Consider switching to an encoding preset that uses less CPU.

### IVS Stream Health dashboard:

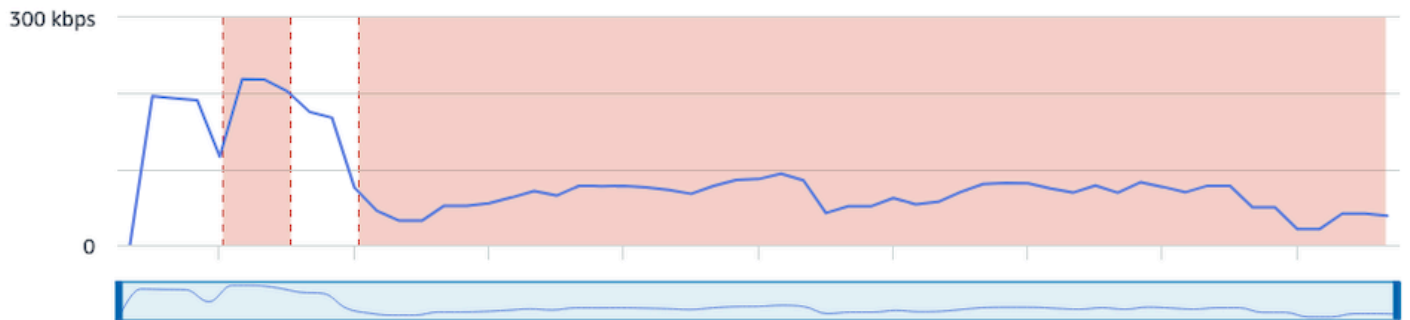
### Video bitrate



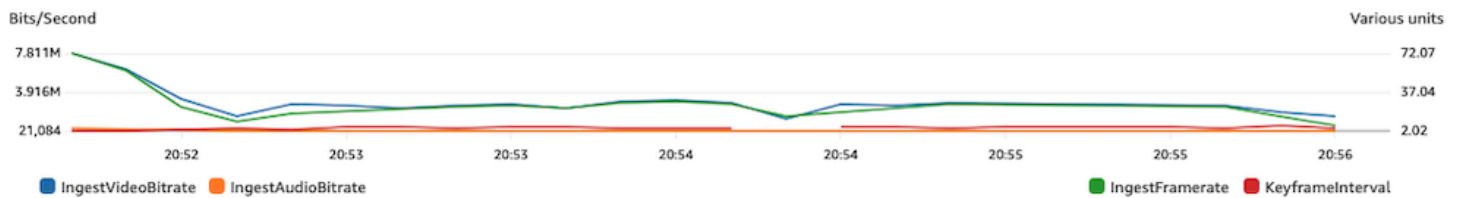
### Frame rate



### Audio bitrate



### CloudWatch:



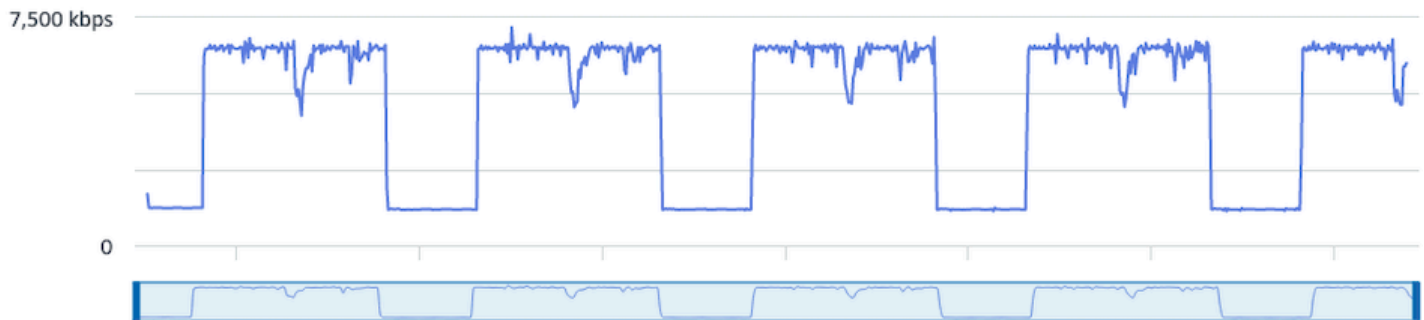


## Bitrate Spikes and Dips

Sometimes streaming encoders try to be too smart and optimize bitrate, often depending on the complexity of the frame being compressed. If the bitrate fluctuates rapidly, viewers may experience buffering from trying to load too much data. Ensure that Constant Bitrate (CBR) is enabled, as it maintains a consistent bitrate across the stream, regardless of frame complexity. Be aware that dips also can happen; that can be a sign that your machine does not have enough CPU power for the encoder to compress video.

### IVS Stream Health dashboard:

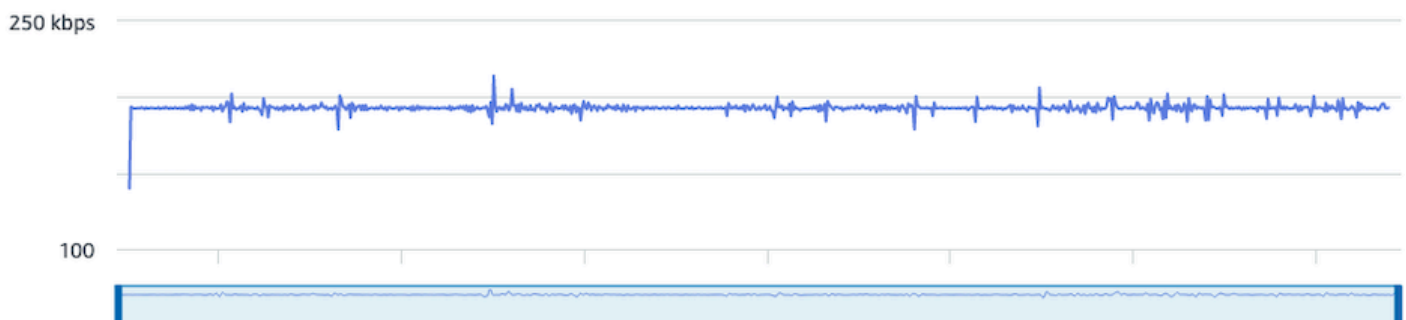
#### Video bitrate



#### Frame rate



#### Audio bitrate



## CloudWatch:

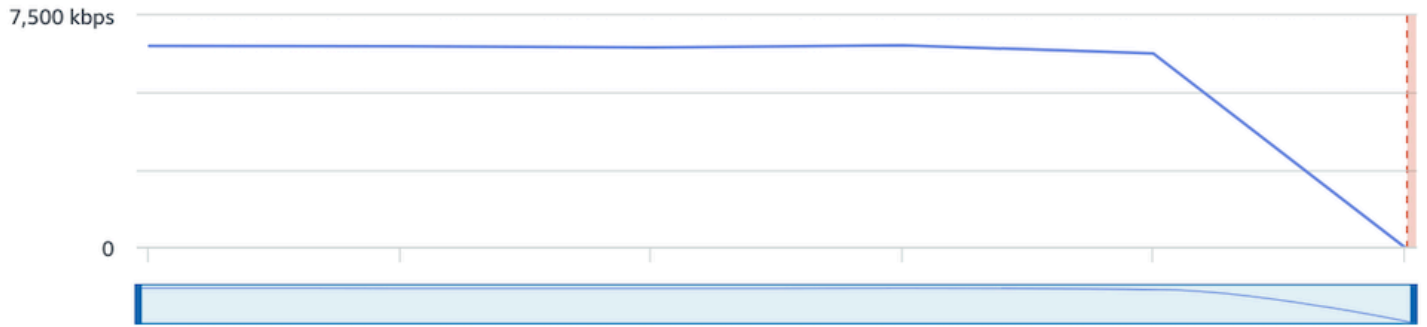


## Internet Disconnection

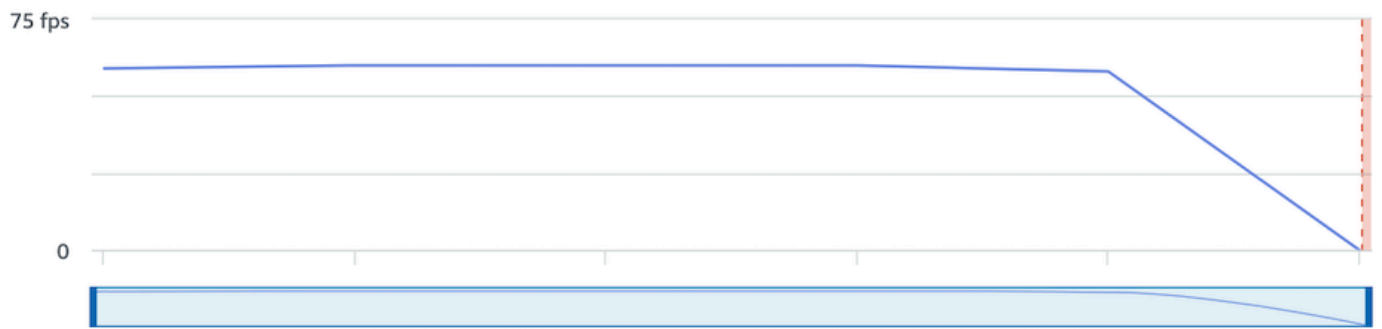
When a broadcast device experiences an internet issue, IVS servers enter a 30-second period in which they evaluate whether the same connection is re-established. If the same connection is not re-established, the IVS server ends the stream session. Some encoders will try to reconnect to the broadcast session if the internet connection is lost, in which case a new stream session may be started after the initial stream ends.

## IVS Stream Health dashboard:

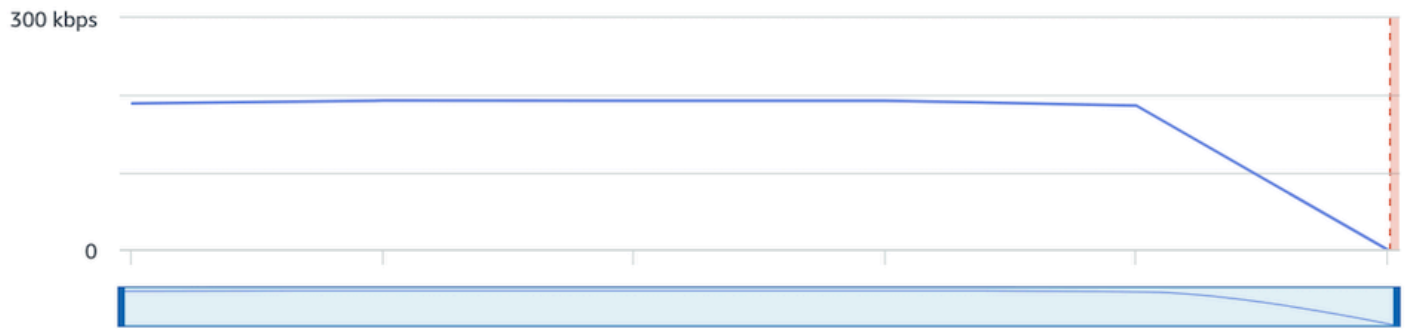
### Video bitrate



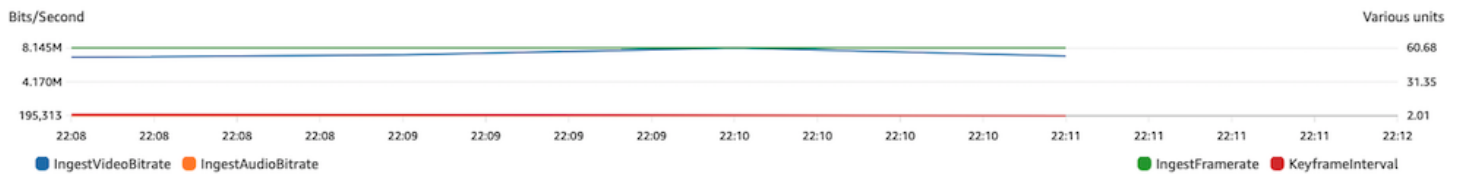
### Frame rate



### Audio bitrate



### CloudWatch:



# Stream Playback

Most of the information in this section is specific to the IVS Player SDK and may not apply to other players. For more information, see [Amazon IVS Player](#).

Topics:

- [the section called "How do I debug IVS player behaviors?"](#)
- [the section called "Why did playback freeze/stop for all viewers?"](#)
- [the section called "What is causing the IVS player to buffer?"](#)

## How do I debug IVS player behaviors?

To enable verbose logging to assist in debugging the IVS Player, use the `setLogLevel` player method. Alter the log level of the player to use the `DEBUG` argument; then the IVS Player will produce verbose logging around the state and logic occurring on the IVS Player.

To quickly test using the IVS Player, with or without `DEBUG` logs enabled, use the <https://debug.ivsdemos.com/> testing site. If `DEBUG` logs are enabled via the settings menu, you can view the logs in the browser console view.

## Why did playback freeze/stop for all viewers?

If playback for all viewers freezes/stops at the same time within the content, this probably is the result of an upstream behavior. Often the root cause is the broadcast encoder.

[Stream starvation](#) or adverse broadcast-encoder behaviors can have an impact on all viewers simultaneously. If the broadcasting encoding disconnects and a new stream session is started, all viewers stop receiving content concurrently. When you are evaluating this behavior, we recommend you evaluate the stream session using [Monitoring Amazon IVS Low-Latency Streaming](#).

## What is causing the IVS player to buffer?

In the context of playback of live-streaming video and audio, "buffering" means the playback device is unable to download the content before the content is supposed to be played. Buffering can manifest in several ways: content may randomly stop and start (also known as stuttering), content may stop for long periods of time (also known as freezing), or the player may enter a `BUFFERING` state.

There are many causes of buffering, which we can organize into three main categories:

- **Viewer-side buffering** often occurs when a single viewer or small group of viewers are impacted by a buffering event. The root cause of these buffering events often stems from a local network (LAN) or playback-device issue. In the case of a slow local network or device issue, the buffering may be resolved by ensuring that adaptive bitrate playback (ABR) is enabled, manually selecting a lower quality, or reducing the bandwidth being used by other programs and devices.
- **Network-level buffering** — Issues can occur between the local network and the IVS distribution server, otherwise known as the ISP level. Buffering behaviors that arise at the ISP level can be hard to troubleshoot, as full visibility into the ISP may be impossible. Behaviors like latency and network strain (e.g., the ISP cannot handle the overall incoming/outgoing traffic) can cause delays in providing content to the viewer.
- **Broadcast-side buffering** — Issues on the broadcast side of the live stream session can cause large-scale viewer-buffering problems. For example, if a broadcasting device stops sending data to IVS, IVS has no content to deliver to the player, and the IVS Player enters a buffering state when no content is being downloaded. In many cases, a broadcast-side buffering event results in most, if not all, viewers being impacted simultaneously.

## Auto-Record to Amazon S3

For more information, see [Auto-Record to Amazon S3](#).

Topics:

- [the section called “Why is some recording content missing?”](#)
- [the section called “Can KMS-S3 encryption be used with auto-record to S3?”](#)

### Why is some recording content missing?

There are various reasons why recorded content may be missing. We recommend the following steps to troubleshoot the missing content:

1. Ensure that Auto-Record to S3 is enabled for the desired IVS channel:
  - a. Console — On the details page for the relevant channel, in the **General configuration** section, ensure that Auto-record to S3 is Enabled. If it is enabled, check the **Recording configuration** to ensure that both **Storage** and **Recording prefix** are correct.

b. CLI — Run `get-channel` and pass in the desired IVS channel ARN:

```
aws ivs get-channel --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh"
```

See if a `recordingConfigurationArn` is returned.

2. Look in the designated S3 bucket for the Recording Contents for the specific stream session (see [S3 Prefix](#).) The S3 key prefix for a recorded session is in the Amazon EventBridge [Recording State Change event](#). Note: If the [merge fragmented streams](#) feature is enabled, some content may be another recorded session.
3. If the overall stream duration was less than 10 seconds or the content of the stream was missing (i.e., stream starvation occurred), recorded content may be missing as nothing was generated.

## Can KMS-S3 encryption be used with auto-record to S3?

The IVS auto-record to Amazon S3 feature does not support [KMS-S3 encryption](#). When attempting to use KMS-S3 encryption, the recording start will fail and produce a [Recording Start Failure EventBridge event](#). The recommended workaround is to use the supported [SSE-S3 encryption](#), which is enabled by default on all objects uploaded to Amazon S3.

## Miscellaneous Topics

Questions in this section are about topics that cannot be categorized elsewhere.

Topics:

- [the section called "What does the "pending verification" error mean?"](#)
- [the section called "Can I estimate the cost of IVS usage?"](#)

## What does the "pending verification" error mean?

When using IVS, an error may appear that states: "Your account is pending verification. Until the verification process is complete, you may not be able to carry out requests with this account. If you have questions, contact AWS Support."

This indicates that the AWS account you are using must be verified with AWS before you can use IVS. (While your account may work with other AWS services, IVS uses an enhanced verification method.)

To verify your AWS account, contact AWS Account Support — with the error message that you are receiving — from the AWS Support Center: <https://support.console.aws.amazon.com/support/home?#/>

## Can I estimate the cost of IVS usage?

While the exact cost of IVS usage cannot be determined before a stream session, a rough cost estimator is at: <https://ivs.rocks/calculator>. Additional pricing information is at: <https://aws.amazon.com/ivs/pricing/>.

# Undesired Content and Viewers

Malicious users may try to re-stream undesired content (e.g., professional sports) on your platform. This kind of streaming can dramatically increase the amount of live-streamed video that your application is serving as well as the costs associated with it, without adding value to your business. In addition to providing you with controls to stop active streams, Amazon IVS provides resources to help detect and prevent this kind of behavior in the first place.

## Detecting Undesired Content

### Anomaly Detection

You can detect and alert on the kind of anomalous spike in viewership that happens when certain undesired content is being streamed. (Once you detect that a spike has occurred, you can take the steps mentioned in [stop the stream and reset the stream key](#), as discussed below.)

Amazon CloudWatch allows you to create alarms which can send alerts under specific circumstances; for example, when your viewership spikes. Amazon IVS automatically reports concurrent views (CCV) metrics to Amazon CloudWatch for all your channels, so you only need to set up an alarm. To set up an anomaly-detection alarm based on CCV, follow these steps:

1. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. On the left navigation bar, select **Alarms**, then select **All alarms**.
3. On the top right of the page, select **Create alarm**.
4. Select **Select Metric**. Under *Metrics*, select **IVS**, then **All**, then select the checkbox next to **ConcurrentViews**.
5. On the lower right, select **Select metric**. A 4-step alarm-creation wizard opens.
6. Wizard: In **Step 1, Specify metric and conditions**, specify these settings:
  - a. **Statistic = Maximum**
  - b. **Period = 1 minute**
  - c. **Threshold type = Anomaly Detection**
  - d. **Whenever concurrent views is... = Greater than the band**
  - e. **Anomaly detection threshold = 3**



This threshold value is an initial suggestion. You may want to select a different value depending on your typical traffic patterns and needs. Use a lower value to watch your metrics more closely; a higher value, to get fewer alarms.

- f. Select **Next**.
7. Wizard: In **Step 2, Configuration actions**, choose an existing SNS topic or create a new one, to send email to an address you specify. To create a topic which sends an email, select **Create new topic**, provide a topic name, enter your email address, and select **Create topic**. Select **Next** to continue.
8. Wizard: In **Step 3, Add name and description**, add a name and optional description for the alarm, then select **Next**.
9. Wizard: In **Step 4, Preview and create**, verify that the information is correct, then select **Create alarm**.
10. Your alarm is created. If prompted, follow any instructions for confirming SNS subscriptions.

For more information, see:

1. [Monitoring Amazon IVS Low-Latency Streaming](#)
2. [Creating a CloudWatch alarm based on anomaly detection](#)

## Custom Content Moderation

You can explore custom content-moderation solutions to detect undesired content via image recognition. Amazon IVS provides the ability to [automatically record Amazon IVS live streams to Amazon S3](#), including the generation of thumbnail images for use in this kind of solution.

Consider these additional detection and prevention techniques:

- The [Amazon IVS moderation with Amazon Rekognition](#) demo showcases how to use IVS Auto-Record to S3 in conjunction with Amazon Rekognition to moderate live content.
- [Add Hive content moderation to your Amazon IVS video streams](#)
- [Creating Safer Online Communities with AI/ML Content Moderation](#) is a blog post about using Amazon Rekognition within an IVS application.

# Preventing Undesired Content and Viewers

## Stop the Stream and Reset the Stream Key

If you detect that a channel is being used to stream undesired content, you can use the Amazon IVS console to shut down the stream:

1. Open the [Amazon IVS console](#). (You can also access the Amazon IVS console through the [AWS Management Console](#).)
2. If needed, from the navigation bar, use the **Select a Region** drop-down to choose the region in which the channel is hosted.
3. Select the channel on which the stream that you want to stop is running.
4. On the channel page, navigate down to the **Live Stream** section and select **Stop stream**.

Even after you stop the stream, the broadcaster can restart the stream on that channel. To prevent this, reset the stream key; that prevents the broadcaster from restarting a stream without first acquiring a new stream key. To reset the stream key:

- While still on the channel page, navigate down to the **Stream configuration** section and select **Reset stream key**.

You also can stop a stream and reset (delete/create) the stream key programmatically. See the [Amazon IVS Low-Latency Streaming API Reference](#).

Depending on how your application issues stream keys, you may need to take further measures to prevent any new stream keys from being acquired.

## Use Private Channels

In many cases, undesired content is streamed to a large audience outside of your platform by simply embedding the playback URL in a third-party website. The best solution to prevent this kind of behavior is Amazon IVS private channels. By using private channels, you can restrict playback to viewers with valid playback tokens. Playback tokens are used to validate the viewer within the playback application, impeding viewership on unintended platforms. In addition, you can enable origin enforcement, which prevents viewers from watching streams on websites that aren't hosted on your domains. You can extend this protection to cover common streaming applications by also enabling strict origin enforcement.

Note that you can get the protection of private channels and authentication without forcing users to create and/or log in to formal accounts. Your playback application can simply acquire a token anonymously behind the scenes. You'll still be able to take advantage of origin enforcement.

To learn more about private channels, see:

- [Setting Up Private Channels](#) in the *IVS Low-Latency Streaming User Guide*. Within that document, to learn more about origin enforcement, see [Generate and Sign Playback Tokens](#).
- [Creating a Private Channel for Authorized Live Stream Playback with Amazon IVS](#) (blog post)

## Use Playback Restriction Policies

If you do not want to use [private channels](#), you can still benefit from some of the same protections by leveraging playback restriction policies. These policies allow you to enable features such as GeoBlocking and origin enforcement on public channels. You create a playback restriction policy using the IVS console or API, then attach the policy's ARN to your channels.

To learn more about playback restriction policies, see:

- [Getting Started with IVS Low-Latency Streaming](#) – See the information on preventing undesired content and Viewers.
- [IVS Low-Latency Streaming API Reference](#) – See playback restriction policy endpoints and the `PlaybackRestrictionPolicy` object.

# Costs (Low-Latency Streaming)

There are separate costs for Amazon IVS live video and Amazon S3 storage related to the auto-record-to-S3 feature.

## Live Video

The [Amazon IVS pricing](#) model incorporates separate fees for video input and output.

Video-input fees depend on your channel type. For details about channel types, see [Channel Types](#) in *IVS Streaming Configuration*.

For help selecting the right channel type for your use case, use the "Help me choose" tool in the console:

1. On the console's **Create channel** page, select **Custom configuration**.
2. Under **Channel type**, select **Help me choose**.
3. Follow the prompts until a recommendation is made, then choose **Select recommendation**.

For video output, you pay an hourly rate for video delivered to viewers. Rates vary by resolution and "billing region" (where the video is delivered from). There are several tiers of video-output costs based on usage, including a free tier.

A useful interactive tool is the [IVS Cost Estimator](#). You can plug in values for channel type, resolution, hours streamed, number of viewers, and billing region. When estimating costs, note the following rules of thumb:

- Viewers come and go, and on average, 50% of a stream is "delivered." The Cost Estimator includes a selector for "Average viewer watch duration." This defaults to 50%. Expect viewership for paid events to be higher; even in this case, though, it's likely that not all ticket-holders will view at the same time.
- Some viewers watch at a lower resolution than the source resolution of the broadcast. This is especially true for high-resolution streams: some viewers will watch at lower resolutions, which are less expensive. This is due to various viewer constraints, including bandwidth, network conditions, ISP, and hardware.
- Timing matters. For instance, if your stream competes with school, work, or vacation, this can affect your audience size.

- It is very hard to build a live audience from non-live users. Of course, there are exceptions; bringing in external talent (like influencers with their own following) can increase audience size.

## Auto-Record to Amazon S3

There are no Amazon IVS charges for using the auto-record to Amazon S3 feature or for writing to S3. There are charges for Amazon S3 storage, S3 API calls that Amazon IVS makes on behalf of the customer, and serving the stored video to viewers.

### Storing Recorded Video

Customers can generate estimates of S3 storage needs and costs by using the IVS console. When a customer uses the console to set up recording for a channel (either when the channel is created or later), a data-use estimator is offered. These data-use estimates can be plugged into the [AWS Pricing Calculator for S3](#) to estimate the monthly cost of S3 storage and data movement.

In the console, when creating a new channel or editing an existing channel, turn on **Enable automatic recording** in the **Record and store streams** area. This displays information about **Associated costs**.

## Record and store streams [Info](#)

### Auto-record to S3 [Info](#)

For improved redundancy, always record locally via your streaming tool.

Enable automatic recording

### Recording configuration

configuration-1 ▼



Create recording configuration

#### State

Active

#### Storage

s3-bucket-name [↗](#)

#### Recording prefix [Info](#)

s3://ivs-r2s3-ivsstoragebucket-1kem14abgbit8/ivs/v1/298083573632/<attached\_channel\_id>/

#### Recorded renditions

All renditions

#### Merge fragmented streams

Disabled

#### Thumbnail recording

At 60-second intervals

#### Thumbnail storage

Store thumbnails sequentially

#### Thumbnail resolution

Source (same resolution as input stream)



#### Associated costs

There are four cost components to consider when enabling record to S3: storage, request and data retrieval, data transfer, and data management. [Estimate data use.](#)

### ► Tags [Info](#)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Cancel

Create channel

Select **Estimate data use** to display the data-use calculator:

## Estimate data use



### Channel type

The channel type to use in estimations.

Standard

### Average input bitrate

8.5

Mbps

Only use numbers between 0 and 8.5

### Input resolution

1080p

### Input framerate

60fps

### Recording configuration

Choose an existing recording configuration



### Recorded renditions

All renditions

### Thumbnail recording

Record at an interval - 60s

### Target thumbnail resolution

Source (same resolution as input stream)

### Thumbnail storage

Store thumbnails sequentially

### Merge fragmented streams

Disabled

As noted on the screen, the estimates that are provided can be used with the [AWS Pricing Calculator](#) to compute estimates of the monthly cost incurred by S3 storage and data movement.

## Serving Recorded Video

The cost of serving recorded video to viewers depends on the CDN that is used. For example, see the Amazon CloudFront [pricing page](#).



# Resources and Support (Low-Latency Streaming)

## Resources

<https://ivs.rocks/> is a dedicated site to browse published content (demos, code samples, blog posts), estimate cost, and experience Amazon IVS through live demos.

[Getting Started with Amazon Interactive Video Service Series' Articles](#) is a series of articles about using Amazon IVS, for beginners. The articles give step-by-step walkthroughs of IVS APIs with interactive demos embedded in the posts. All the demos can be run directly in the posts themselves via an embedded CodePen. Over time this will cover various topics.

There are many Amazon IVS blog postings on a variety of topics:

- On the [AWS Blog](#) site, filter for Amazon IVS by selecting **Product or solution > Media Services > Amazon Interactive Video Service** on the right side of the page.
- See [this part](#) of the DEV Community site.

## Demos

For demos, code samples, and blog posts, see <https://ivs.rocks/examples>.

## Partner Solutions

Amazon IVS partners with third-party providers in the [Amazon Partner Network \(APN\)](#) to provide technology solutions to augment live-streaming applications. There are several types of partner solution areas:

This type of partner:	Offers solutions that do this ...
Analytics	Provide operational and business insights into your live-streaming video application. These insights, in turn, can drive increased engagement from viewers and identify opportunities to improve return on investment.

This type of partner:	Offers solutions that do this ...
Interactivity	Help drive engagement with audiences of your live-streaming video application.
Face and background filters	Enable broadcasters to change their facial or background appearance to audiences of their live streams.

## Analytics

[Bitmovin's](#) Analytics is a fully managed service with [analytics collectors](#) built for the Amazon IVS Player. Analytics enables you to track and monitor playback health across devices, understand viewer demographics, monitor quality of playback experience, and quickly identify any issues affecting viewers.

With actionable data collected across all your channels, Bitmovin's Analytics aids in increasing viewer engagement and retention through metric dashboards for Audience, Quality of Experience (QoE), and Top Errors.

This gives you access to about 40 metrics with 30 filters and breakdowns. Also, 200 dimensions and filters are available through Bitmovin's API and data exports.

To integrate Bitmovin's Analytics with the Amazon IVS Player SDK, see the following Getting Started guides: [Android](#) and [iOS](#).

## Interactivity

[LiveLike](#) offers a ready-to-use engagement platform that can enhance your online user experience in just a few weeks. Boost your average revenue per user through increased registration, interactions, impressions, and sponsorships. See results such as a 70% increase in year-over-year registrations (2022 vs 2021) with our NASCAR case study. Reduce churn and increase retention by creating interactive and engaging experiences on your platform with our solution. To integrate LiveLike with Amazon IVS, see the following blog post: [A Quick Guide to LiveLike: How to Enhance Live Stream Interactivity](#).

## Face and Background Filters

DeepAR is a technology company that builds AR infrastructure for digital product teams. Businesses of every size - from startups to public companies - use our software to provide world

class AR experiences to billions of users around the world. To integrate DeepAR with Amazon IVS, see the DeepAR page on [Amazon IVS Integration](#).

[BytePlus Effects](#) combines a huge library of AR effects, stickers, and filters, giving app developers all the tools they need to drive deeper engagement with their audience. To integrate BytePlus with Amazon IVS, see the following blog post: [How to improve user engagement with real-time AR effects using BytePlus Effects and Amazon IVS](#).

Camera Kit is Snap AR's SDK that allows partners to leverage Snap AR technology in their applications and websites. Using Camera Kit, businesses can bring a new dimension to their customer experiences and unleash new applications for Snap's underlying AR technology. To integrate Snap AR Lenses using Snap's Camera Kit SDK with Amazon IVS, see the following blog post: [Unlocking creator expressions to enhance live streaming experiences with Amazon IVS and Snap's Camera Kit AR SDK](#).

## Support

The [AWS Support Center](#) offers a range of plans that provide access to tools and expertise to support your AWS solutions. All support plans provide 24/7 access to customer service. For technical support and more resources to plan, deploy, and improve your AWS environment, choose a support plan that best aligns with your AWS use case.

[AWS Premium Support](#) is a one-on-one, fast-response support channel to help you build and run applications on AWS.

[AWS re:Post](#) is a community-based Q&A site for developers to discuss technical questions related to Amazon IVS.

[Contact Us](#) has links for nontechnical inquiries about your billing or account. For technical questions, use the discussion forums or support links above.

# Glossary

Also see the [AWS glossary](#). In the table below, LL stands for IVS low-latency streaming; RT, IVS real-time streaming.

Term	Description	LL	RT	Chat
AAC	Advanced Audio Coding. AAC is an audio coding standard for lossy digital audio <a href="#">compression</a> . Designed to be the successor of the MP3 format, AAC generally achieves higher sound quality than MP3 at the same bitrate. AAC has been standardized by ISO and IEC as part of the MPEG-2 and MPEG-4 specifications.	✓	✓	
Adaptive bitrate streaming	Adaptive Bitrate (ABR) streaming allows the IVS player to switch to a lower <a href="#">bitrate</a> when connection quality suffers, and to switch back to a higher bitrate when connection quality improves.	✓		
Adaptive streaming	See <a href="#">Layered encoding with simulcast</a> .		✓	
Administrative user	An AWS user with administrative access to resources and services available in an AWS account. See <a href="#">Terminology</a> in <i>AWS Setup User Guide</i> .	✓	✓	✓
ARN	<a href="#">Amazon Resource Name</a> , a unique identifier for an AWS resource. Specific ARN formats depend on the resource type. For ARN formats used by IVS resources, see in <i>Service Authorization Reference</i> .	✓	✓	✓
Aspect ratio	Describes the ratio of frame width to frame height. For example, 16:9 is the aspect ratio that corresponds to the Full HD or 1080p <a href="#">resolution</a> .	✓	✓	
Audio mode	A preset or custom audio configuration optimized for different types of mobile device users and the		✓	

Term	Description	LL	RT	Chat
	equipment that they use. See <a href="#">IVS Broadcast SDK: Mobile Audio Modes (Real-Time Streaming)</a> .			
AVC, H.264, MPEG-4 Part 10	Advanced Video Coding, also referred to as H.264 or MPEG-4 Part 10, a video compression standard for lossy digital video <a href="#">compression</a> .	✓	✓	
Background replacement	A type of <a href="#">camera filter</a> that enables live-stream creators to change their backgrounds. See <a href="#">Background Replacement</a> in <i>IVS Broadcast SDK: Third-Party Camera Filters (Real-Time Streaming)</i> .		✓	
Bitrate	A streaming metric for the number of bits transmitted or received per second.	✓	✓	
Broadcast, broadcaster	Other terms for <a href="#">stream</a> , <a href="#">streamer</a> .	✓		
Buffering	A condition that occurs when the playback device is unable to download the content before the content is supposed to be played. Buffering can manifest in several ways: content may randomly stop and start (also known as stuttering), content may stop for long periods of time (also known as freezing), or the IVS player may pause playback.	✓	✓	
Byte-range playlist	A more granular playlist than the standard <a href="#">HLS playlist</a> . The standard HLS playlist is made up of 10-second media files. With a byte-range playlist, the segment duration is the same as the <a href="#">keyframe interval</a> configured for the <a href="#">stream</a> .  Byte-range playlist is available only for the broadcasts that were auto-recorded to an <a href="#">S3 bucket</a> . It is created in addition to the <a href="#">HLS playlist</a> . See <a href="#">Byte-Range Playlists</a> in <i>Auto-Record to Amazon S3 (Low-Latency Streaming)</i> .	✓		

Term	Description	LL	RT	Chat
CBR	Constant Bitrate, a rate-control method for encoders that maintains a consistent bitrate throughout the entire playback of a video, regardless of what is happening during the broadcast. Lulls in the action may be padded to achieve the desired bitrate, and peaks may be quantized by adjusting the quality of encoding to match the target bitrate. <i>We strongly recommend using CBR instead of <a href="#">VBR</a>.</i>	✓	✓	
CDN	Content Delivery Network or Content Distribution Network, a geographically distributed solution that optimizes delivery of content such as streaming video by bringing it closer to where users are located.	✓		
Channel	An IVS resource that stores configuration for streaming, including an <a href="#">ingest server</a> , a <a href="#">stream key</a> , a <a href="#">playback URL</a> , and recording options. Streamers use the stream key associated with a channel to start a broadcast. All metrics and <a href="#">events</a> generated during a broadcast are associated with a channel resource.	✓		
Channel type	Determines the allowable <a href="#">resolution</a> and <a href="#">frame rate</a> for the <a href="#">channel</a> . See <a href="#">Channel Types</a> in the <i>IVS Low-Latency Streaming API Reference</i> .	✓		
Chat logging	An advanced option that can be enabled by associating a logging configuration with a <a href="#">chat room</a> .			✓

Term	Description	LL	RT	Chat
Chat room	An IVS resource that stores configuration for a chat session, including optional features such as <a href="#">Message Review Handler</a> and <a href="#">Chat Logging</a> . See <a href="#">Step 2: Create a Chat Room</a> in <i>Getting Started with IVS Chat</i> .			✓
Client-side composition	Uses a <a href="#">host</a> device to mix audio and video streams from stage participants and then sends them as a composite stream to an IVS <a href="#">channel</a> . This allows more control over the look of the <a href="#">composition</a> at the cost of higher utilization of client resources and a higher risk of a <a href="#">stage</a> or a <a href="#">host</a> issue impacting the viewers.  Also see <a href="#">server-side composition</a> .	✓	✓	
CloudFront	A <a href="#">CDN</a> service provided by Amazon.	✓		
CloudTrail	An AWS service for collecting, monitoring, analyzing, and retaining events and account activity from AWS and external sources. See <a href="#">Logging IVS API Calls with AWS CloudTrail</a> .	✓	✓	✓
CloudWatch	An AWS service for monitoring applications, responding to performance changes, optimizing resource use, and providing insights into operational health. You can use CloudWatch to monitor IVS metrics; see <a href="#">Monitoring IVS Real-Time Streaming</a> and <a href="#">Monitoring IVS Low-Latency Streaming</a> .	✓	✓	✓
Composition	The process of combining audio and video streams from multiple sources into a single stream.	✓	✓	
Composition pipeline	A sequence of processing steps required to combine multiple streams and encode the resulting stream.	✓	✓	

Term	Description	LL	RT	Chat
Compression	Encoding of information using fewer bits than the original representation. Any particular compression is either lossless or lossy. Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by removing unnecessary or less important information.	✓	✓	
Control plane	Stores information about IVS resources such as <a href="#">channels</a> , <a href="#">stages</a> , or <a href="#">chat rooms</a> and provides interfaces for creating and managing these resources. It is regional (based on AWS <a href="#">regions</a> ).	✓	✓	✓
CORS	Cross-Origin Resource Sharing, an AWS feature that allows client web applications that are loaded in one domain to interact with resources such as <a href="#">S3 buckets</a> in a different domain. Access can be configured based on headers, HTTP methods, and origin domains. See <a href="#">Using cross-origin resource sharing (CORS) - Amazon Simple Storage Service</a> in <i>Amazon Simple Storage Service User Guide</i> .	✓		
Custom image source	An interface provided by the IVS Broadcast <a href="#">SDK</a> that allows an application to provide its own image input instead of being limited to the preset cameras.	✓	✓	
Data plane	The infrastructure that carries data from <a href="#">ingest</a> to egress. It operates based on the configuration managed in the <a href="#">control plane</a> and is not restricted to an AWS region.	✓	✓	✓
Encoder, encoding	The process of converting video and audio content into a digital format, suitable for streaming. Encoding can be hardware or software based.	✓	✓	



Term	Description	LL	RT	Chat
Event	An automatic notification published by IVS to the AmazonEventBridge monitoring service. An event represents a state or health change of a streaming resource such as a <a href="#">stage</a> or a <a href="#">composition pipeline</a> . See <a href="#">Using Amazon EventBridge with IVS Low-Latency Streaming</a> and <a href="#">Using Amazon EventBridge with IVS Real-Time Streaming</a> .	✓	✓	✓
FFmpeg	A free and open-source software project consisting of a suite of libraries and programs for handling video and audio files and streams. <a href="#">FFmpeg</a> provides a cross-platform solution to record, convert and stream audio and video.	✓		
Fragmented stream	Created when a broadcast disconnects and then reconnects within the interval specified in the <a href="#">channel's</a> recording configuration. The resulting multiple streams are considered a single broadcast and merged together into a single recorded stream. See <a href="#">Merge Fragmented Streams in Auto-Record to Amazon S3 (Low-Latency Streaming)</a> .	✓		
Frame rate	A streaming metric for the number of video frames transmitted or received per second.	✓	✓	
HLS	HTTP Live Streaming (HLS), an HTTP-based <a href="#">adaptive bitrate streaming</a> communications protocol used to deliver IVS streams to viewers.	✓		
HLS playlist	A list of media segments that make up a stream. Standard HLS playlists are made up of 10-second media files. HLS also supports more granular <a href="#">byte-range playlists</a> .	✓		
Host	A real-time event <a href="#">participant</a> who sends video and/or audio to the stage.		✓	

Term	Description	LL	RT	Chat
IAM	Identity and Access Management, an AWS service that allows users to securely manage identities and access to AWS services and resources, including IVS.	✓	✓	✓
Ingest	IVS process for receiving video streams from a host or broadcaster for processing or delivery to viewers or other participants.	✓	✓	
Ingest server	<p>Receives video streams and delivers them to a transcoding system, where streams are <a href="#">transmuxed</a> or <a href="#">transcoded</a> into <a href="#">HLS</a> for delivery to viewers.</p> <p>Ingest servers are specific IVS components that receive streams for <a href="#">channels</a>, along with an ingestion protocol (<a href="#">RTMP</a>, <a href="#">RTMPS</a>). See the information on creating a channel in <a href="#">Getting Started with IVS Low-Latency Streaming</a>.</p>		✓	
Interlaced video	Transmits and displays only odd or even lines of subsequent frames to create perceived doubling of <a href="#">frame rate</a> without consuming extra bandwidth. We do not recommend using interlaced video due to the video quality concerns.	✓	✓	
JSON	JavaScript Object Notation, an open-standard file format that uses human-readable text to transmit data objects consisting of attribute-value pairs and array data types or other serializable values.	✓	✓	✓

Term	Description	LL	RT	Chat
Keyframe, delta frame, keyframe interval	The keyframe (also referred to as intra-coded or i-frame) is a full frame of the image in a video. Subsequent frames, the delta frames (also referred to as predicted or p-frames), only contain the information that has changed. Keyframes will appear multiple times within a <a href="#">stream</a> , depending on the keyframe interval defined in the encoder.	✓	✓	
Lambda	An AWS service for running code (referred to as Lambda functions) without provisioning any server infrastructure. Lambda functions can run in response to events and invocation requests, or based on a schedule. For example, IVS Chat uses Lambda functions to enable <a href="#">message review</a> for a <a href="#">chat room</a> .	✓	✓	✓
Latency, glass-to-glass latency	A delay in data transfer. IVS defines latency ranges as: <ul style="list-style-type: none"> <li>• Low latency: under 3 sec</li> <li>• Real-time latency: under 300 ms</li> </ul> <p><i>Glass-to-glass</i> latency refers to the delay from when a camera captures a live stream to when the stream appears on a viewer's screen.</p>	✓	✓	
Layered encoding with simulcast	Enables simultaneous encoding and publishing of multiple video streams with different quality levels. See <a href="#">Adaptive Streaming: Layered Encoding with Simulcast</a> in <i>Real-Time Streaming Optimizations</i> .		✓	

Term	Description	LL	RT	Chat
Message review handler	Enables IVS Chat customers to automatically review/filter user chat messages before they are delivered to the <a href="#">chat room</a> . It is enabled by associating a <a href="#">Lambda</a> function with a chat room. See <a href="#">Creating a Lambda Function</a> in <i>Chat Message Review Handler</i> .			✓
Mixer	A feature of the IVS Mobile Broadcast <a href="#">SDKs</a> that takes multiple audio and video sources and generates a single output. It supports management of on-screen video and audio elements representing sources such as cameras, microphones, screen captures, and audio and video generated by the application. The output can then be streamed to IVS. See <a href="#">Configuring a Broadcast Session for Mixing</a> in <i>IVS Broadcast SDK: Mixer Guide (Low-Latency Streaming)</i> .	✓		
Multi-host streaming	Combines streams from multiple <a href="#">hosts</a> into a single stream. This can be accomplished using either <a href="#">client-side</a> or <a href="#">server-side composition</a> .  Multi-host streaming enables scenarios such as inviting viewers onto a stage for Q&A, competitions between hosts, video chat, and hosts conversing with each other in front of a large audience.		✓	
Multivariant playlist	An index of all the <a href="#">variant streams</a> available for a broadcast.	✓		
OAC	Origin Access Control, a mechanism for restricting access to an <a href="#">S3 bucket</a> , so that content such as a recorded stream can be served only through <a href="#">CloudFront CDN</a> .	✓		

Term	Description	LL	RT	Chat
OBS	Open Broadcaster Software, free and open source software for video recording and live streaming . <a href="#">OBS</a> offers an alternative (to the IVS broadcast <a href="#">SDK</a> ) for desktop publishing. More sophisticated streamers familiar with OBS may prefer it for its advanced production features, such as scene transitions, audio mixing, and overlay graphics.	✓	✓	
Participant	A real-time user connected to a stage as a <a href="#">host</a> or <a href="#">viewer</a> .		✓	
Participant token	Authenticates a real-time event <a href="#">participant</a> when they join a <a href="#">stage</a> . A participant token also controls whether a participant can send video to the stage.		✓	
Playback token, playback key pair	An authorization mechanism that allows customers to restrict video playback on <a href="#">private channels</a> . Playback tokens are generated from a playback key pair.  A playback key pair is the public-private pair of keys used to sign and validate the viewer authorization token for playback. See <a href="#">Create or Import a Playback Key</a> in <i>Setting up Private Channels</i> and see the Playback Key Pair endpoints in the <a href="#">IVS Low-Latency API Reference</a> .	✓		
Playback URL	Identifies the address a viewer uses to start playback for a specific <a href="#">channel</a> . This address can be used globally. IVS automatically selects the best location on the IVS global <a href="#">content delivery network</a> for delivering the video to each <a href="#">viewer</a> . See the information on creating a channel in <a href="#">Getting Started with IVS Low-Latency Streaming</a> .	✓		

Term	Description	LL	RT	Chat
Private channel	Allows customers to restrict access to their streams using an authorization mechanism based on <a href="#">playback tokens</a> . See <a href="#">Workflow for Private Channels</a> in <i>Setting up Private Channels</i> .	✓		
Progressive video	Transmits and displays all lines of each frame in sequence. We recommend using progressive video during all stages of a broadcast.	✓	✓	
Quotas	The maximum numbers of IVS service resources or operations for your AWS account. That is, these limits are per AWS account, unless noted otherwise. All quotas are enforced per region. See <a href="#">Amazon Interactive Video Service endpoints and quotas</a> in <i>AWS General Reference Guide</i> .	✓	✓	✓
Regions	<p>Provide access to AWS services that physically reside in a specific geographic area. Regions provide fault tolerance, stability, and resilience, and can also reduce latency. With Regions, you can create redundant resources that remain available and unaffected by a regional outage.</p> <p>Most AWS service requests are associated with a particular geographic region. The resources that you create in one region do not exist in any other region unless you explicitly use a replication feature offered by an AWS service. For example, Amazon S3 supports cross-region replication. Some services, such as <a href="#">IAM</a>, do not have cross-regional resources.</p>	✓	✓	✓
Resolution	Describes the number of pixels in a single video frame, for example, Full HD or 1080p defines a frame with 1920x1080 pixels.	✓	✓	

Term	Description	LL	RT	Chat
Root user	The owner of an AWS account. The root user has complete access to all AWS services and resources in the AWS account.	✓	✓	✓
RTMP, RTMPS	Real-Time Messaging Protocol, an industry standard for transmitting audio, video, and data over a network. RTMPS is the secure version of RTMP, running over a Transport Layer Security (TLS/SSL) connection.	✓	✓	
S3 bucket	A collection of objects stored in Amazon S3. Many policies, including access and replication, are defined at the bucket level and apply to all objects in the bucket. For example, an IVS broadcast is stored as multiple objects in an S3 bucket.	✓		
SDK	Software Development Kit, a collection of libraries for the developers building applications with IVS.	✓	✓	✓
Selfie segmentation	Enables replacing the background in a live stream, using a client-specific solution that accepts a camera image as input and returns a mask that provides a confidence score for each pixel of the image, indicating whether it is in the foreground or the background. See <a href="#">Background Replacement</a> in <i>IVS Broadcast SDK: Third-Party Camera Filters (Real-Time Streaming)</i> .		✓	
Semantic versioning	A version format in the form of Major.Minor.Patch. Bug fixes not affecting the API increment the patch version, backward compatible API additions /changes increment the minor version, and backward incompatible API changes increment the major version.	✓	✓	✓

Term	Description	LL	RT	Chat
Server-side composition	<p>Uses an IVS server to mix audio and video from stage participants and then sends this mixed video to an IVS <a href="#">channel</a> to reach a larger audience or to store it in an <a href="#">S3 bucket</a>. Server-side composition reduces client load, improves resilience of the broadcast, and enables more efficient use of bandwidth.</p> <p>Also see <a href="#">client-side composition</a>.</p>		✓	
Service quotas	<p>An AWS service that helps you manage your <a href="#">quotas</a> for many AWS services from one location. Along with looking up the quota values, you can also request a quota increase from the Service Quotas console.</p>	✓	✓	✓
Service-linked role	<p>A unique type of <a href="#">IAM</a> role that is linked directly to an AWS service. Service-linked roles are automatically created by IVS and include all the permissions that the service requires to call other AWS services on your behalf, for example, to access an <a href="#">S3 bucket</a>. See <a href="#">Using Service-Linked Roles for IVS in IVS Security</a>.</p>	✓		
Stage	<p>An IVS resource that represents a virtual space where real-time event participants can exchange video in real time. See <a href="#">Create a Stage</a> in <i>Getting Started with IVS Real-Time Streaming</i>.</p>		✓	
Stage session	<p>Begins when the first participant joins a <a href="#">stage</a> and ends a few minutes after the last participant stops publishing to the stage. A long-lived stage may have multiple sessions over its lifetime.</p>		✓	
Stream	<p>Data representing video or audio content being sent continuously from a source to a destination.</p>	✓	✓	



Term	Description	LL	RT	Chat
Stream key	An identifier assigned by IVS when you create a <a href="#">channel</a> ; it is used to authorize streaming to the channel. <b>Treat the stream key like a secret, since anyone with it can stream to the channel.</b> See <a href="#">Getting Started with IVS Low-Latency Streaming</a> .	✓		
Stream starvation	A delay or halt in stream delivery to IVS. It occurs when IVS does not receive the expected amount of bits that the encoding device advertised it would send over a certain timeframe. An occurrence of stream starvation results in a stream starvation <a href="#">event</a> .  From a viewer's perspective, stream starvation may appear as video that lags, buffers, or freezes. Stream starvation can be brief (less than 5 seconds) or long (several minutes), depending on the specific situation that resulted in stream starvation. See <a href="#">What is Stream Starvation</a> in <i>Troubleshooting FAQ</i> .	✓	✓	
Streamer	A person or a device sending a video or audio <a href="#">stream</a> to IVS.	✓	✓	
Subscriber	A real-time event participant who receives video and/or audio of the hosts. See <a href="#">What is IVS Real-Time Streaming</a> .		✓	
Tag	A metadata label that you assign to an AWS resource. Tags can help you identify and organize your AWS resources. On the <a href="#">IVS documentation landing page</a> , see "Tagging" in any of the IVS API documentation (for real-time streaming, low-latency streaming, or chat).	✓	✓	✓

Term	Description	LL	RT	Chat
Third-party camera filters	Software components that can be integrated with the IVS Broadcast <a href="#">SDK</a> to allow an application to process images before providing them to the Broadcast SDK as a <a href="#">custom image source</a> . A third-party camera filter may process images from the camera, apply a filter effect, etc.	✓	✓	
Thumbnail	A reduced-size image taken from a stream. By default, thumbnails are generated every 60 seconds, but a shorter interval can be configured. Thumbnail resolution depends on the <a href="#">channel type</a> . See <a href="#">Recording Contents</a> in <i>Auto-Record to Amazon S3 (Low-Latency Streaming)</i> .	✓		
Timed metadata	Metadata tied to specific timestamps within a stream. It can be added programmatically using the IVS API and becomes associated with specific frames. This ensures that all viewers receive the metadata at the same point relative to the stream.  Timed metadata can be used to trigger actions on the client such as updating team statistics during a sporting event. See <a href="#">Embedding Metadata within a Video Stream</a> .	✓		
Transcoding	Converts video and audio from one format to another. An incoming stream may be transcoded to a different format at multiple bitrates and resolutions, to support a range of playback devices and network conditions.	✓	✓	

Term	Description	LL	RT	Chat
Transmuxing	A simple repackaging of an <a href="#">ingested</a> stream to IVS, with no re-encoding of the video stream. "Transmux" is short for transcode multiplexing, a process that changes the format of an audio and/or video file while keeping some or all of the original streams. Transmuxing converts to a different container format without changing the file contents. Distinguished from <a href="#">transcoding</a> .	✓	✓	
Variant streams	<p>A set of encodings of the same broadcast in several distinct quality levels. Each variant stream is encoded as a separate <a href="#">HLS playlist</a>. An index of the available variant streams is referred to as a <a href="#">multivariant playlist</a>.</p> <p>After the IVS player receives a multivariant playlist from IVS, it can then choose between the variant streams during playback, changing back and forth seamlessly as network conditions change.</p>	✓		
VBR	Variable Bitrate, a rate-control method for encoders that uses a dynamic bitrate that changes throughout playback, depending on the level of detail needed. We strongly recommend against using VBR due to video-quality concerns; use <a href="#">CBR</a> instead.	✓	✓	

Term	Description	LL	RT	Chat
View	<p>A unique viewing session which is actively downloading or playing video. Views are the basis for the concurrent views <a href="#">quota</a>.</p> <p>A view starts when a viewing session begins video playback. A view ends when a viewing session stops video playback. Playback is the sole indicator of viewership; engagement heuristics such as audio levels, browser tab focus, and video quality are not considered. When counting views, IVS does not consider the legitimacy of individual viewers or try to deduplicate localized viewership, such as multiple video players on a single machine. See <a href="#">Other Quotas</a> in <i>Service Quotas (Low-Latency Streaming)</i>.</p>	✓		
Viewer	A person receiving a <a href="#">stream</a> from IVS.	✓		
WebRTC	<p>Web Real-Time Communication, an open-source project providing web browsers and mobile applications with real-time communication. It allows audio and video communication to work inside web pages by allowing direct peer-to-peer communication, eliminating the need to install plugins or download native apps.</p> <p>The technologies behind <a href="#">WebRTC</a> are implemented as an open web standard and are available as regular JavaScript APIs in all major browsers or as libraries for native clients, like Android and iOS.</p>	✓	✓	

Term	Description	LL	RT	Chat
WHIP	<p>WebRTC-HTTP Ingestion Protocol, an HTTP based protocol that allows <a href="#">WebRTC</a> based <a href="#">ingestion</a> of content into streaming services and/or <a href="#">CDNs</a>. <a href="#">WHIP</a> is an IETF draft developed to standardize WebRTC ingestion.</p> <p>WHIP enables compatibility with software like <a href="#">OBS</a>, offering an alternative (to the IVS broadcast <a href="#">SDK</a>) for desktop publishing. More sophisticated streamers familiar with OBS may prefer it for its advanced production features, such as scene transitions, audio mixing, and overlay graphics</p> <p>WHIP is also beneficial in situations where using the IVS broadcast SDK isn't feasible or preferred . For example, in setups involving hardware encoders, the IVS broadcast SDK might not be an option. However, if the encoder supports WHIP, you can still publish directly from the encoder to IVS.</p> <p>See <a href="#">OBS and WHIP Support</a>.</p>		✓	
WSS	<p>WebSocket Secure, a protocol for establishing WebSockets over an encrypted TLS connection. It is being used for connecting to IVS Chat endpoints . See <a href="#">Step 4: Send and Receive Your First Message</a> in <i>Getting Started with IVS Chat</i>.</p>			✓

# Document History (Low-Latency Streaming)

## Low-Latency Streaming User Guide Changes

Change	Description	Date
<a href="#">Getting Started with IVS</a>	Updated <a href="#">Streaming with OBS Studio using RTMPS</a> to align with OBS Studio v30.2.	June 27, 2024
<a href="#">Broadcast SDK: Android 1.19.0, iOS 1.19.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the low-latency-streaming broadcast SDK guides: <a href="#">Android</a> and <a href="#">iOS</a> . Also see the <a href="#">Release Notes</a> .  In <i>IVS Broadcast SDK</i> , we updated " <a href="#">Native Platforms</a> " to change the iOS Supported Version to 13+ and deleted the iOS 12 deprecation notice.	June 13, 2024
<a href="#">Broadcast SDK: Web 1.13.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the low-latency-streaming broadcast SDK guide: <a href="#">Web</a> . Also see the <a href="#">Release Notes</a> .	June 13, 2024
<a href="#">Player SDK: Android 1.29.0, iOS 1.29.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the player SDK guides:	June 6, 2024

---

	<a href="#">Android</a> and <a href="#">iOS</a> . Also see the <a href="#">Release Notes</a> .	
<a href="#">Player SDK: Web 1.29.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the player SDK guides: <a href="#">Web</a> , <a href="#">Video.js Integration</a> , and <a href="#">JW Player Integration</a> . Also see the <a href="#">Release Notes</a> .	June 6, 2024
<a href="#">Broadcast SDK: Web 1.12.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the low-latency-streaming broadcast SDK guide: <a href="#">Web</a> . Also see the <a href="#">Release Notes</a> .	May 20, 2024
<a href="#">Broadcast SDK: Android 1.18.0, iOS 1.18.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the low-latency-streaming broadcast SDK guides: <a href="#">Android</a> and <a href="#">iOS</a> . Also see the <a href="#">Release Notes</a> .	May 16, 2024
<a href="#">Player SDK: Web 1.28.0</a>	Updated version number and artifact links on the <a href="#">IVS documentation landing page</a> and in the player SDK guides: <a href="#">Web</a> , <a href="#">Video.js Integration</a> , and <a href="#">JW Player Integration</a> . Also see the <a href="#">Release Notes</a> .	May 9, 2024

[Player SDK: Android 1.28.0, iOS 1.28.0](#)

Updated version number and artifact links on the [IVS documentation landing page](#) and in the player SDK guides: [Android](#) and [iOS](#). Also see the [Release Notes](#).

May 9, 2024

[Broadcast SDK: Web 1.11.0](#)

Updated version number and artifact links on the [IVS documentation landing page](#) and in the low-latency-streaming broadcast SDK guide: [Web](#). Also see the [Release Notes](#).

May 6, 2024

[Broadcast SDK: Web 1.10.1](#)

Updated version number and artifact links on the [IVS documentation landing page](#) and in the low-latency-streaming broadcast SDK guide: [Web](#). Also see the [Release Notes](#).

April 30, 2024

[Broadcast SDK: Android 1.15.2, iOS 1.15.2](#)

Updated version number and artifact links on the [IVS documentation landing page](#) and in the low-latency-streaming broadcast SDK guides: [Android](#) and [iOS](#). Also see the [Release Notes](#).

April 30, 2024



### [Broadcast SDK: Android 1.17.0, iOS 1.17.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Android](#) and [iOS](#). On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version. Also see the Amazon IVS [Release Notes](#) for this release.

April 22, 2024

### [Player SDK 1.27.0](#)

Updated version number and artifact links for the new release, in the player SDK guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#). On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions. Also see the Amazon IVS [Release Notes](#) for this release.

April 11, 2024

In the [Web Player Guide](#), we added more information on “Demos.”

[SRT additional information](#)

Added SRT information to:

April 10, 2024

- [What is IVS Low-Latency Streaming](#): See the introduction.
- [Getting Started with IVS Low-Latency Streaming](#): See "Final Channel Creation" and the introduction to "Step 5: Set Up Streaming Software."
- Streaming Configuration: See [Stream Ingest: Codecs and Ingest Protocols](#).

[Getting Started with Low-Latency Streaming](#)

In "Step 6: View Your Live Stream," we added a section on "Viewing with the Amazon IVS Player."

April 4, 2024

[Secure Reliable Transport ingest support](#)

IVS now supports H.264-encoded video content using the SRT protocol. In [Getting Started With IVS Low-Latency Streaming](#), in "Step 5: Set Up Streaming Software," we added new sections, "Streaming with OBS Studio using SRT" and "Streaming a Recorded Video with FFmpeg using SRT." Also see [API Reference Changes](#).

April 4, 2024

[Broadcast SDK: Android  
1.16.0, iOS 1.16.0, Web 1.10.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Android](#), [iOS](#), and [Web](#). On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version. Also see the Amazon IVS [Release Notes](#) for this release.

March 21, 2024

[Player SDK 1.26.0](#)

Updated version number and artifact links for the new release, in the player SDK guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#). On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions. Also see the Amazon IVS [Release Notes](#) for this release.

March 14, 2024

---

<a href="#">Broadcast SDK: Android 1.15.1, iOS 1.15.1</a>	Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: <a href="#">Android</a> and <a href="#">iOS</a> . On the <a href="#">Amazon IVS documentation landing page</a> , updated the broadcast SDK Reference links to point to the new version. Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	March 13, 2024
<a href="#">Monitoring</a>	Changed the unit for <code>IngestFramerate</code> from Frames per second to Count/Second.	March 11, 2024
<a href="#">Setting Up Private Channels</a>	In <a href="#">Token Schema</a> , added a note about the maximum number of domains for <code>access-control-all-ow-origin</code> .	March 11, 2024
<a href="#">Player SDK: Web 1.25.0</a>	Updated version number and artifact links for the new release, in the player SDK guides: <a href="#">Web</a> , <a href="#">Video.js Integration</a> , and <a href="#">JW Player Integration</a> . On the <a href="#">Amazon IVS documentation landing page</a> , updated the player SDK Reference links to point to the new versions. Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	February 29, 2024

---

<a href="#">iOS 12 support deprecation for Player and Broadcast SDKs</a>	Added a deprecation notice for iOS 12, in the "Native Platforms" tables of <a href="#">Player SDK overview</a> and <a href="#">Broadcast SDK overview (low-latency streaming)</a> .	February 23, 2024
<a href="#">Broadcast SDK: Android 1.15.0, iOS 1.15.0, Web 1.9.0</a>	Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: <a href="#">Android</a> , <a href="#">iOS</a> , and <a href="#">Web</a> . On the <a href="#">Amazon IVS documentation landing page</a> , updated the broadcast SDK Reference links to point to the new version. Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	February 22, 2024
<a href="#">IAM managed policies</a>	Added entry to Policy Updates table in <a href="#">Managed Policies for Amazon IVS</a> to reflect updates to IVSReadOnlyAccess for Server-Side Composition, Real-Time Composite Recording, and Tokenless Playback Restrictions.	February 16, 2024

### [Player SDK: Mobile 1.25.0](#)

Updated version number and artifact links for the new release, in the player SDK guides: [Android](#) and [iOS](#). On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions. Also see the Amazon IVS [Release Notes](#) for this release.

February 15, 2024

### [Service Quotas](#)

In the "API Call Rate Quotas" table, we added StartViewerSessionRevocation and BatchStartViewerSessionRevocation . (These are not new endpoints but were missing from the table.) They are in the same part of the table as the playback-key-pair endpoints; the **Endpoint Type** is "Private channel."

February 5, 2024

[Broadcast SDK: Android 1.14.1, iOS 1.14.1, Web 1.8.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Android](#), [iOS](#), and [Web](#). On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version. Also see the Amazon IVS [Release Notes](#) for this release.

February 1, 2024

For the Android Guide, we added a new Known Issue (video size less than 176x176).

## [Tokenless playback restrictions](#)

This release enables origin enforcement and geofencing outside of playback authorization. Several low-latency streaming documents were changed:

January 31, 2024

- [Getting Started](#) - Updated "Step 4: Create a Channel" and "Step 8: Prevent Undesired Content and Viewers."
- [Service Quotas](#) - Added TPS limits for new endpoints , and in "Other Quotas," added new quotas.
- [Undesired Content and Viewers](#) - Added "Use Playback Restriction Policies."
- [Private Channels](#) - Updated the location of Playback Keys on the console navigation pane.

Also see [API changes](#).

## [Audio-only playback](#)

Added [Audio-Only Playback](#) to the Player overview.

January 25, 2024



## [Player SDK 1.24.0](#)

Updated version number and artifact links for the new release, in the player SDK guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#). On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions. Also see the Amazon IVS [Release Notes](#) for this release.

January 18, 2024

In the Web guide, we added a new section, "Audio-Only Playback," and deleted the "Known Issue" about lack of support for the audio-only rendition.

## [Troubleshooting Auto-Record to Amazon S3](#)

In Troubleshooting, we added a section, [Can KMS-S3 encryption be used with auto-record to S3?](#)

January 4, 2024

## [Broadcast SDK: Android 1.13.4, iOS 1.13.4, Web 1.7.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Android](#), [iOS](#), and [Web](#). On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version. Also see the Amazon IVS [Release Notes](#) for this release.

January 3, 2024

---

<a href="#">Split out a Chat UG</a>	Major documentation changes accompany this release. We moved chat information from the IVS Low-Latency Streaming User Guide to a new IVS Chat User Guide, located in the existing IVS Chat section of the <a href="#">IVS documentation landing page</a> .	December 28, 2023
	For other documentation changes, see <a href="#">Document History (Chat)</a> .	
<a href="#">IVS Glossary</a>	Extended the glossary, covering IVS real-time, low-latency, and chat terms.	December 20, 2023
<a href="#">IAM managed policies</a>	Added two managed policies, IVSReadOnlyAccess and IVSFullAccess. See: <ul style="list-style-type: none"><li>• The new section on <a href="#">Managed Policies for Amazon IVS</a> on the <i>Security</i> page.</li><li>• Changes to <a href="#">Step 3: Set Up IAM Permissions</a> in <i>Getting Started with IVS Low-Latency Streaming</i>.</li></ul>	December 5, 2023

[Broadcast SDK: Android  
1.13.2, iOS 1.13.2](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Android](#) and [iOS](#).

December 4, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

[Broadcast SDK: Android  
1.13.1](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guide: [Android](#).

November 21, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

[Broadcast SDK: Android 1.13.0, iOS 1.13.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Android](#) and [iOS](#).

November 17, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

[Auto-Record to S3](#)

In [Merge Fragmented Streams > Eligibility](#), we added a bullet: "Each stream must start 10 seconds or more after the previous stream."

November 17, 2023

[Server-side composition and real-time composite recording](#)

In [Enabling Multiple Hosts on an IVS Stream](#), we added "Broadcasting a Stage: Client-Side versus Server-Side Composition" and updated "4. Broadcast the Stage."

November 16, 2023

In [Security](#), we added S3 endpoints to the policy in "Identity-Based Policy Examples > Use the Amazon IVS Console."

For additional changes, see [Document History \(Real-Time Streaming\)](#).

---

<a href="#">Player SDK 1.23.0</a>	Updated version number and artifact links for the new release, in the player SDK guides: <a href="#">Web</a> , <a href="#">Android</a> , <a href="#">iOS</a> , <a href="#">Video.js Integration</a> , and <a href="#">JW Player Integration</a> .	November 14, 2023
	On the <a href="#">Amazon IVS documentation landing page</a> , updated the player SDK Reference links to point to the new versions.	
	Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	
<a href="#">IVS player and broadcast SDKs</a>	In the <a href="#">Player overview</a> and <a href="#">Broadcast SDK overview</a> , we updated Platform Requirements > Native Platforms to clarify which SDK versions are supported.	November 9, 2023
<a href="#">Getting Started with IVS Low-Latency Streaming</a>	We updated procedures in <a href="#">Set Up IAM Permissions</a> .	October 20, 2023

## [Broadcast SDK: Web 1.6.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guide: [Web](#).

October 16, 2023

The [Amazon IVS documentation landing page](#) points to the current version of Broadcast SDK References.

Also see the Amazon IVS [Release Notes](#) for this release.

In the Web Guide, in "Retrieve a MediaStream from a Device," we also deleted the two max lines; best practice is to specify only `ideal`.

## [Monitoring IVS Low-Latency Streaming](#)

Renamed the "Monitoring Live Stream Health" page and added information from "Monitoring IVS with CloudWatch" (which has been deleted as a separate page). Updated the CloudWatch console instructions.

October 12, 2023

## [Broadcast SDK: Android](#) [1.12.1](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guide: [Android](#). Also added a new section, [Using Bluetooth Microphones](#).

October 12, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

## [Player SDK 1.22.0](#)

Updated version number and artifact links for the new release, in the player SDK guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

October 3, 2023

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

## [In-console streaming](#)

In *Getting Started with Low-Latency Streaming*, we added in-console streaming to [Step 5: Set Up Streaming Software](#).

October 2, 2023

---

<a href="#">Broadcast SDK: Mixer Guide</a>	Added <a href="#">Mirroring the Broadcast</a> , with Android and iOS examples.	September 18, 2023
<a href="#">Broadcast SDK: Web 1.5.2</a>	Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guide: <a href="#">Web</a> .  The <a href="#">Amazon IVS documentation landing page</a> points to the current version of Broadcast SDK References.  Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	September 14, 2023
<a href="#">Undesired Content</a>	Split out existing content from Troubleshooting FAQs into its own top-level page.  In <a href="#">Getting Started with IVS Low-Latency Streaming</a> , added "Step 8: Prevent Undesired Content (Recommended)."	September 8, 2023
<a href="#">Auto-Record to Amazon S3</a>	In <a href="#">Byte-Range Playlists</a> , clarified that segment duration is the same as the keyframe interval configured for the stream (not a fixed duration of approximately 2 seconds).	August 25, 2023



[Broadcast SDK: Web 1.5.1, Android 1.12.0, and iOS 1.12.0](#)

Updated version number and artifact links for the new release, in the low-latency-streaming broadcast SDK guides: [Web](#), [Android](#), and [iOS](#).

August 23, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

[Player SDK 1.21.0](#)

Updated version number and artifact links for the new release, in the player SDK guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

August 22, 2023

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

[Channel-type definitions](#)

Updated channel-type definitions to provide more detail, especially about rendered transcode ladders. See [Channel Types](#) in *IVS Streaming Configuration*.

August 18, 2023

## [Real-time streaming launch](#)

Major documentation changes August 7, 2023

accompany this release.

We renamed the previous documentation to be IVS Low-Latency Streaming and published new IVS Real-Time Streaming documentation. The [IVS documentation landing page](#) now has separate sections for real-time streaming and low-latency streaming. Each section has its own User Guide and API Reference.

We moved some information from the IVS Low-Latency User Guide to the new IVS Real-Time User Guide:

- Most information about stages and multiple hosts.
- Monitoring Stage Health is now [Monitoring Real-Time Streaming](#).

For other documentation changes, see:

- [Stage API Reference Changes](#)
- [Document History \(Real-Time Streaming\)](#)

<a href="#">Broadcast SDK: Web 1.5.0, Android 1.11.0, and iOS 1.11.0</a>	Updated version number and artifact links for the new release, in the broadcast SDK guides: <a href="#">Web</a> , <a href="#">Android</a> , and <a href="#">iOS</a> .	August 7, 2023
	On the <a href="#">Amazon IVS documentation landing page</a> , updated the broadcast SDK Reference links to point to the new version.	
	Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	
<a href="#">Setting Up Private Channels</a>	In <a href="#">Token Schema</a> , added clarifying information on the exp field.	July 31, 2023
<a href="#">Security: getting IVS status information</a>	In <a href="#">Incident Response</a> , updated information on getting IVS status, to point to the AWS Health Dashboard.	July 31, 2023
<a href="#">Auto-Record to Amazon S3: OAC and CORS</a>	In <a href="#">Playback of Recorded Content from Private Buckets</a> , replaced origin access identity (OAI) with origin access control (OAC). Also added information about configuring the S3 bucket for CORS, to play back recorded streams.	July 31, 2023
<a href="#">Resources and Support</a>	In "Partner Solutions" > "Face and Background Filters," added a paragraph on Camera Kit.	July 21, 2023

[Broadcast SDK: Android Guide](#)

Minor changes. In the introduction, mentioned that there is no support for emulators. In "Create the Player and Set Up Event Listener" changed `PlayerActivity` class to `Activity`. In "Thread Safety" changed the text.

July 21, 2023

## [R2S3 rendition filtering and thumbnail enhancements](#)

July 17, 2023

IVS customers can now control what renditions are generated for a stream when recording to Amazon S3 and what resolutions are generated for thumbnails. In the IVS User Guide, see:

- [Getting Started with IVS](#) – In "Step 4: Create a Channel" > "Console Instructions," we updated screenshots and instructions.
- [Auto-Record to Amazon S3](#) – In "JSON Metadata Files," we added `latest_thumbnail` and updated `thumbnail`. In "Thumbnails" and "Discovering the Renditions of a Recording," we added rendition-resolution descriptions.
- [Costs](#) – In "Storing Recorded Video," we updated screenshots.

Also see [IVS API Reference Changes](#).

## [Player SDK 1.20.0](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

July 14, 2023

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

## [Getting Started with IVS](#)

In [How to Disable Recording](#), fixed the CLI example.

July 14, 2023

## [Broadcast SDK: Web 1.4.0, Android 1.10.0, and iOS 1.10.0](#)

Updated version number and artifact links for the new release, in the broadcast SDK guides: [Web](#), [Android](#), and [iOS](#).

July 13, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Viewer session revocation for private channels](#)

IVS customers can now revoke the viewer session associated with an auth token, to prevent and stop playback using that token. For more information, see [Setting Up Private Channels](#):

- "Token Schema" – We added `viewer-id` and modified `viewer-session-version`.
- "Revoke Viewer Sessions" – New section.

Also see [IVS API Reference Changes](#).

### [Security TLS update](#)

In "Infrastructure Security" > "[API Calls](#)," updated the TLS version to 1.2 minimum and 1.3 recommended.

### [Broadcast SDK: iOS 1.9.1 and iOS 1.7.5](#)

Updated version number and artifact links for the new release in the broadcast SDK guide: [iOS](#).

The [Amazon IVS documentation landing page](#) points to the latest version of the Broadcast SDK Reference.

Also see the Amazon IVS [Release Notes](#) for this release.

[Broadcast SDK: Web 1.3.3](#)

Updated version number and artifact links for the new release in the broadcast SDK guide: [Web](#).

June 16, 2023

The [Amazon IVS documentation landing page](#) points to the latest version of the Broadcast SDK Reference.

Also see the Amazon IVS [Release Notes](#) for this release.



## [Advanced channel types](#)

Introduced two new channel types, `ADVANCED_SD` and `ADVANCED_HD`. We updated several pages:

June 2, 2023

- [Player SDK Overview](#) – In "Reducing Latency in Third-Party Players," noted that the reducing-latency feature isn't required with Advanced streams
- [Broadcast Web SDK Guide](#) – Changes in "Create an Instance of the AmazonIVS BroadcastClient."
- [Broadcast Android SDK Guide](#) – Change in "Get Recommended Broadcast Settings."
- [Broadcast iOS SDK Guide](#) – Change in "Get Recommended Broadcast Settings."
- [Service Quotas](#) – In Other Quotas > IVS, added two rows for "Ingest bitrate" for the new channel types.
- [Streaming Configuration](#) – Changes in "Channel Types."
- [Costs](#) – Added the new channel types and mentioned the "Help me choose" tool

## [Broadcast SDK: Android 1.9.0 & iOS 1.9.0](#)

Updated version number and artifact links for the new release, in the broadcast SDK guides: [Android](#) and [iOS](#).

June 1, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

In the [Broadcast SDK overview](#), updated supported iOS versions from 11+ to 12+ (for the SDK without stage functionality).

In the [iOS Guide](#), added a new section, "How iOS Chooses Camera Resolution and Frame Rate."

Also see the Amazon IVS [Release Notes](#) for this release.

## [Auto-Record to AmazonS3](#)

In "Example: recording \_ended.json," updated the byte\_range\_playlist value from byte-range-multivariant.m3u8 to byte-range-variant.m3u8 .

May 25, 2023

## [Player SDK 1.19.0](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

May 23, 2023

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

## [Broadcast SDK: iOS 1.8.1 and iOS 1.7.4](#)

Updated version number and artifact links for the new release in the broadcast SDK guide: [iOS](#).

May 16, 2023

The [Amazon IVS documentation landing page](#) points to the latest version of the Broadcast SDK Reference.

Also see the Amazon IVS [Release Notes](#) for this release.

## [Monitoring stage health](#)

Added [Monitoring Stage Health](#), a new User Guide page for new Amazon IVS functionality. For Stage Health, we also:

May 11, 2023

- Added Health information to [Enabling Multiple Hosts on an IVS Stream](#).
- Added two Stage Update events to [Using Amazon EventBridge with Amazon IVS](#).
- Added call-rate quotas for the new endpoints to [IVS Service Quotas](#).

Note: With the IVS real-time streaming launch on Aug 2, 2023, this document was renamed "Monitoring Amazon IVS Real-Time Streaming" and moved to the new *IVS Real-Time Streaming user Guide*.

## [Stage participant limits](#)

In [Service Quotas](#), deleted the "stage participants" limit. This is superceded by the limits for subscriber and publisher participants.

May 2, 2023

[Broadcast SDK: Web 1.3.2](#)

Updated version number and artifact links for the new release in the broadcast SDK guide: [Web](#).

May 1, 2023

The [Amazon IVS documentation landing page](#) points to the latest version of the Broadcast SDK Reference.

Also see the Amazon IVS [Release Notes](#) for this release.

[RTMP support: documentation errata](#)

Changed [Broadcast Android SDK Guide](#) and [Broadcast iOS SDK Guide](#) to indicate that these SDKs support only RTMPS ingest (not insecure RTMP ingest).

April 29, 2023

[Stage participant limits](#)

This release includes the following changes:

April 27, 2023

- [Enabling Multiple Hosts](#)
  - Updated the maximum number of stage participants from 12 to 1,000.
- [Service Quotas](#) – Updated the participant limit to 1,000 and added add new limits for subscriber and publisher participants. Changes TPSs for some endpoints.

---

<a href="#">IVS User Guide landing page</a>	On the <a href="#">What is IVS?</a> home page, we added sections for "Multiple Hosts" and "IVS Chat" and updated the section on "Latency."	April 27, 2023
<a href="#">Resources and Support</a>	In "Partner Solutions" > "Face and Background Filters," updated the DeepAR link.	April 25, 2023
<a href="#">Resources and Support</a>	Added a section on Partner Solutions.	April 17, 2023
<a href="#">Player SDK: Web Guide</a>	In "Known Issues and Workarounds," added an issue (the Web Player does not support the <code>audio_only</code> rendition).	April 17, 2023
<a href="#">Streaming Configuration</a>	In <a href="#">Closed Captioning</a> , added a link to a new blog post on captioning.	April 14, 2023

## [Broadcast Web SDK Guide](#)

Made miscellaneous updates: April 10, 2023

- In “Create an Instance of the AmazonIVSBroadcast Client,” added a note about making sure your client-side configuration aligns with the back-end channel type.
- In “Hide Video” code examples, changed `VIDEO_DEVICE_NAME` to `VIDEO_DEVICE_NAME.source`.
- In “Enabling Multiple Hosts,” changed `ConnectionState` references to `StageConnectionState`.
- In “Add Multiple Hosts with the Broadcast SDK” and “Known Issues,” synchronized information here and on [GitHub](#).

## [Streaming Configuration](#)

In [Video Settings](#), added a `ColorSpace` bullet. April 5, 2023

## [Enabling Multiple Hosts](#)

In [Set Up the AWS CLI](#), changed the stage namespace from `ivsrealtime` to `ivs-realtime`. April 5, 2023

## [Player SDK 1.18.0](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

April 4, 2023

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

## [RTMP support](#)

In several documents, indicated that both RTMP (insecure ingest) and RTMPS are now supported. Among other things, this affects the ingest endpoint; see [Set Up Streaming Software](#), [Broadcast Android SDK Guide](#), and [Broadcast iOS SDK Guide](#).

March 30, 2023

## [Setting Up Private Channels](#)

In [Generate and Sign Playback Tokens](#), added to the payload an optional field, `single-use-uuid`, for generating a single-use token.

March 29, 2023



[Broadcast SDK: Web 1.3.1](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [Web](#).

March 28, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

[Support for multiple hosts on a stream](#)

Added a new page, [Enabling Multiple Hosts on an IVS Stream](#). And in [Service Quotas](#), added "Amazon IVS Stage" endpoints and added stage limits to Other Quotas > Amazon IVS.

March 23, 2023

Also see [Stage API Reference Changes](#).

### [Broadcast SDK: Android 1.8.0, iOS 1.8.0, and Web 1.3.0](#)

Updated version number and artifact links for the new release, in the broadcast SDK guides: [Android](#), [iOS](#), and [Web](#).

March 23, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

In the [Broadcast SDK overview](#), added stage platform requirements.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Web Broadcast SDK](#)

In [Known Issues and Workarounds](#), added an issue: viewers of a Safari broadcast sometimes see green artifacts in the video feed.

March 17, 2023

### [Broadcast SDK: Android 1.7.3](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [Android](#).

March 2, 2023

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference link to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

<a href="#">Player SDK 1.17.0</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, <a href="#">Video.js Integration</a>, and <a href="#">JW Player Integration</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	February 28, 2023
<a href="#">Service Quotas</a>	Clarified that all quotas are enforced per region.	February 24, 2023
<a href="#">Troubleshooting FAQs</a>	In "Use Private Channels," clarified the use of private channels to prevent undesirable content. In "Broadcasting and Encoding," added two subsections on troubleshooting an IVS Web Broadcast SDK session and using Chrome WebRTC-internals.	February 17, 2023
<a href="#">Byte-range tags and manifest files for auto-record to S3</a>	In <a href="#">Auto-Record to Amazon S3</a> , updated "Recording Contents," added "Byte-Range Playlists" and new fields in JSON examples for recording <code>_started</code> and recording <code>_ended</code> .	February 16, 2023

---

<a href="#">Getting Started with IVS Chat</a>	In the beginning, say that IVS Chat also can be used on its own, without a video stream. See <a href="#">Getting Started with IVS Chat</a> in the <i>Amazon IVS Chat User Guide</i> .	February 9, 2023
<a href="#">Troubleshooting FAQs</a>	Added a new section on Undesirable Content.  Sept 8, 2023 update: This section was moved to <a href="#">Undesired Content</a> .	February 6, 2023
<a href="#">Player SDK overview</a>	In <a href="#">Browser &amp; Platform Requirements</a> , added a note that the Web SDK Video.js and Player JW integrations are not supported in browser-like environments.	February 6, 2023
<a href="#">Auto-Record to Amazon S3</a>	In the <a href="#">Eligibility</a> requirements for merge fragmented streams, changed the required bitrate difference from 10% to 50%.	February 6, 2023
<a href="#">Streaming configuration</a>	Revised <a href="#">Stream with the Amazon IVS Broadcast SDK</a> to include the Web Broadcast SDK (not just Android and iOS).	February 2, 2023

## [IVS Chat Client Messaging SDK: Android 1.1.0](#)

Updated version number and artifact links for the new release, in the Chat SDK Guide: [Android](#).

January 31, 2023

The [Amazon IVS documentation landing page](#) points to the current version of the SDK Reference.

Also see the Amazon IVS [Release Notes](#) for this release.

This release includes an extensive Chat Kotlin Coroutines tutorial, split into two parts:

- [Part 1: Chat Rooms](#)
- [Part 2: Messages and Events](#)

## [Chat Android SDK tutorial](#)

Added an extensive Android tutorial for the Chat Client Messaging SDK. The tutorial is split into two parts:

January 24, 2023

- [Part 1: Chat Rooms](#)
- [Part 2: Messages and Events](#)

## [Service Quotas](#)

Increased some Chat quotas: January 19, 2023

- TPS for CreateChatToken, DeleteMessage, DisconnectUser, and SendEvent Rooms
- Other Quotas: concurrent chat connections; rate of DeleteMessage, DisconnectUser, and SendMessage requests; rate of messaging requests per connection; and Rooms

## [Private Channels](#)

In [Token Schema](#), added the strict-origin-enforcement field to the token payload. January 17, 2023

## [Player SDK 1.16.0](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#). January 17, 2023

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

---

<a href="#">Chat React &amp; React Native Best Practices</a>	Added this new Chat page.  Note: On December 28, 2023, <a href="#">this document</a> was moved to the new <i>IVS Chat User Guide</i> .	January 13, 2023
<a href="#">Chat React Native SDK tutorial</a>	Added an extensive React Native tutorial for the Chat Client Messaging SDK. The tutorial is split into two parts: <ul style="list-style-type: none"><li>• <a href="#">Part 1: Chat Rooms</a></li><li>• <a href="#">Part 2: Messages and Events</a></li></ul>	January 10, 2023
<a href="#">Troubleshooting</a>	Added a new Troubleshooting FAQs page, describing best practices and troubleshooting tips.	January 6, 2023
<a href="#">Added timestamp to record-to-S3 manifest files</a>	Added a timestamp to S3 manifest files created by the auto-record to S3 feature. See the Amazon IVS <a href="#">Release Notes</a> .	December 9, 2022
<a href="#">Player SDK latency</a>	Added <a href="#">Reducing Latency in Third-Party Players</a> .	December 8, 2022
<a href="#">Broadcast Web SDK Guide</a>	Added content (previously only on GitHub) to this page.	December 8, 2022

<a href="#">Broadcast SDK: Android 1.7.2</a>	<p>Updated version number and artifact links for the new release, in the broadcast SDK guide: <a href="#">Android</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the broadcast SDK Reference link to point to the new version.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	December 6, 2022
<a href="#">IVS setup</a>	<p>In <a href="#">IVS Getting Started</a>, updated steps to create an AWS account and set up permissions. Added "Step 2: Set Up Root and Administrative Users."</p> <p>In <a href="#">Security</a>, made minor changes to the beginning of the IAM section.</p>	December 5, 2022
<a href="#">Chat: setup and iOS SDK tutorial</a>	<p>In <a href="#">Getting Started with IVS Chat</a>, updated and renamed "Initial Setup."</p> <p>Added a <a href="#">Chat iOS Tutorial</a> page to the User Guide, pointing to an existing tutorial on GitHub.</p>	December 5, 2022
<a href="#">Costs for Auto-Record to S3</a>	In <a href="#">Auto-Record to Amazon S3</a> , clarified costs.	December 2, 2022



---

<a href="#">Chat JavaScript SDK tutorial</a>	Added an extensive JS tutorial for the Chat Client Messaging SDK. The tutorial is split into two parts: <ul style="list-style-type: none"><li>• <a href="#">Part 1: Chat Rooms</a></li><li>• <a href="#">Part 2: Messages and Events</a></li></ul>	December 2, 2022
<a href="#">Web Player known issue</a>	In the Player Web SDK Guide, we added a <a href="#">Known Issue and Workaround</a> : when playing a muted live stream on an iOS mobile browser, player instability may be seen when resuming an inactive player tab.	November 18, 2022
<a href="#">Setting Up Private Channels</a>	In "Create or Import a Playback Key," reorganized content and clarified use of private and public keys. In "Generate and Sign Playback Tokens," clarified that you do not have to enter the public key in jwt.io.	November 18, 2022

## [Chat Logging](#)

Initial release of this new functionality. See these User Guide changes:

November 17, 2022

- [Chat Logging](#) – New page.
- [Getting Started with Chat](#) – Updated IAM permissions and added procedures for setting up chat logging.
- [Service Quotas](#) – Added limits for new endpoints and logging configurations.
- Cloudwatch – Added log-destination metrics.

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

Dec 28, 2023 update: Chat-related CloudWatch content was moved to [Monitoring Amazon IVS Chat](#).

---

<a href="#">Chat Client Messaging SDK: JavaScript 1.0.2</a>	Updated version number and artifact links for the new release, in the Chat SDK guide: <a href="#">JavaScript</a> .	November 9, 2022
	The <a href="#">Amazon IVS documentation landing page</a> points to the current version of the SDK Reference.	
	Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	
<a href="#">Split-view on live channels (for monitoring live stream health)</a>	In <a href="#">Access Stream Session Data</a> , added console instructions for accessing the new split view. This is a faster way to get session health data, right from the "Live channels" page.	November 8, 2022
<a href="#">Resources and Support</a>	Added a link to IVS blogs on the DEV community site.	November 7, 2022
<a href="#">Auto-Record to Amazon S3</a>	In "Merge Fragmented Streams" > " <a href="#">Eligibility</a> ," deleted the redundant bullet, "Source video quality must be the same."	November 7, 2022

<a href="#">Player SDK 1.14.0</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, <a href="#">Video.js Integration</a>, and <a href="#">JW Player Integration</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	November 1, 2022
<a href="#">Player SDK: Web Guide</a>	<p>Updated <a href="#">Working with Content Security Policy</a>, to reflect the fact that newer versions of all browsers have been updated to deal with new CSP rules. Deleted old sections on "Hosting Assets on the Same Origin" and "Hosting Assets on a Separate Origin."</p>	October 27, 2022
<a href="#">Getting Started with Amazon IVS Chat</a>	<p>Updated and clarified Step 3, formerly "Authenticate and Authorize Chat Clients," now <a href="#">Create a Chat Token</a>.</p>	October 27, 2022
<a href="#">Player SDK: Web Guide</a>	<p>In "Sample Code," added quotes around PLAYBACK_URL and clarified that it should be replaced with a URL string.</p>	October 24, 2022

---

<a href="#">Chat Client Messaging SDK: JavaScript Guide</a>	Added a new section, <a href="#">React Native Support</a> .	October 24, 2022
<a href="#">IVS Chat Client Messaging SDK: JavaScript 1.0.1</a>	Initial release of this new SDK. See <a href="#">Amazon IVS Chat Client Messaging SDK</a> in the <i>IVS User Guide</i> .  The <a href="#">Amazon IVS documentation landing page</a> points to the current version of the SDK References.  Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	October 18, 2022
<a href="#">Broadcast SDK: iOS 1.7.1</a>	Updated version number and artifact links for the new release, in the broadcast SDK guide: <a href="#">iOS</a> .  On the <a href="#">Amazon IVS documentation landing page</a> , updated the broadcast SDK Reference links to point to the new version.  Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	October 6, 2022
<a href="#">Web Player SDK 1.13.0 Release Notes</a>	Added a known issue to the Release Notes for Web Player 1.13.0, about the Sawmill Enabled log.	September 27, 2022

### [Broadcast SDK 1.7.0 release](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [Android](#), [iOS](#).

September 22, 2022

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Player 1.13.0 release](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

September 20, 2022

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Broadcast SDK: iOS 1.5.2](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [iOS](#).

September 12, 2022

The [Amazon IVS documentation landing page](#) points to the current version of Broadcast SDK References.

Also see the Amazon IVS [Release Notes](#) for this release.

### [IVS Chat Client Messaging SDK: Android 1.0.0 and iOS 1.0.0](#)

Initial release of these new SDKs. See [Amazon IVS Chat Client Messaging SDK](#) in the *IVS User Guide*.

September 8, 2022

The [Amazon IVS documentation landing page](#) points to the current version of the SDK References.

Also see the Amazon IVS [Release Notes](#) for this release.

We updated [Getting Started with Amazon IVS Chat](#) with links to various demos (including a backend server app that demonstrates token generation) and sample code for deleting a chat message.

### [Monitoring Amazon IVS with Amazon CloudWatch](#)

For some Amazon IVS metrics with the Channel dimension, we corrected the description. Channel values are not ARNs (as previously stated). They are the channel's resource-id, which is the last part of an ARN.

September 2, 2022

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

### [Resources and Support](#)

Added a new page to the Amazon IVS User Guide. This points to additional information about, and support for, Amazon IVS.

September 1, 2022



### [Merge fragmented streams](#)

Initial release of this new functionality. See these documentation changes:

August 30, 2022

- Getting Started with Amazon IVS – Updated console and CLI instructions in [Step 3: Create a Channel with Optional Recording](#).
- Auto-Record to S3 – Added [Merge Fragmented Streams](#)
- EventBridge – Added recording\_session\_id and recording\_session\_stream\_ids fields to [Examples: Recording State Change](#).

### [Monitoring Live Stream Health](#)

In [Filter Streams by Health](#), corrected the CLI example: changed filter-by name to filter-by health.

August 17, 2022

## [Expand BASIC channel](#)

The maximum resolution and bitrate for BASIC channels have changed. Resolution can be up to 1080p and bitrate can be up to 1.5 Mbps for 480p and up to 3.5 Mbps for resolutions between 480p and 1080p. See these documentation changes:

August 16, 2022

- *Getting Started with IVS* – Updated the screenshot in [Initial channel setup](#).
- *Streaming Configuration* – Updated definitions in [Channel Types](#).
- *Costs* – Updated channel definitions in [Live Video](#).
- *Service Quotas* – In [Other Quotas](#), updated IVS information for Ingest bitrate & Ingest resolution, for BASIC channels.

## [Player SDK 1.12.0 release: Web](#)

Updated version numbers and artifact links for the new release in Player guides: [Web](#), [Video.js Integration](#), and [JW Player Integration](#).

August 9, 2022

The [Amazon IVS documentation landing page](#) points to the current version of Player SDK References.

Also see the Amazon IVS [Release Notes](#) for this release.

---

<a href="#">Broadcast SDK: iOS 1.5.1</a>	Updated Release Notes for the July 28 release: added a fixed item (memory leak).	August 8, 2022
<a href="#">Auto-Record to Amazon S3</a>	In <a href="#">JSON Metadata Files</a> , added notes for recording <code>_started_at</code> and <code>recording_ended_at</code> , about using <code>duration_ms</code> to determine the duration of a recording.	August 8, 2022
<a href="#">Amazon IVS Broadcast SDK: Web</a>	Updated (here and in Release Notes) the July 21 entry for this release, by deleting the 1.0.0 version number and adding a note that documentation for future releases will be updated only on GitHub.	August 4, 2022
<a href="#">Clarify console instructions</a>	Noted that you click the hamburger icon to open the nav pane only if the pane is collapsed. This occurs in three places: <ul style="list-style-type: none"><li>• <a href="#">IVS Getting Started</a> – "Step 5: View Your Live Stream"</li><li>• <a href="#">Monitoring Live Streams</a> – "Access Stream Session Date" &amp; "Filter Streams by Health"</li></ul>	August 3, 2022

[Broadcast SDK release: iOS 1.5.1](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [iOS](#).

July 28, 2022

The [Amazon IVS documentation landing page](#) points to the current version of Broadcast SDK References.

Also see the Amazon IVS [Release Notes](#) for this release.

[Amazon IVS Broadcast SDK: Web](#)

Initial release of the Web broadcast SDK. See the documentation under "Amazon IVS Broadcast SDK" on the [Amazon IVS documentation landing page](#).

July 21, 2022

Also updated [Streaming with Amazon IVS Broadcast SDK](#) in *Getting Started with Amazon IVS*.

**Important:** For future releases, documentation will be updated only on GitHub: <https://aws.github.io/amazon-ivs-web-broadcast/> (not here).

---

<a href="#">IVS Chat metric</a>	Monitoring Amazon IVS with Amazon CloudWatch – A metric (Deliveries ) was added for IVS Chat.  Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to <a href="#">Monitoring IVS Low-Latency Streaming</a> .	July 15, 2022
<a href="#">Player SDK release: iOS 1.8.3</a>	Updated version number and artifact links for the new release, in the <a href="#">iOS Player Guide</a> .  The <a href="#">Amazon IVS documentation landing page</a> points to the current version of Player SDK References.	July 14, 2022
<a href="#">Estimate data use screenshot</a>	In <a href="#">Costs</a> , the screenshot for "Estimate data use" was updated: the "audio" rendition is no longer provided.	June 30, 2022

[Player SDK 1.11.0 release:](#)  
[Web](#)

June 28, 2022

Updated version number and artifact links for the new release, in Player guides: [Web](#), [Video.js Integration](#), and [JW Player Integration](#).

On the [Amazon IVS documentation landing page](#), updated the player SDK Web Reference link to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

In the [Player SDK: Web Guide](#), we deleted two items from "Known Issues and Workarounds" which no longer apply:

- When playing recorded content on an iOS mobile browser using the Video.js integration, the replay button does not work properly.
- When playing a live stream on a Google Pixel 4 or 4a mobile browser, playback may stop unexpectedly.

[Broadcast SDK 1.5.0 release](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [Android](#), [iOS](#).

June 22, 2022

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

[Streaming ingest configuration](#)

In [Stream Ingest: Codecs, RTMPS, and Port 443](#), clarified terminology: you specify an IVS *ingest server* (which includes port 443 in the path).

June 20, 2022

[Service Quotas](#)

For IVS Chat quotas, added quota for "rate of SendMessage requests per room" and clarified that the existing rate quota for SendMessage requests applies across all your rooms.

June 14, 2022

### [Ingest server format](#)

In [Getting Started with Amazon IVS](#), updated the screenshot in "Final Channel Creation" to show the current format of **Ingest server** (with port 443 and path /app/). Updated instructions in "Streaming with OBS Studio" and "Streaming a Recorded Video with FFmpeg."

June 14, 2022

### [Player SDK 1.10.0 release: Web and Android](#)

Updated version number and artifact links for the new release, in Player guides: [Web](#), [Android](#), [Video.js Integration](#), and [JW Player Integration](#).

May 24, 2022

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Service Quotas](#)

Added call-rate quotas for GetStreamSession and ListStreamSessions. (These IVS endpoints were added previously, when Stream Health was launched.)

May 16, 2022



<a href="#">iOS Player Guide</a>	<p>In "Known Issues and Workarounds," deleted a bullet about iOS 10, which is no longer supported:</p> <ul style="list-style-type: none"><li>• iOS 10 devices may experience a crash when returning from background.</li></ul> <p><b>Workaround:</b> Set the layer's <code>player</code> property to <code>nil</code> before entering the background.</p>	May 10, 2022
<a href="#">Broadcast SDK: Custom Image Sources</a>	Added a bullet for a new <code>CIFilter</code> implementation in the sample iOS app.	May 10, 2022
<a href="#">Web Player Guide</a>	<p>In "Content Security Policy," added domains for video streams from third-party CDNs (<code>*.akamaized.net</code> and <code>*.ext.cloudfront.live.hls.ttvnw.net</code> ).</p>	April 29, 2022
<a href="#">Video.js Player Guide</a>	<p>In "Events," deleted <code>MetadataEventType</code> (which is no longer available ) from the list of allowable event values.</p>	April 29, 2022
<a href="#">Security policy updates</a>	<p>In <a href="#">Identity-Based Policy Examples</a>, changed the console policy (added Chat, lambda, and Amazon CloudWatch) and the introductory text to it.</p>	April 29, 2022

[Private channels](#)

In [Generate and Sign Playback Tokens](#), specified that the exp timestamp value in the **payload** field of the token schema is UTC.

April 29, 2022

[OBS Studio setup](#)

IVS Getting Started – In [Streaming with OBS Studio](#), clarified how to specify the server and stream key, and added steps to set video resolution, bitrate, and keyframe interval.

April 29, 2022

## [Stream Health updates](#)

### [Monitoring Amazon IVS](#)

April 28, 2022

[Live Stream Health](#) – In "Console Instructions," noted that charts of the high-resolution CloudWatch metrics are available in the stream session details pages. In "Filter Streams by Health," added "CloudWatch Health Dimension for ConcurrentStreams."

Monitoring Amazon IVS with Amazon CloudWatch – A new dimension (Health) was added to the ConcurrentStreams metric, to filter the results by channel health.

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

## [Amazon IVS Chat](#)

April 26, 2022

Initial release of this new functionality. New and updated information is accessible from the [Amazon IVS documentation landing page](#):

- [Getting Started with Amazon IVS Chat](#) -- New page (in the *Amazon IVS Chat User Guide*).
- [Chat Message Review Handler](#) – New page (in the *Amazon IVS Chat User Guide*).
- Monitoring Amazon IVS with Amazon CloudWatch -- Added new metrics and a new namespace for chat.

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

Dec 28, 2023 update: Chat-related CloudWatch content was moved to [Monitoring Amazon IVS Chat](#).

- [Security](#) -- In "Data Protection," added chat bullets. In "Identity and Access Management," added a section on "Resource-Based Policy for Amazon IVS Chat." In

- "Infrastructure Security,"  
" added a section on  
"Amazon IVS Chat."
- [Service Quotas](#) -- In "Service Quota Increases ," updated which quotas are adjustable. Merged two sections into "Other Quotas." Added chat information in "API Call Rate Quotas," "Other Quotas," and "Service Quotas Integration with CloudWatch Usage Metrics."
  - On the [Amazon IVS documentation landing page](#), added an *Amazon IVS Chat* section with two API reference documents. See [IVS Chat API Documentation Changes](#) (a new section of this page).

Dec 28, 2023 update:

We moved chat-related information to the new IVS Chat User Guide. For other documentation changes see [Document History \(Chat\)](#).

<a href="#">iOS Player 1.8.2 release</a>	<p>Updated version number and artifact links for the new release, in the <a href="#">iOS Player Guide</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the iOS Player SDK Reference link to point to the new version.</p>	April 22, 2022
<a href="#">Manual SDK installation</a>	<p>In the "Getting Started" &gt; "Install the Library" section of <a href="#">Broadcast SDK: Android</a> and <a href="#">Player: Android Guide</a>, added a sentence about installing manually.</p>	April 19, 2022
<a href="#">Broadcast SDK 1.4.0 release</a>	<p>Updated version number and artifact links for the new release, in the broadcast SDK guide: <a href="#">Android</a>, <a href="#">iOS</a>.</p> <p>Added a new page on <a href="#">Broadcast SDK: Custom Image Sources</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the broadcast SDK Reference links to point to the new version.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	April 19, 2022

---

<a href="#">iOS Player 1.8.1 release</a>	Updated version number and artifact links for the new release, in the <a href="#">iOS Player Guide</a> .  On the <a href="#">Amazon IVS documentation landing page</a> , updated the iOS Player SDK Reference link to point to the new versions.	March 31, 2022
<a href="#">Device support for Android player</a>	In the <a href="#">Android Player Guide</a> , clarified which native Android devices are supported (phones and tablets). In the <a href="#">Player overview</a> , added a Supported Devices table column in the "Native Platform" section.	March 23, 2022
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Modified the Session Ended event and updated its description. Also clarified the event descriptions of Session Created and Stream End.	March 18, 2022
<a href="#">Player Video.js Integration</a>	In "Setup with Script Tag," step 1, added a closing <code>&lt;/script&gt;</code> to the example.	March 4, 2022

### [Broadcast SDK 1.3.0 release](#)

Updated version number and artifact links for the new release, in the broadcast SDK guide: [Android](#), [iOS](#).

March 3, 2022

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new version.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Player 1.8.0 release](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

March 1, 2022

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

### [Using Amazon EventBridge with Amazon IVS](#)

For the Recording End Failure event, added an example failure case: the attempt to write a master playlist fails.

February 10, 2022



---

<a href="#">Using Amazon EventBridge with Amazon IVS</a>	For the Recording Start event, added a note that it takes awhile before manifest files and video segments are written.	February 9, 2022
<a href="#">Broadcast SDK: Android 1.2.1 release</a>	Updated version number and artifact links for the new release, in the broadcast SDK guide: <a href="#">Android</a> .  On the <a href="#">Amazon IVS documentation landing page</a> , updated the broadcast SDK Reference link to point to the new version.  Also see the Amazon IVS <a href="#">Release Notes</a> for this release.	February 3, 2022
<a href="#">React Native wrapper for Player SDK</a>	In the <a href="#">Player Android Guide</a> and <a href="#">Player iOS Guide</a> , added a link to GitHub code and documentation for the new React Native wrapper.	January 27, 2022
<a href="#">React Native wrapper for Player SDK</a>	In the <a href="#">Player Android Guide</a> and <a href="#">Player iOS Guide</a> , added a link to GitHub code and documentation for the new React Native wrapper.	January 27, 2022
<a href="#">Web Player CSP change</a>	In "Hosting Assets on a Separate Origin," add information for Chrome.	January 25, 2022

---

<a href="#">Setting Up Private Channels</a>	In "Token Schema," added information about support for multiple domains and wildcard domains in the <code>access-control-allow-origin</code> <code>token-payload</code> field.	January 24, 2022
<a href="#">Web Player 1.7.0 Release Notes</a>	In the <a href="#">Release Notes</a> , updated the bullet on <code>setInitialBufferDuration()</code> to say it does <b>not</b> work on iOS mobile browsers.	January 21, 2022
<a href="#">Player 1.7.0 release</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, <a href="#">Video.js Integration</a>, and <a href="#">JW Player Integration</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	January 20, 2022

---

<a href="#">R2S3 thumbnail configuration release</a>	<p>In <i>Getting Started with Amazon IVS</i>, we updated <a href="#">Step 3: Create a Channel with Optional Recording</a>.</p> <p>In <a href="#">Auto-Record to Amazon S3</a>, we added a note to "Recording Contents" about modifying the <code>thumbnails</code> folder, added a new "Thumbnails" section, and changed the information about the <code>thumbnails</code> and <code>path</code> fields in "JSON Metadata Files."</p>	January 18, 2022
<a href="#">Android Player Guide</a>	In "Install the Library," deleted the <code>jcenter()</code> line, as JCenter is deprecated.	January 7, 2022
<a href="#">iOS Player</a>	Added a "Known Issue" about the player crashing when testing against the arm64e architecture.	December 20, 2021
<a href="#">Broadcast SDK 1.2.0 release</a>	<p>Updated version number and artifact links for the new release, in all broadcast SDK guides: <a href="#">Android</a>, <a href="#">iOS</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the broadcast SDK Reference links to point to the new versions.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	December 9, 2021

---

<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Expanded the descriptions of stream/session start/create/end events and added usage suggestions.	December 3, 2021
<a href="#">Streaming Configuration</a>	For streaming from Android and iOS, replaced the information on Larix Broadcaster with a pointer to documentation on the Amazon IVS broadcast SDK.	November 24, 2021
<a href="#">Broadcasting: SDK for Android Guide</a>	Added an issue for Android 5/6/7 devices, which can use only the system's default microphone, hence cannot receive the broadcast SDK's onDeviceAdded and onDeviceRemoved callbacks for microphones.	November 24, 2021
<a href="#">Player 1.6 release</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, <a href="#">Video.js Integration</a>, and <a href="#">JW Player Integration</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	November 23, 2021

[Amazon IVS Player](#)

At the end of the introductory text, added a paragraph about casting support and a pointer to Amazon IVS Broadcast SDK documentation.

November 23, 2021

## [Monitoring Amazon IVS Live Stream Health](#)

New User Guide page for new Amazon IVS functionality. For Stream Health, we also:

November 18, 2021

- Updated the IAM policy in "Step 2: Set up IAM Permissions" of [Getting Started with Amazon IVS](#): added three IVS permissions (GetStream , GetStreamSession , ListStreamSessions ) and cloudwatch:GetMetricData .
- Added four high-resolution metrics to Monitoring Amazon IVS with Amazon CloudWatch: IngestAudioBitrate , IngestFrameRate , IngestVideoBitrate , and KeyframeInterval .

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

- Added two events to [Using Amazon EventBridge with Amazon IVS](#): Session Created and Session Ended.

## [Using Amazon EventBridge with Amazon IVS](#)

Updated the description of the Recording Start event.

November 5, 2021

[Broadcasting: SDK for iOS Guide](#)

Add a "Known Issue" for AirPods connected to an iOS 12 device.

November 4, 2021

[Stream with FFmpeg](#)

In *Streaming Configuration*, clarified that FFmpeg can be used with many OSs/devices (not just Windows Desktop) and fixed the format of the example in the Webcam bullet.

November 3, 2021

## [Broadcast SDK \(Android and iOS\) 1.1.0 release](#)

Updated version number and artifact links for the new release, in broadcast SDK guides: [Android](#) and [iOS](#). In Android, there are new `setPosition` coordinates in "Create a Broadcast Configuration." In iOS, there is a new advanced use case ("Use Background Video"), slot-position changes in "Create a Broadcast Configuration," and a new "Known Issue."

October 20, 2021

On the [Amazon IVS documentation landing page](#), updated the broadcast SDK Reference links to point to the new versions.

Added a new page, [Broadcasting: Mixer Guide](#), to the documentation for this feature.

Also see the Amazon IVS [Release Notes](#) for this release.

## [Setting Up Private Channels](#)

In "Token Schema," updated `access-control-allow-origin` definition to refer to "origin" instead of "domain."

October 11, 2021



<a href="#">Android Player 1.5.1 release</a>	Bug-fix release; see Amazon IVS <a href="#">Release Notes</a> . Also updated version-number references in links and text in the <a href="#">Android Player Guide</a> .	September 29, 2021
<a href="#">Player 1.5.0 release</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, <a href="#">Video.js Integration</a>, and <a href="#">JW Player Integration</a>.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>Also see the Amazon IVS <a href="#">Release Notes</a> for this release.</p>	September 28, 2021
<a href="#">Streaming Configuration</a>	In "Audio Settings," specified a minimum bitrate, 96 Kbps.	September 22, 2021
<a href="#">Getting Started with Amazon IVS</a>	In "Step 4: Set Up Streaming Software," added a note about disconnecting if no data is sent for 30 seconds.	September 20, 2021
<a href="#">Identity-based policy example</a>	In Amazon IVS Security, fixed a typo in the example in <a href="#">Access an Amazon IVS Channel</a> : added ending punctuation (} ]}).	September 17, 2021

[SDK sizes for Player 1.4.1 and 1.4.0](#)

In the Release Notes for Player [1.4.1](#) and [1.4.0](#), we made corrections to the tables of mobile SDK sizes.

September 16, 2021

[Player 1.4.1 release](#)

Bug-fix release; see [Amazon IVS Release Notes](#). Also updated version number and artifact links in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

September 8, 2021

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

In Streaming Configuration, updated the information on [Closed Captioning](#).

[Broadcasting: SDK for Android Guide](#)

In "Set the ImagePreviewView for Preview," made minor text clarifications. In "Swap Cameras," fixed two typos. In "Create a Broadcast Configuration," deleted the line referencing `video.setDefaultAspectMode` , which is not usable now.

September 1, 2021

<a href="#">Streaming configuration with FFmpeg</a>	Modified settings for capturing video files. Specifically, <code>-g 120</code> was changed to <code>-force_key_frames expr:gte(t,n_force_d*2)</code> . This causes the encoder to insert a keyframe every 2 seconds, regardless of source-input frame rate.	August 23, 2021
<a href="#">Amazon IVS Player: SDK for Web Guide</a>	Added new "Known Issue" for Pixel 4/4a mobile browsers.	August 20, 2021
<a href="#">Amazon IVS Player: Video.js Integration</a>	In "Sample Code," updated the version number to 7.14.3. There is a security vulnerability in versions of Video.js earlier than 7.14.3.	August 19, 2021
<a href="#">Streaming Configuration</a>	For the STANDARD channel type, added a note that audio is transcoded only for renditions 360p and below; above that, audio is passed through.	August 18, 2021
<a href="#">Getting Started with Amazon IVS</a>	In "Step 2: Set Up IAM Permissions," added steps to attach the policy to an existing user. This new procedure is in addition to the old procedure, which is for creating a new user and attaching a policy to that user.	August 11, 2021

## [Player 1.4.0 release](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#).

August 10, 2021

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

Also see the Amazon IVS [Release Notes](#) for this release.

## [Amazon IVS Player: SDK for Web Guide](#)

In "Setup With NPM," added a note about hosting player static assets from your own domain.

July 30, 2021

## [Getting Started with Amazon IVS](#)

In "Step 2: Set Up IAM Permissions," updated policy information and instructions.

July 29, 2021

In "Step 3: Create a Channel with Optional Recording," added a section, "Auto-Record to S3" (to replace a prior paragraph).

In "Step 4: Set up Streaming Software," added a section, "Streaming with the Amazon IVS Broadcast SDK."

---

<a href="#">Auto-Record to S3</a>	Added a new section, <a href="#">“Playback of Recorded Content from Private Buckets.”</a> Also updated the introduction to this page.	July 28, 2021
<a href="#">Amazon IVS Broadcast SDK (Android and iOS)</a>	Initial release of the broadcast SDK for Android and iOS. See the documentation under "Amazon IVS Broadcast SDK," a new section of the Amazon IVS <a href="#">documentation landing page</a> .	July 27, 2021
<a href="#">Amazon IVS Player</a>	Updated <a href="#">Desktop Browsers</a> to indicate Amazon IVS Player 1.3.0 support for ultra-low latency on new versions of Safari for macOS.	July 14, 2021
<a href="#">Amazon IVS Service Quotas</a>	For the PutMetadata endpoint, added a limit of 155 TPS per account.	June 29, 2021
<a href="#">ivs.rocks</a>	On the Amazon IVS User Guide <a href="#">landing page</a> , added a link to and brief description of ivs.rocks.	June 25, 2021
<a href="#">Player Browser &amp; Platform Requirements</a>	For the Amazon IVS Player, added links to sites listing the latest versions of supported browsers.	June 25, 2021

---

<a href="#">Streaming Configuration</a>	In "Channel Types," updated definitions of channel types. For STANDARD channels, resolution can be up to 1080p; for BASIC channels, 480p. (The prior definitions were only in terms of vertical resolution.)	June 17, 2021
<a href="#">Costs</a>	Added a new page on costs.	June 17, 2021
<a href="#">Amazon IVS Player: SDK for Android Guide</a>	Added a new "Permissions" section.	June 17, 2021
<a href="#">Player mobile-browser support</a>	In <a href="#">Mobile Browsers</a> , added information about support for Chrome for iPadOS and Safari for iPadOS.	June 14, 2021
<a href="#">Player SDK size</a>	Added a new "SDK Size" section to the <a href="#">Android</a> and <a href="#">iOS</a> Player SDK guides.	June 11, 2021
<a href="#">Amazon IVS Player: SDK for Web Guide</a>	Added two "Known Issues" when playing content on an iOS mobile browser (with <code>player.getQualities()</code> and <code>player.getLiveLatency()</code> calls).	June 9, 2021

## [Supported regions and service endpoints](#)

Replace lists of supported regions with a link to the [Amazon IVS page in the AWS General Reference](#), which is updated automatically when support for new regions is added. Changes were made on the Monitoring Amazon IVS with Amazon CloudWatch page.

June 8, 2021

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

## [Amazon IVS Player issues](#)

In "Known Issues and Workarounds," for the [Web](#), [Android](#), and [iOS](#) Player, asked customers to report all issues to Support. Also added an issue with Android 11 emulators.

June 4, 2021

## [Android and iOS Player 1.3.3 release](#)

Bug-fix release; see [Amazon IVS Release Notes](#). Also updated version-number references in links and text in the [Android Player Guide](#) and [iOS Player Guide](#).

June 1, 2021

The [Amazon IVS documentation landing page](#) always points to the most current versions of the Player SDK References.

---

<a href="#">Setting Up Private Channels</a>	Updated "Generate and Sign Playback Tokens" (information on creating the signature and steps in "Instructions").	May 26, 2021
<a href="#">Global versus regional</a>	Moved "Global Solution, Regional Control" from Getting Started with Amazon IVS to <a href="#">What Is Amazon IVS</a> .	May 21, 2021
<a href="#">Amazon IVS Player: Video.js Integration</a>	In "Sample Code," updated the Cloudflare version number from 7.6.6 to 7.11.4.	May 20, 2021
<a href="#">Android Player 1.3.2 release</a>	Bug-fix release; see <a href="#">Amazon IVS Release Notes</a> . Also updated version-number references in links and text in the <a href="#">Android Player Guide</a> .	May 19, 2021
<a href="#">Amazon IVS Service Quotas</a>	Minor wording changes. Deleted information about maximum number of tags; this was moved to the API Reference.	May 12, 2021
<a href="#">Amazon IVS Release Notes</a>	Added a note for Web Player 1.3.1: the 1.3.0 NPM package exists but does not work.	May 11, 2021
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Updated <code>stream_id</code> to be a "sanitized" value in all relevant examples.	May 10, 2021



[Amazon IVS Player: SDK for Web Guide](#)

Added a known issue and workaround, for `player.seekTo()` calls when playing recorded content on an iOS mobile browser.

May 10, 2021

[Streaming Configuration](#)

Renamed the Encoder Configuration page to Streaming Configuration.

May 6, 2021

[Using Amazon EventBridge with Amazon IVS](#)

In "Examples: Recording State Change," added the `recording_duration_ms` field, changed the example value of the `recording_s3_key_prefix` field, and changed the value of the `recording_status_reason` field.

May 5, 2021

## [Player 1.3 release](#)

Updated version number and artifact links for the new release, in all player guides: [Web](#), [Android](#), [iOS](#), [Video.js Integration](#), and [JW Player Integration](#). For Android, added `mavenCentral()` to "Install the Library."

May 5, 2021

On the [Amazon IVS documentation landing page](#), updated the player SDK Reference links to point to the new versions.

In Player 1.3.0 and later, timed metadata is now supported on Chrome and Safari for iOS. This is noted in the [IVS Player SDK overview](#) (table on "Mobile Browsers" ) and [Embedding Metadata within a Video Stream](#) (in "Consuming Metadata").

Also see the Amazon IVS [Release Notes](#) for this release.

## [Amazon IVS Service Quotas](#)

Added a new section, "Service Quotas Integration with CloudWatch Usage Metrics."

April 26, 2021

[Maximum duration of a stream](#)

In [Getting Started with Amazon IVS](#) ("Step 4: Set Up Streaming Software"), added a note about the maximum duration of a stream, 48 hours.

April 23, 2021

[IAM policy changes](#)

Made several IAM policy changes:

April 22, 2021

- [Getting Started with Amazon IVS](#) – In "Step 2: Set Up IAM Permissions," added service quotas.
- [Amazon IVS Security](#) – In "Use the Amazon IVS Console," simplified the policy example and added service quotas.

## [New CloudWatch metrics](#)

Various doc changes for the release of new CloudWatch metrics:

April 13, 2021

- [Monitoring Amazon IVS with Amazon CloudWatch](#) — Added new metrics: concurrent views and concurrent streams.

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

- [Service Quotas](#) – Updated the names of related quotas to match the new metrics.
- [Glossary](#) – Added "view."

## [Auto-Record to S3](#)

New User Guide page for this new Amazon IVS functionality. This also affects several existing documents:

April 7, 2021

- [Getting Started with Amazon IVS](#) — Added IAM policy info for R2S3. Rewrote the step to create a channel. Added a paragraph on optionally enabling local recording in OBS Studio. New section on disabling recording.
- [Using Amazon EventBridge with Amazon IVS](#) — Added Recording State Change events.
- [Monitoring Amazon IVS with Amazon CloudWatch](#) — Added RecordedTime metric.

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

- [Amazon IVS Security](#) — Added a section on "Using Service-Linked Roles (SLRs) for Amazon IVS."
- [Service Quotas](#) — Added "API Call Rate Quotas" for the new recording-configuration endpoints and a

"Resource Quotas" limit for recording configurations.

[Amazon IVS Streaming Configuration](#)

In "Closed Captioning," clarified that the Player SDKs support only 1 language, not multi-track captions playback.

March 29, 2021

[Global versus regional](#)

In [What is Amazon IVS](#), added a new section, "Global Solution, Regional Control," to clarify what is global versus regional. In [Getting Started with Amazon IVS](#), mentioned selecting a region, in the instructions for creating a channel.

March 25, 2021

[EventBridge event latency & IDR/Keyframe encoder setting](#)

Clarified the relationship between the IDR/Keyframe video-encoder setting and latency in some EventBridge events. This affects two documents:

March 25, 2021

- ["Amazon IVS Streaming Configuration"](#) – See the IDR/Keyframe bullet in "Reducing Latency."
- ["Using Amazon EventBridge with Amazon IVS"](#) – See the new "Note on latency of Stream State Change events."

<a href="#">Monitoring Amazon IVS with Amazon CloudWatch</a>	Clarified how long CloudWatch retains data.	March 18, 2021
	Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to <a href="#">Monitoring IVS Low-Latency Streaming</a> .	
<a href="#">Streaming Configuration</a>	In "Audio Settings," changed the supported bitrate to 320 Kpbs (from 192).	March 15, 2021
<a href="#">Required versions of TLS</a>	Clarified requirements for TLS (Transport Layer Security) . For API calls, clients must support TLS 1.0 or later, but we recommend TLS 1.2 or later. For streaming/playback , TLS version 1.2 or later is required.  Changes were made in two documents: Streaming Configuration (section on " <a href="#">Stream Ingest: Codecs, RTMPS, and Port 443</a> ") and Security (section on " <a href="#">Infrastructure Security</a> ").	March 15, 2021
<a href="#">Amazon IVS Player: SDK for Web Guide</a>	Added a known issue with HTML5 and <code>setQuality()</code> .	March 15, 2021
<a href="#">Amazon IVS Player: SDK for Web Guide</a>	Added a known issue with captions.	March 11, 2021

<a href="#">Amazon IVS Player</a>	<p>Added sections on "Thread Safety" in <a href="#">SDK for Android Guide</a> and <a href="#">SDK for IOS Guide</a>.</p> <p>Also, for Android, noted that after the <code>player.release()</code> method is called, the player can no longer be used.</p>	March 2, 2021
<a href="#">Monitoring Amazon IVS with Amazon CloudWatch</a>	<p>Updated the procedure for accessing Amazon IVS metrics using the CloudWatch console: added information on when "IVS" is listed and a screenshot.</p> <p>Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to <a href="#">Monitoring IVS Low-Latency Streaming</a>.</p>	February 26, 2021
<a href="#">Security</a>	<p>In "Infrastructure Security," added a note that Amazon IVS streaming requires TLS 1.2. Also listed a new web page for details on AWS global network security procedures.</p>	February 17, 2021
<a href="#">Amazon IVS Player: JW Player Integration</a>	<p>New User Guide page on the JW Player plug-in for the Amazon IVS player. Also added a JW Player row to the <a href="#">Framework Integrations table</a> in the <i>Web Player Guide</i>.</p>	January 28, 2021



<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Expanded the wording about guarantees for sending events.	January 22, 2021
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Added: Events are sent on a best-effort basis.	January 13, 2021
<a href="#">Streaming Configuration</a>	Changed the codec audio setting from AAC to AAC (LC).	December 18, 2020
<a href="#">Amazon IVS Service Quotas</a>	In "Resource Quotas," added the maximum number of tags for a resource.	December 17, 2020
<a href="#">Android Player 1.2.1 release</a>	Bug-fix release; see <a href="#">Amazon IVS Release Notes</a> . Also updated version-number references in links and text in the <a href="#">Android Player Guide</a> .	December 16, 2020
<a href="#">Amazon IVS Release Notes</a>	For Amazon IVS Android Player 1.2.0 and 1.1.0, added a known issue which causes the SDK to crash.	December 11, 2020
<a href="#">Getting Started with Amazon IVS</a>	In bullet on playback URLs (in "Step 3: Create a Channel") , added a note that custom domains for playback are not supported.	December 4, 2020
<a href="#">Amazon IVS Release Notes</a>	Deleted download links for iOS Player 1.0.6 and 1.0.0; these versions are deprecated.  Added a "Known Issue" for iOS Player 1.2.0.	December 4, 2020

<a href="#">Player 1.2.0 release</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, and <a href="#">Video.js Integration</a>.</p> <p>Added a Known Issue to the Android guide.</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>Also see the updated Amazon IVS <a href="#">Release Notes</a>.</p>	November 23, 2020
<a href="#">Setting Up Private Channels</a>	<p>In the section on "Generate and Sign Playback Tokens," the <code>channel-arn</code> value in the JWT payload is a string.</p>	November 18, 2020
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	<p>Added <code>stream_id</code> field to many events. This is a unique stream identifier assigned each time a channel goes live. For a given channel, each live stream has a new <code>stream_id</code>. Stream IDs allow customers to distinguish different stream sessions on the same channel.</p>	November 12, 2020
<a href="#">Embedding Metadata Within a Video Stream</a>	<p>Added new section on "Viewing Timed Metadata" from the Amazon IVS console.</p>	November 9, 2020

<a href="#">Web Player Guide</a>	Updated the section on "Content Security Policy," especially for hosting assets on a separate page when using Safari.	November 4, 2020
<a href="#">Service Quotas (CCV and CCB limits)</a>	Added notes about the importance of ensuring adequate concurrent-viewer and concurrent-broadcaster limits, especially before large streaming events. See <a href="#">Getting Started with Amazon IVS</a> and <a href="#">Amazon IVS Service Quotas</a> .	November 4, 2020
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Updated Limit Breach events: the detail section of the JSON blob uses <code>limit_name</code> for all these events. (Previously only Concurrent Broadcasts showed that and the others showed <code>limit</code> .)	October 28, 2020
<a href="#">Setting Up Private Channels</a>	In the section on "Generate and Sign Playback Tokens," noted that the <code>exp</code> (expiration) field in JWT payloads is an integer.	October 27, 2020
<a href="#">Amazon IVS Service Quotas</a>	Increased three limits: number of channels, concurrent viewers, and concurrent broadcasts.	October 27, 2020

<a href="#">Web Player 1.1.2 release</a>	Bug-fix release; see the <a href="#">Amazon IVS Release Notes</a> . Version-number references in links and text were updated in the <a href="#">Web Player Guide</a> and <a href="#">Video.js integration Guide</a> .	October 9, 2020
<a href="#">Ingest resolution quotas &amp; event</a>	Added service quotas and EventBridge events for ingest resolution. See <a href="#">Amazon IVS Service Quotas</a> and <a href="#">Using Amazon EventBridge with Amazon IVS</a> .	October 9, 2020
<a href="#">Player 1.1.0 release</a>	<p>Updated version number and artifact links for the new release, in all player guides: <a href="#">Web</a>, <a href="#">Android</a>, <a href="#">iOS</a>, and <a href="#">Video.js Integration</a>.</p> <p>In the iOS and Web guides, added a new section on "Known Issues."</p> <p>On the <a href="#">Amazon IVS documentation landing page</a>, updated the player SDK Reference links to point to the new versions.</p> <p>In the <a href="#">Amazon IVS Player overview</a>, deleted the Android getSessionId function (which does not yet work).</p>	October 7, 2020

<a href="#">Setting Up Private Channels</a>	Added a new section, "Workflow for Private Channels." In the section on generating and signing tokens, clarified payload field descriptions and example. Corrected examples for listing and getting playback key pairs.	September 21, 2020
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	The <code>channel_name</code> field was added to several events.	September 14, 2020
<a href="#">Embedding Metadata Within a Video Stream</a>	Expanded information on setting up IAM permissions (full procedure and policy), inserting metadata (added a CLI procedure), and consuming metadata (linked to several GitHub demos).	September 14, 2020
<a href="#">Player guides</a>	Clarified which is the most current version of each player ( <a href="#">Web</a> , <a href="#">Android</a> , <a href="#">iOS</a> , and <a href="#">Video.js Integration</a> ).	September 9, 2020
<a href="#">Getting Started with Amazon IVS</a>	Mentioned that there is a short delay before a new stream can be viewed in the console.	September 9, 2020
<a href="#">Amazon IVS Release Notes</a>	Changed the Player iOS download link to be the same as what is in the Player iOS Guide.	September 9, 2020
<a href="#">Embedding Metadata within a Video Stream</a>	Added link to relevant AWS blog posts.	September 3, 2020

<a href="#">Amazon IVS Player</a>	Expanded the discussion of player features. Clarified that we can guarantee the performance of only the Amazon IVS player (not third-party players).	September 3, 2020
<a href="#">Amazon IVS Service Quotas</a>	Corrected this to indicate that only the channels, concurrent viewers, and concurrent broadcasts quotas can be adjusted.	August 31, 2020
<a href="#">Streaming Configuration</a>	Several changes, including adding <a href="#">Reducing Latency</a> subsection on “Avoid Third-Party Streaming/Forwarding Services” and clarifying why we strongly recommend <a href="#">CBR</a> over VBR.	August 24, 2020
<a href="#">Embedding Metadata within a Video Stream</a>	Updated Web example in <a href="#">Consuming Timed Metadata</a> .	August 24, 2020
<a href="#">Amazon IVS Player: SDK for Android Guide</a>	Updated code example in <a href="#">Install the Library</a> .	August 24, 2020
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	In the section on “Examples : Limit Breach,” updated several field names: <code>limit_name</code> , <code>limit_value</code> , <code>exceeded_by</code> , and <code>limit_unit</code> . These names include underscores (not dashes).	August 19, 2020

<a href="#">Setting Up Private Channels</a>	<p>New User Guide page on new Amazon IVS functionality, supporting private channels. This also affects several existing documents:</p> <p><a href="#">Getting Started with Amazon IVS</a> and <a href="#">Logging Amazon IVS API Calls with AWS CloudTrail</a>: Added authorized field to channel.</p> <p><a href="#">Security</a>: Several changes including a new section on “Privileged and Unprivileged Access.”</p> <p><a href="#">Service Quotas</a>: Added several playback quotas.</p> <p><a href="#">Glossary</a>: Added playback key pair.</p>	August 19, 2020
<a href="#">Getting Started with Amazon IVS</a>	Added a new section on <a href="#">AWS Regional Service</a> .	August 11, 2020
<a href="#">Amazon IVS Player: SDK for iOS Guide</a>	Updated links to the reference documentation and framework download to point to the 1.0.6 release. Also updated reference-doc link on the Amazon IVS <a href="#">doc landing page</a> .	August 11, 2020

---

<a href="#">Using Amazon EventBridge with Amazon IVS</a>	Amazon IVS EventBridge events are now available through the Amazon EventBridge console. See the section on "Creating Amazon EventBridge Rules for Amazon IVS."	August 5, 2020
<a href="#">Amazon IVS Player: Video.js Integration</a>	In the "Setup With NPM" section, updated the link to the Video.js npm package to install, to version 7.6.6.	July 30, 2020
<a href="#">Using Amazon EventBridge with Amazon IVS</a>	For Amazon IVS stream-state and stream-health changes, the event name is provided in a field called <code>event_name</code> (not <code>eventName</code> , as previously documented).	July 29, 2020
<a href="#">Getting Started with Amazon IVS</a>	Changed the instructions for setting up streaming software, to indicate that port 443 is required for Amazon IVS ingest. This also affects the <i>Streaming Configuration</i> document; see the new section on <a href="#">RTMPS and Port 443</a> .	July 27, 2020
<a href="#">Amazon IVS Player: SDK for iOS Guide</a>	Changed the download location of the latest version, in the instructions for installing the framework manually.	July 27, 2020



[Embedding Metadata Within a Video Stream](#)

Added Android and iOS examples of consuming timed metadata. July 24, 2020

[New service and User Guide](#)

This is the initial release of Amazon Interactive Video Service (IVS). July 15, 2020

## IVS Low-Latency Streaming API Reference Changes

API Change	Description	Date
Remove <code>svs</code> from ARN patterns	ARN patterns which specified <code>[is]vs</code> were updated to specify <code>ivs</code> . This affects several endpoints (including all Tags endpoints) and the following objects: <code>BatchError</code> , <code>BatchStartViewerSessionRevocationError</code> , <code>BatchStartViewerSessionRevocationViewerSession</code> , <code>Channel</code> , <code>ChannelSummary</code> , <code>PlaybackKeyPair</code> , <code>PlaybackKeyPairSummary</code> , <code>Stream</code> , <code>StreamSummary</code> , <code>StreamKey</code> , <code>StreamKeySummary</code> .	April 25, 2024
Secure Reliable Transport ingest support	We added the <code>srt</code> field to the <code>Channel</code> object. This affects six responses: <code>BatchGetChannel</code> , <code>CreateChannel</code> , <code>GetChannel</code> , <code>UpdateChannel</code> , and <code>GetStreamSession</code> .	April 4, 2024
Tokenless playback restrictions	<ul style="list-style-type: none"> <li>Added a new resource, <code>PlaybackRestrictionPolicy</code>.</li> <li>Added five <code>PlaybackRestrictionPolicy(ies)</code> endpoints (<code>Create/Delete/Get/List/Update</code>).</li> <li>Added the <code>PlaybackRestrictionPolicy</code> and <code>PlaybackRestrictionPolicySummary</code> objects.</li> </ul>	January 31, 2024

API Change	Description	Date
	<ul style="list-style-type: none"> <li>Added <code>playbackRestrictionPolicyArn</code> to the <code>Channel</code> and <code>ChannelSummary</code> objects. This affects Channel endpoint responses (<code>Create/BatchGet/Get/List/Update</code>).</li> <li>In the <code>ListChannel</code> request, added <code>filterByPlaybackRestrictionPolicyArn</code>.</li> </ul>	
Channel-type definitions	Updated channel-type definitions to provide more detail, especially about rendered transcode ladders. See <a href="#">Channel Types</a> in the <i>IVS Low-Latency Streaming API Reference</i> .	August 18, 2023
R2S3 rendition filtering and thumbnail enhancements	<ul style="list-style-type: none"> <li>In <code>ThumbnailConfiguration</code>, added <code>resolution</code> and <code>storage</code>. This affects the <code>CreateRecordingConfiguration</code> request and response, <code>GetRecordingConfiguration</code> response, and <code>GetStreamSession</code> response.</li> <li>In <code>ThumbnailConfiguration</code>, changed the <code>targetIntervalSeconds</code> minimum from 5 to 1 and updated the "Important" note to say it applies only to BASIC channels.</li> <li>Added the <code>RenditionConfiguration</code> object.</li> <li>Added <code>renditionConfiguration</code> to the <code>RecordingConfiguration</code> object. This affects three responses: <code>CreateRecordingConfiguration</code>, <code>GetRecordingConfiguration</code>, and <code>GetStreamSession</code>. We also added <code>renditionConfiguration</code> to the <code>CreateRecordingConfiguration</code> request.</li> </ul>	July 17, 2023

API Change	Description	Date
Viewer session revocation for private channels	<ul style="list-style-type: none"> <li>Added two endpoints: <code>StartViewerSessionRevocation</code> and <code>BatchStartViewerSessionRevocation</code>.</li> <li>Added two objects: <code>BatchStartViewerSessionRevocationError</code> and <code>BatchStartViewerSessionRevocationViewerSession</code>.</li> </ul>	June 28, 2023
Advanced channel types	<ul style="list-style-type: none"> <li>Added new channel type values and definitions. This affects two requests (<code>Create/UpdateChannel</code>) and the <code>Channel</code> object.</li> <li>Added the <code>preset</code> field to the <code>Channel</code> and <code>ChannelSummary</code> objects. This affects several requests (<code>Create/UpdateChannel</code>) and responses (<code>BatchGetChannel</code>, <code>Create/Get/UpdateChannel</code>, <code>GetStreamSession</code>, <code>ListChannels</code>).</li> <li>Added <code>type</code> to the <code>ChannelSummary</code> object. This affects the <code>ListChannels</code> response.</li> </ul>	June 2, 2023
RTMP support	Added the <code>insecureIngest</code> field to the <code>Channel</code> and <code>ChannelSummary</code> objects. This affects several requests and responses.	March 30, 2023
Stream state	In the <code>Stream</code> and <code>StreamSummary</code> objects, noted that the <code>OFFLINE</code> value of the <code>state</code> field should not be relied on. Instead, a "NotBroadcasting" error will indicate that the stream is not live.	February 8, 2023

API Change	Description	Date
Merge fragmented streams	Added the <code>recordingReconnectWindowSeconds</code> field to the <code>CreateRecordingConfiguration</code> request and the <code>RecordingConfiguration</code> object. This affects three responses ( <code>CreateRecordingConfiguration</code> , <code>GetRecordingConfiguration</code> , and <code>GetStreamSession</code> ).	August 30, 2022
Expand BASIC channel to 1080p	Updated channel type definitions in <code>CreateChannel</code> , <code>UpdateChannel</code> , and the <code>Channel</code> object.	August 16, 2022
Restrictions on tags	Updated information on tags restrictions. Amazon IVS has no constraints on tags beyond what is documented in the AWS documentation that we link to. This affects the "Welcome" section and several endpoints and data types.	August 12, 2022
Max and default values of <code>maxResults</code>	Updated the maximum and default values of <code>maxResults</code> to reflect the actual behavior of the system. Affects all List endpoints that use <code>maxResults</code> .	August 12, 2022
Timestamp fields	For ISO 8601 fields, added a note that these are returned as strings. Due to an auto-generation issue, these appear in our documented syntax as <code>number</code> .	March 28, 2022
API authorization	In "Authentication versus Authorization," clarify the bullet on authorization.	March 18, 2022
ARN encoding in tag endpoints	For the three tag endpoints, added a statement that the <code>resourceArn</code> field must be URL-encoded.	March 18, 2022

API Change	Description	Date
Audio/video configuration objects	Updated the definitions of the AudioConfiguration and VideoConfiguration data types, to indicate that they are used for monitoring. (Configuration is done in the broadcaster's encoder.)	February 17, 2022
R2S3 thumbnail configuration release	<p>Added a new field (<code>thumbnailConfiguration</code>) in the RecordingConfiguration object. This in turn affects the CreateRecordingConfiguration request and response, GetRecordingConfiguration response, and GetStreamSession response.</p> <p>Added a new object: ThumbnailConfiguration.</p>	January 18, 2022
Stream Health release	<p>Added 2 endpoints: GetStreamSession and ListStreamSessions.</p> <p>Added 7 objects: AudioConfiguration, IngestConfiguration, StreamEvent, StreamFilters, StreamSession, StreamSessionSummary, and VideoConfiguration.</p> <p>Added the <code>streamID</code> field to the Stream and StreamSummary objects. This in turn affects the GetStream and ListStreams responses.</p> <p>Added the <code>filtersBy</code> field to the ListStreams request.</p>	November 18, 2021
Format of time fields	Updated the description of <code>startTime</code> in the Stream and StreamSummary objects, to add that it's an ISO 8601 timestamp returned as a string.	September 21, 2021

API Change	Description	Date
STANDARD channel type	For the STANDARD channel type, added notes that audio is transcoded only for renditions 360p and below; above that, audio is passed through.	August 18, 2021
ListTagsForResource endpoint	Removed support for pagination; i.e., the <code>maxResults</code> request field and <code>nextToken</code> request/response field. (Pagination did not work correctly.)	August 13, 2021
PutMetadata TPS limit per account	For the PutMetadata endpoint, added a limit of 155 TPS per account.	June 29, 2021
Channel-type definitions	Updated the definitions of channel types. For STANDARD channels, resolution can be up to 1080p; for BASIC channels, 480p. (The prior definitions were only in terms of vertical resolution.)	June 17, 2021
Supported regions and service endpoints	Replace lists of supported regions with a link to the <a href="#">Amazon IVS page in the AWS General Reference</a> , which is updated automatically when support for new regions is added. Changes were made on the "Welcome" page.	June 8, 2021
Tagging	In "Tagging" (in the "Welcome" section), added the maximum number of tags that can be applied to a resource (50).	May 12, 2021
New CloudWatch Metrics	Changed the definition of <code>viewerCount</code> in the Stream and StreamSummary objects.	April 13, 2021

API Change	Description	Date
Auto-Record to S3	<ul style="list-style-type: none"> <li>Added 4 recording-configuration endpoints (Create, Delete, Get, List).</li> <li>Add 4 data types (DestinationConfiguration, RecordingConfiguration, RecordingConfigurationSummary, S3DestinationConfiguration).</li> <li>Added a RecordingConfigurationArn field to the Channel and ChannelSummary objects and channel endpoints.</li> <li>Modified ListChannels to filter by recording-configuration ARN.</li> </ul>	April 7, 2021
Authentication & authorization	<ul style="list-style-type: none"> <li>Added an "<a href="#">Authentication versus Authorization</a>" section to clarify the difference between these concepts.</li> <li>Changed the description of the authorized field (in the Channel data type and channel endpoints), to: "Whether the channel is private (enabled for playback authorization)."</li> </ul>	March 16, 2021
PutMetadata	Added a minimum length (1) for the metadata request field.	March 4, 2021
Channel latency mode	In Create/UpdateChannel and Channel/ChannelSummary objects, added a description of latencyMode values.	December 18, 2020
Channel default values	<ul style="list-style-type: none"> <li>In Channel data type, add default value for authorized .</li> <li>In Channel data type and CreateChannel, add default value for type.</li> </ul>	December 17, 2020
All List endpoints	Indicated that the maxResults request field has a default value, 50.	December 5, 2020

API Change	Description	Date
Stream & StreamSummary objects	Changed the description of the <code>viewerCount</code> field to indicate that a value of -1 indicates that the request timed out; in this case, retry.	November 10, 2020
Authentication	Added Sigv4 signing info. See "Authentication" in the <a href="#">Welcome</a> section.	October 9, 2020
DeleteChannel & DeleteStreamKey	Changed the HTTP response from 200 to 204.	August 26, 2020
DeleteChannel	Clarified how to delete a channel that's live, to avoid an error.	August 20, 2020
Playback authorization (for private channels)	<ul style="list-style-type: none"> <li>• New PlaybackKeyPair endpoints</li> <li>• A new <code>authorized</code> field in the Channel and ChannelSummary objects</li> <li>• New objects, PlaybackKeyPair and PlaybackKeyPairSummary</li> </ul>	August 19, 2020
New service and <a href="#">API Reference</a>	This is the initial release of Amazon Interactive Video Service (IVS).	July 15, 2020

## Stage API Reference Changes

API Change	Description	Date
IVS Real-Time Streaming launch	Major documentation changes accompany this release. We renamed the previous documentation to be IVS Low-Latency Streaming and published new IVS Real-Time Streaming documentation. The <a href="#">IVS documentation landing page</a> now has separate sections for real-time streaming and low-latency	August 7, 2023



API Change	Description	Date
	<p>streaming. Each section has its own User Guide and API Reference.</p> <p>The Stage API Reference is part of IVS real-time streaming documentation, where it was renamed <a href="#">IVS Real-Time Streaming API Reference</a>. We will list future updates to this API Reference in <a href="#">Document History (Real-Time Streaming)</a>, not here.</p>	
Stage Health	<ul style="list-style-type: none"> <li>Added five endpoints: GetParticipant, ListParticipants, GetStageSession, ListStageSessions, ListParticipantEvents.</li> <li>Added five objects: Event, Participant, ParticipantSummary, StageSession, StageSessionSummary.</li> </ul>	May 11, 2023
New functionality	This is the initial release of the stage API. We added a <a href="#">Stage API Reference</a> tile to the documentation landing page.	March 23, 2023

## IVS Chat API Documentation Changes

API Change	Description	Date
New error message	Added the ConflictException error to UpdateLoggingConfiguration.	March 17, 2023
maximumMessageRatePerSecond max value	Changed the maximum value of maximumMessageRatePerSecond from 10 to 100. This affects the CreateRoom, GetRoom, and UpdateRoom endpoints.	January 30, 2023
Event (Subscribe) MessageID field	In the Chat Messaging API Reference, in <a href="#">Event (Subscribe)</a> , added a deprecated field (MessageID) to Attributes. This is included for backward compatibility.	January 25, 2023

API Change	Description	Date
New error type	Added the PendingVerification error for DeleteMessage, DisconnectUser, & SendEvent.	December 5, 2022
WebSocket errors	In the <a href="#">Chat Messaging API Reference</a> , we updated descriptions of two WebSocket errors (Unauthorized & Forbidden).	November 18, 2022
Chat Logging	<p>Initial release of this new functionality. We added the following:</p> <ul style="list-style-type: none"> <li>• A new resource, LoggingConfiguration</li> <li>• Five data types: CloudWatchDestinationConfiguration, DestinationConfiguration, KinesisDestinationConfiguration, LoggingConfigurationSummary, S3DestinationConfiguration</li> <li>• Five endpoints: Create/Delete/Get/List/UpdateLoggingConfiguration(s)</li> <li>• The loggingConfigurationIdentifiers field to the RoomSummary object and Room requests/responses</li> </ul>	November 17, 2022
CreateChatToken description	Updated the description of CreateChatToken, including new guidance on using the attributes field.	November 17, 2022
Restrictions on tags	Updated information on tags restrictions. Amazon IVS Chat has no constraints on tags beyond what is documented in the AWS documentation that we link to. This affects the "Welcome" section, four endpoints (CreateRoom, ListTagsForResource, TagResource, UntagResource), and the RoomSummary data type.	August 12, 2022

API Change	Description	Date
New functionality	<p>This is the initial release of Amazon IVS Chat. We added an <i>Amazon IVS Chat</i> section of the documentation landing page, with two API reference documents :</p> <ul style="list-style-type: none"><li>• <a href="#">Chat API Reference</a> -- Control-plane API (HTTPS)</li><li>• <a href="#">Chat Messaging API Reference</a> -- Data-plane API (WebSocket)</li></ul>	April 26, 2022

# Release Notes (Low-Latency Streaming)

## June 13, 2024

### Amazon IVS Broadcast SDK: Android 1.19.0, iOS 1.19.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.19.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.19.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.19.0/android/</a></p> <ul style="list-style-type: none"><li>Recent Android versions require an icon in the notification that is displayed when capturing the screen. If desired, you can now customize the icon by calling <code>setSmallIcon</code> on the <code>Notification.Builder</code> returned by <code>Session#createServiceNotificationBuilder</code>.</li></ul>
<a href="#">iOS Broadcast SDK 1.19.0</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.19.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.19.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.19.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.19.0/ios/</a></p> <ul style="list-style-type: none"><li>The iOS Broadcast SDK now requires iOS 13+ as the native platform.</li></ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.696 MB	4.768 MB
armeabi-v7a	1.508 MB	3.390 MB
x86_64	1.840 MB	5.315 MB
x86	1.827 MB	5.038 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.834 MB	2.081 MB

# June 13, 2024

## IVS Broadcast SDK: Web 1.13.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.13.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"><li>• Minor bug fixes.</li></ul>

# June 6, 2024

## IVS Player SDK: Android 1.29.0, iOS 1.29.0

Platform	Downloads and Changes
<a href="#">Android player 1.29.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.29.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.29.0/android/</a></p> <ul style="list-style-type: none"> <li>Added the <code>getChannelMetadata()</code> method, which returns a list of String values to communicate channel features.</li> </ul>
<a href="#">iOS Player 1.29.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.29.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.29.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.29.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.29.0/ios/</a></p> <ul style="list-style-type: none"> <li>The iOS 12 deprecation that was planned for iOS player 1.29.0 is delayed until version 1.30.0.</li> <li>Added the <code>getChannelMetadata()</code> method, which returns a list of String values to communicate channel features.</li> <li>Improved consistency when deallocating <code>IVSPlayerLayer</code> on a background queue.</li> </ul>

### Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.044 MB	2.834 MB
armeabi-v7a	0.916 MB	1.969 MB
x86_64	1.127 MB	2.919 MB

Architecture	Compressed Size	Uncompressed Size
x86	1.097 MB	2.976 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.465 MB	1.183 MB

## June 6, 2024

### IVS Player SDK: Web 1.29.0

Platform	Downloads and Changes
<a href="#">Web player 1.29.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.29.0/amazon-ivs-player.min.js">https://player.live-video.net/1.29.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.29.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.29.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.29.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.29.0/web/</a></p> <ul style="list-style-type: none"> <li>• Minor bug fixes.</li> </ul>

## May 20, 2024

### IVS Broadcast SDK: Web 1.12.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.12.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>

## May 16, 2024

### Amazon IVS Broadcast SDK: Android 1.18.0, iOS 1.18.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.18.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.18.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.18.0/android/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes.</li> </ul>
<a href="#">iOS Broadcast SDK 1.18.0</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.18.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.18.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.18.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.18.0/ios/</a></p> <ul style="list-style-type: none"> <li>Added the <code>IVSCamera setVideoZoomFactor</code> method and the associated <code>IVSCameraDelegate</code> methods.</li> <li>Minor bug fixes.</li> </ul>



## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.692 MB	4.758 MB
armeabi-v7a	1.504 MB	3.382 MB
x86_64	1.834 MB	5.304 MB
x86	1.822 MB	5.026 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.834 MB	2.064 MB

# May 9, 2024

## IVS Player SDK: Web 1.28.0

Platform	Downloads and Changes
<a href="#">Web player 1.28.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.28.0/amazon-ivs-player.min.js">https://player.live-video.net/1.28.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.28.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.28.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.28.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.28.0/web/</a></p> <ul style="list-style-type: none"> <li>• Minor bug fixes.</li> </ul>

# May 9, 2024

## IVS Player SDK: Android 1.28.0, iOS 1.28.0

Platform	Downloads and Changes
<a href="#">Android player 1.28.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.28.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.28.0/android/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<a href="#">iOS Player 1.28.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.28.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.28.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.28.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.28.0/ios/</a></p> <ul style="list-style-type: none"> <li>Added the <code>CF_RETURNS_RETAINED</code> annotation to the <code>IVSPlayerLayer.copyDisplayedPixelBuffer</code> method for <a href="#">automatic memory management when using Swift</a>.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.045 MB	2.830 MB
armeabi-v7a	0.918 MB	1.967 MB
x86_64	1.127 MB	2.913 MB
x86	1.097 MB	2.971 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.468 MB	1.199 MB

## May 6, 2024

### IVS Broadcast SDK: Web 1.11.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.11.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"><li>• No changes.</li></ul>

## April 30, 2024

### IVS Broadcast SDK: Web 1.10.1 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.10.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"><li>• Minor bug fixes.</li></ul>

# April 30, 2024

## Amazon IVS Broadcast SDK: Android 1.15.2, iOS 1.15.2 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.15.2</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.15.2/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.15.2/android/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes. Upgrade to this version only if you have a specific reason to do so; otherwise, use the highest version that is released.</li> </ul>
<a href="#">iOS Broadcast SDK 1.15.2</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.15.2/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.15.2/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.15.2/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.15.2/ios/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes. Upgrade to this version only if you have a specific reason to do so; otherwise, use the highest version that is released.</li> </ul>

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.678 MB	4.723 MB		
armeabi-v7a	1.492 MB	3.356 MB		

Architecture	Compressed Size	Uncompressed Size		
x86_64	1.819 MB	5.267 MB		
x86	1.808 MB	4.991 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	0.813 MB	2.001 MB		

## April 22, 2024

## Amazon IVS Broadcast SDK: Android 1.17.0, iOS 1.17.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.17.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.17.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.17.0/android/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<a href="#">iOS Broadcast SDK 1.17.0</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.17.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.17.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.17.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.17.0/ios/</a></p>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>The <code>AmazonIVSBroadcast</code> framework now includes a privacy manifest, as required by Apple.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.692 MB	4.757 MB		
armeabi-v7a	1.504 MB	3.381 MB		
x86_64	1.834 MB	5.303 MB		
x86	1.822 MB	5.025 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	0.831 MB	2.047 MB		

## April 11, 2024

## Amazon IVS Player SDK: Mobile & Web 1.27.0

Platform	Downloads and Changes
<a href="#">Web player 1.27.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a>

Platform	Downloads and Changes
	<p><b>Script asset:</b> <a href="https://player.live-video.net/1.27.0/amazon-ivs-player.min.js">https://player.live-video.net/1.27.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.27.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.27.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.27.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.27.0/web/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<p><a href="#">Android player 1.27.0</a></p>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.27.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.27.0/android/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<p><a href="#">iOS Player 1.27.0</a></p>	<p><b>Download:</b> <a href="https://player.live-video.net/1.27.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.27.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.27.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.27.0/ios/</a></p> <ul style="list-style-type: none"> <li>The AmazonIVSPlayer framework now includes a privacy manifest, as required by Apple.</li> <li>Fixed key value observation for several properties on the IVSPlayer class.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.044 MB	2.826 MB
armeabi-v7a	0.916 MB	1.963 MB
x86_64	1.096 MB	2.965 MB

Architecture	Compressed Size	Uncompressed Size
x86	1.124 MB	2.907 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.445 MB	1.131 MB

## April 4, 2024

### Secure Reliable Transport (SRT) Ingest Support

Amazon IVS introduces support for streaming using the [SRT](#) protocol. SRT is an open-source transport technology optimized for live audio/video streaming. SRT enables secure and reliable transport of content across unpredictable, noisy networks, like the Internet. SRT offers multiple benefits when transporting live video content over the internet:

- It helps compensate for jitter and bandwidth fluctuations.
- It is resilient to packet loss.
- It supports AES encryption to protect content in transit.

We support H.264-encoded video content using the SRT protocol.



# March 21, 2024

## Amazon IVS Broadcast SDK: Android 1.16.0, iOS 1.16.0, Web 1.10.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.10.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<a href="#">Android Broadcast SDK 1.16.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.16.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.16.0/android/</a></p> <ul style="list-style-type: none"> <li>Fixed a previews freeze on the Exynos variant of Samsung devices with Android 14.</li> </ul>
<a href="#">iOS Broadcast SDK 1.16.0</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.16.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.16.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.16.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.16.0/ios/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes.</li> </ul>

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.683 MB	4.730 MB		
armeabi-v7a	1.498 MB	3.362 MB		

Architecture	Compressed Size	Uncompressed Size		
x86_64	1.824 MB	4.998 MB		
x86	1.813 MB	5.274 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	0.813 MB	2.001 MB		

# March 14, 2024

## Amazon IVS Player SDK 1.26.0

Platform	Downloads and Changes
<a href="#">Web player 1.26.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.26.0/amazon-ivs-player.min.js">https://player.live-video.net/1.26.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.26.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.26.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.26.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.26.0/web/</a></p> <ul style="list-style-type: none"> <li>Added an API to expose the synchronization time.</li> <li>Added an event to indicate when the syncTime has changed.</li> </ul>

Platform	Downloads and Changes
<a href="#">Android player 1.26.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.26.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.26.0/android/</a></p> <ul style="list-style-type: none"> <li>• Added an API to expose the synchronization time.</li> <li>• Added an event to indicate when the syncTime has changed.</li> </ul>
<a href="#">iOS Player 1.26.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.26.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.26.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.26.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.26.0/ios/</a></p> <ul style="list-style-type: none"> <li>• Added an API to expose the synchronization time.</li> <li>• Added an event to indicate when the syncTime has changed.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.07 MB	2.969 MB
armeabi-v7a	0.943 MB	2.098 MB
x86_64	1.123 MB	3.107 MB
x86	1.151 MB	3.039 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.44 MB	1.11 MB

## March 13, 2024

## Amazon IVS Broadcast SDK: Android 1.15.1, iOS 1.15.1 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.15.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.15.1/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.15.1/android/</a></p> <ul style="list-style-type: none"> <li>No changes in this release.</li> </ul>
<a href="#">iOS Broadcast SDK 1.15.1</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.15.1/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.15.1/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.15.1/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.15.1/ios/</a></p> <ul style="list-style-type: none"> <li>No changes in this release.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.678 MB	4.723 MB		
armeabi-v7a	1.492 MB	3.356 MB		

Architecture	Compressed Size	Uncompressed Size		
x86_64	1.808 MB	4.991 MB		
x86	1.819 MB	5.267 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	0.813 MB	2.001 MB		

## February 29, 2024

### Amazon IVS Player SDK: Web 1.25.0

Platform	Downloads and Changes
<a href="#">Web player 1.25.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.25.0/amazon-ivs-player.min.js">https://player.live-video.net/1.25.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.25.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.25.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.25.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.25.0/web/</a></p> <ul style="list-style-type: none"> <li>• Minor bug fixes.</li> </ul>

## February 22, 2024

### Amazon IVS Broadcast SDK: Android 1.15.0, iOS 1.15.0, Web 1.9.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.9.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<a href="#">Android Broadcast SDK 1.15.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.15.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.15.0/android/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes.</li> </ul>
<a href="#">iOS Broadcast SDK 1.15.0</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.15.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.15.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.15.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.15.0/ios/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes.</li> </ul>

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.678 MB	4.723 MB		
armeabi-v7a	1.492 MB	3.356 MB		

Architecture	Compressed Size	Uncompressed Size		
x86_64	1.808 MB	4.991 MB		
x86	1.819 MB	5.267 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	0.813 MB	2.001 MB		

## February 15, 2024

### Amazon IVS Player SDK: Mobile 1.25.0

Platform	Downloads and Changes
<a href="#">Android player 1.25.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.25.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.25.0/android/</a></p> <ul style="list-style-type: none"> <li>• Minor bug fixes.</li> </ul>
<a href="#">iOS Player 1.25.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.25.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.25.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.25.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.25.0/ios/</a></p> <ul style="list-style-type: none"> <li>• Minor bug fixes.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.047 MB	2.9 MB
armeabi-v7a	0.921 MB	2.047 MB
x86_64	1.128 MB	2.97 MB
x86	1.1 MB	3.036 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.42 MB	1.08 MB

## February 1, 2024

### Amazon IVS Broadcast SDK: Android 1.14.1, iOS 1.14.1, Web 1.8.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.8.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>
<a href="#">Android Broadcast SDK 1.14.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.14.1/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.14.1/android/</a></p> <ul style="list-style-type: none"> <li>Minor bug fixes and improvements.</li> </ul>



Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.14.1</a>	<p><b>Download for low-latency streaming:</b>  <a href="https://broadcast.live-video.net/1.14.1/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.14.1/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.14.1/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.14.1/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed multiple stability issues on iOS 12.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.663 MB	4.708 MB		
armeabi-v7a	1.482 MB	3.350 MB		
x86_64	1.804 MB	5.246 MB		
x86	1.793 MB	4.973 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	1.640 MB	4.010 MB		

## January 31, 2024

### Tokenless Playback Restrictions

This release enables origin enforcement and geofencing outside of playback authorization. The IVS Low-Latency Streaming User Guide and API Reference were updated; see the [Document History](#) for details of the changes.

## January 25, 2024

### Audio-Only Playback

IVS now fully supports audio-only playback. See [Audio-Only Playback](#) in the IVS Player overview and [Audio-Only Playback](#) in the IVS Web Player Guide.

## January 18, 2024

### Amazon IVS Player SDK 1.24.0

Platform	Downloads and Changes
<a href="#">Web player 1.24.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.24.0/amazon-ivs-player.min.js">https://player.live-video.net/1.24.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.24.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.24.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.24.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.24.0/web/</a></p> <ul style="list-style-type: none"><li>• Added support for audio-only playback. Audio-only quality must be selected manually with <code>setQuality()</code> ; it will not be selected automatically in auto quality</li></ul>

Platform	Downloads and Changes
	mode. See <a href="#">Audio-Only Playback</a> in the <i>Player Web Guide</i> .
<a href="#">Android player 1.24.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.24.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.24.0/android/</a></p> <ul style="list-style-type: none"> <li>No changes</li> </ul>
<a href="#">iOS Player 1.24.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.24.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.24.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.24.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.24.0/ios/</a></p> <ul style="list-style-type: none"> <li>No changes</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.006 MB	2.846 MB
armeabi-v7a	0.88 MB	1.995 MB
x86_64	1.085 MB	2.916 MB
x86	1.058 MB	2.982 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.43 MB	1.08 MB

# January 3, 2024

## Amazon IVS Broadcast SDK: Android 1.13.4, iOS 1.13.4, Web 1.7.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.7.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference/</a></p> <ul style="list-style-type: none"> <li>No changes to the low-latency SDK.</li> </ul>
<a href="#">Android Broadcast SDK 1.13.4</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.4/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.4/android/</a></p> <ul style="list-style-type: none"> <li>No changes to the low-latency SDK.</li> </ul>
<a href="#">iOS Broadcast SDK 1.13.4</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.13.4/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.13.4/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.4/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.4/ios/</a></p> <ul style="list-style-type: none"> <li>No changes to the low-latency SDK.</li> </ul>

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.663 MB	4.704 MB		
armeabi-v7a	1.484 MB	3.352 MB		

Architecture	Compressed Size	Uncompressed Size		
x86_64	1.804 MB	5.243 MB		
x86	1.795 MB	4.97 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	1.63 MB	4.01 MB		

## December 4, 2023

### Amazon IVS Broadcast SDK: Android 1.13.2 and iOS 1.13.2 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.13.2</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.2/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.2/android/</a></p> <ul style="list-style-type: none"> <li>No changes in the low-latency SDK.</li> </ul>
<a href="#">iOS Broadcast SDK 1.13.2</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.13.2/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.13.2/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.2/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.2/ios/</a></p> <ul style="list-style-type: none"> <li>No changes in the low-latency SDK.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.663 MB	4.704 MB		
armeabi-v7a	1.484 MB	3.352 MB		
x86_64	1.804 MB	5.243 MB		
x86	1.795 MB	4.970 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	1.63 MB	4.01 MB		

## November 21, 2023

### Amazon IVS Broadcast SDK: Android 1.13.1 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.13.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.1/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.1/android/</a></p> <ul style="list-style-type: none"> <li>No changes in the low-latency SDK.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.663 MB	4.705 MB		
armeabi-v7a	1.484MB	3.352 MB		
x86_64	1.804 MB	5.243 MB		
x86	1.795 MB	4.971 MB		

## November 17, 2023

### Amazon IVS Broadcast SDK: Android 1.13.0 and iOS 1.13.0 (Low-Latency Streaming)

Platform	Downloads and Changes
All mobile (Android and iOS)	<ul style="list-style-type: none"> <li>Improved IPv6 support by adopting RFC 6555 "Happy Eyeballs" and adding the <code>BroadcastConfiguration.network.useIPv6</code> configuration option to enable or disable IPv6 for broadcasting.</li> </ul>
<a href="#">Android Broadcast SDK 1.13.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.0/android/</a></p> <ul style="list-style-type: none"> <li>Fixed a crash when an <code>AudioSource</code> object is used after releasing it.</li> <li>Added support to <code>Surfaceview</code>-based preview for better performance. The existing <code>getPreview</code> methods in <code>Session</code> and <code>StageStream</code> continue to return a</li> </ul>

Platform	Downloads and Changes
	<p>subclass of <code>TextureView</code> , but this may change in a future SDK version.</p> <ul style="list-style-type: none"> <li>• If your application depends on <code>TextureView</code> specifically, you can continue with no changes. You also can switch from <code>getPreview</code> to <code>getPreviewTextureView</code> to prepare for the eventual change of what the default <code>getPreview</code> returns.</li> <li>• If your application does not require <code>TextureView</code> specifically, we recommend switching to <code>getPreviewSurfaceView</code> for lower CPU and memory usage.</li> <li>• The SDK now implements a new type of preview called <code>ImagePreviewSurfaceTarget</code> which works with the application-provided Android Surface object. It is not a subclass of Android View, which provides better flexibility.</li> </ul>
<p><a href="#">iOS Broadcast SDK 1.13.0</a></p>	<p><b>Download for low-latency streaming:</b>  <a href="https://broadcast.live-video.net/1.13.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.13.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.13.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.13.0/ios/</a></p> <ul style="list-style-type: none"> <li>• There were no changes for this release.</li> </ul>



## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.66 MB	4.70 MB		
armeabi-v7a	1.48 MB	3.35 MB		
x86_64	1.80 MB	5.24 MB		
x86	1.79 MB	4.96 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	1.63 MB	4.01 MB		

# November 14, 2023

## Amazon IVS Player SDK 1.23.0

Platform	Downloads and Changes
<a href="#">Web player 1.23.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.23.0/amazon-ivs-player.min.js">https://player.live-video.net/1.23.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.23.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.23.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.23.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.23.0/web/</a></p>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>Added support for low-latency playback in iOS Safari.</li> </ul>
<a href="#">Android player 1.23.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.23.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.23.0/android/</a></p> <ul style="list-style-type: none"> <li>Updated the reference documentation with a new UI and more details.</li> </ul>
<a href="#">iOS Player 1.23.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.23.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.23.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.23.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.23.0/ios/</a></p> <ul style="list-style-type: none"> <li>No changes.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	0.975 MB	2.744 MB
armeabi-v7a	0.853 MB	1.917 MB
x86_64	1.028 MB	2.873 MB
x86	1.055 MB	2.811 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.39 MB	0.93 MB

## October 16, 2023

### Amazon IVS Broadcast SDK: Web 1.6.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.6.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>No changes to the low-latency SDK.</li> </ul>

## October 12, 2023

### Amazon IVS Broadcast SDK: Android 1.12.1 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.12.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.12.1/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.12.1/android/</a></p> <ul style="list-style-type: none"> <li>Fixed a bug where calling <code>BroadcastSession.setListener</code> resulted in an error.</li> </ul>

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.685 MB	5.046 MB		
armeabi-v7a	1.503 MB	3.702 MB		
x86_64	1.826 MB	5.576 MB		

Architecture	Compressed Size	Uncompressed Size		
x86	1.822 MB	5.290 MB		

## October 3, 2023

### Amazon IVS Player SDK 1.22.0

Platform	Downloads and Changes
<a href="#">Web player 1.22.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.22.0/amazon-ivs-player.min.js">https://player.live-video.net/1.22.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.22.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.22.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.22.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.22.0/web/</a></p> <ul style="list-style-type: none"> <li>Added a static method to get the SDK version, MediaPlayerPackage.getVersion() .</li> </ul>
<a href="#">Android player 1.22.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.22.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.22.0/android/</a></p> <ul style="list-style-type: none"> <li>Added the setNetworkRecoveryMode function to the Player interface, to set the desired playback behavior after a network interruption.</li> </ul>
<a href="#">iOS Player 1.22.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.22.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.22.0/AmazonIVSPlayer.xcframework.zip</a></p>

Platform	Downloads and Changes
	<p data-bbox="831 214 1490 298"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.22.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.22.0/ios/</a></p> <ul data-bbox="831 340 1490 760" style="list-style-type: none"> <li>• Added the <code>setNetworkRecoveryMode</code> function to the <code>IVSPlayer</code> interface, to set the desired playback behavior after a network interruption.</li> <li>• The <code>copyDisplayedPixelBuffer</code> method on <code>IVSPlayerLayer</code> can now be used when the player is playing. Previously it was callable only when the player was idle.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	0.948 MB	2.676 MB
armeabi-v7a	0.828 MB	1.865 MB
x86_64	1.025 MB	2.741 MB
x86	1.000 MB	2.802 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.37 MB	0.89 MB

## October 2, 2023

### In-Console Streaming

You can now stream from the IVS console. In *Getting Started with Low-Latency Streaming*, see [Step 5: Set Up Streaming Software](#).

## September 14, 2023

### Amazon IVS Broadcast SDK: Web 1.5.2 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.5.2</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a>

## August 23, 2023

### Amazon IVS Broadcast SDK: Web 1.5.1, Android 1.12.0, and iOS 1.12.0 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.5.1</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a>
<a href="#">Android Broadcast SDK 1.12.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.12.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.12.0/android/</a> <ul style="list-style-type: none"> <li>Fixed a rare bug that caused broadcasts to end prematurely with the message "Attempted to recv after receiving shutdown from peer."</li> </ul>

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.12.0</a>	<p><b>Download for low-latency streaming:</b>  <a href="https://broadcast.live-video.net/1.12.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.12.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.12.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.12.0/ios/</a></p> <ul style="list-style-type: none"> <li>Corrected the signature of <code>IVSDeviceDiscovery.createAudioSourceWithName</code> to return an <code>IVSCustomAudioSource</code> instead of <code>IVSCustomImageSource</code>.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.685 MB	5.046 MB		
armeabi-v7a	1.503 MB	3.702 MB		
x86_64	1.826 MB	5.576 MB		
x86	1.822 MB	5.290 MB		

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size		
arm64	1.58 MB	3.88 MB		

## August 23, 2023

### Amazon IVS Broadcast SDK: Android 1.7.6 (Low-Latency Streaming)

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.7.6</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.6/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.6/android/</a></p> <ul style="list-style-type: none"> <li>Fixed a rare bug that caused broadcasts to end prematurely with the message "Attempted to recv after receiving shutdown from peer."</li> </ul>

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size		
arm64-v8a	1.630 MB	4.689 MB		
armeabi-v7a	1.520 MB	3.792 MB		
x86_64	1.761 MB	4.748 MB		
x86	1.825 MB	5.219 MB		

## August 22, 2023

### Amazon IVS Player SDK 1.21.0

Platform	Downloads and Changes
<a href="#">Web player 1.21.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a>



Platform	Downloads and Changes
	<p><b>Script asset:</b> <a href="https://player.live-video.net/1.21.0/amazon-ivs-player.min.js">https://player.live-video.net/1.21.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.21.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.21.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.21.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.21.0/web/</a></p>
<a href="#">Android player 1.21.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.21.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.21.0/android/</a></p>
<a href="#">iOS Player 1.21.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.21.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.21.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.21.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.21.0/ios/</a></p> <ul style="list-style-type: none"> <li>Added support for obtaining the most recently displayed video frame, via the new <code>copyDisplayedPixelBuffer</code> method on the <code>IVSPlayerLayer</code> class.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	0.942 MB	2.662 MB
armeabi-v7a	0.823 MB	1.853 MB
x86_64	1.020 MB	2.726 MB
x86	0.993 MB	2.788 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.36 MB	0.87 MB

## August 7, 2023

### Amazon IVS Broadcast SDK: Web 1.5.0, Android 1.11.0, and iOS 1.11.0

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.5.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>Fixed an issue in Safari where a race condition periodically causes an error in media-track retrieval</li> </ul>
<a href="#">Android Broadcast SDK 1.11.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.11.0/android">https://aws.github.io/amazon-ivs-broadcast-docs/1.11.0/android</a></p>
<a href="#">iOS Broadcast SDK 1.11.0</a>	<p><b>Download for low-latency streaming:</b> <a href="https://broadcast.live-video.net/1.11.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.11.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.11.0/ios">https://aws.github.io/amazon-ivs-broadcast-docs/1.11.0/ios</a></p>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.659 MB	4.918 MB

Architecture	Compressed Size	Uncompressed Size
armeabi-v7a	1.482 MB	3.590 MB
x86_64	1.804 MB	5.444 MB
x86	1.795 MB	5.160 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.771 MB	1.879 MB

## July 17, 2023

## R2S3 Rendition Filtering & Thumbnail Enhancements

IVS customers can now control what renditions are generated for a stream when recording to Amazon S3 and what resolutions are generated for thumbnails. For more information, see:

- [Getting Started with IVS](#) – In "Step 4: Create a Channel" > "Console Instructions," we updated screenshots and instructions.
- [Auto-Record to Amazon S3](#) – In "JSON Metadata Files," we added `latest_thumbnail` and updated `thumbnail`. In "Thumbnails" and "Discovering the Renditions of a Recording," we added rendition-resolution descriptions.
- [Costs](#) – In "Storing Recorded Video," we updated screenshots.
- [IVS API Reference](#):
  - In `ThumbnailConfiguration`, we added `resolution` and `storage`. This affects the `CreateRecordingConfiguration` request and response, `GetRecordingConfiguration` response, and `GetStreamSession` response.
  - In `ThumbnailConfiguration`, we changed the `targetIntervalSeconds` minimum from 5 to 1 and updated the "Important" note to say it applies only to BASIC channels.
  - We added the `RenditionConfiguration` object.

- We added `renditionConfiguration` to the `RecordingConfiguration` object. This affects three responses: `CreateRecordingConfiguration`, `GetRecordingConfiguration`, and `GetStreamSession`. We also added `renditionConfiguration` to the `CreateRecordingConfiguration` request.

## July 14, 2023

### Amazon IVS Player SDK 1.20.0

Platform	Downloads and Changes
<p><a href="#">Web player 1.20.0</a> &amp; <a href="#">Video.js integration</a> &amp; <a href="#">JW player integration</a></p>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.20.0/amazon-ivs-player.min.js">https://player.live-video.net/1.20.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.20.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.20.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.20.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.20.0/web/</a></p> <ul style="list-style-type: none"> <li>• Fixed an issue when playing a live stream or recorded content on an iOS mobile browser, where <code>player.getLiveLatency()</code> calls return 0. (This was fixed starting with Web Player 1.17.0.)</li> <li>• Fixed the type definitions of the <code>amazon-ivs-player</code> npm package.</li> <li>• In the <a href="#">Web Player SDK Reference</a>, added a new landing page and removed duplicate entries.</li> <li>• Added support for Video.js version 8+.</li> </ul>

Platform	Downloads and Changes
<a href="#">Android player 1.20.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.20.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.20.0/android/</a></p>
<a href="#">iOS Player 1.20.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.20.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.20.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.20.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.20.0/ios/</a></p> <ul style="list-style-type: none"> <li>The iOS SDK now requires iOS 12.0 or higher. (iOS 11 is no longer supported.)</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.039 MB	2.922 MB
armeabi-v7a	0.909 MB	2.043 MB
x86_64	1.094 MB	3.069 MB
x86	1.126 MB	3.006 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.41 MB	0.99 MB

# July 13, 2023

## Amazon IVS Broadcast SDK: Web 1.4.0, Android 1.10.0, and iOS 1.10.0

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.4.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>• Fixed a bug where the SDK provided insufficient typing information for consumption by host applications.</li> <li>• Fixed a bug where a combination of <code>leave()</code> and subsequent <code>refreshStrategy()</code> could republish media even though weâ€™ve left.</li> <li>• Fixed a bug where <code>stageStreamsToPublish</code> returning a single track (audio or video) can prevent clean updates when the strategy is refreshed.</li> <li>• Added a faster disconnect flow when the browser tab is closed.</li> </ul>
All mobile (Android and iOS)	<ul style="list-style-type: none"> <li>• Improved the stability of stages by reducing occurrences of rare crashes.</li> <li>• Added a new <code>sendTimedMetadata</code> method to <code>BroadcastSession</code>, which allows sending a string through the same socket connection as the current broadcast. This string has timing information attached and can be received by the IVS Player SDK.</li> <li>• When a participant leaves a stage, the participant now has its published state updated to unpublished before <code>onParticipate</code></li> </ul>

Platform	Downloads and Changes
	<p><code>partLeft</code> is called on Android or <code>participantDidLeave</code> is called on iOS.</p>
<p><a href="#">Android Broadcast SDK 1.10.0</a></p>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.10.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.10.0/android/</a></p> <ul style="list-style-type: none"><li>• Fixed a bug where rotating while reconnecting to a stage caused other participants to appear badly cropped.</li><li>• Fixed an issue where the <code>AudioStageStream</code> device could not be cast to <code>AudioDevice</code>.</li><li>• Fixed an issue where rapid background-to-foreground app switch caused subscribed video streams to be muted.</li></ul>

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.10.0</a>	<p><b>Download without stages:</b> <a href="https://broadcast.live-video.net/1.10.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.10.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Download with stages:</b> <a href="https://broadcast.live-video.net/1.10.0/AmazonIVSBroadcast-Stages.xcframework.zip">https://broadcast.live-video.net/1.10.0/AmazonIVSBroadcast-Stages.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.10.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.10.0/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue where viewers could lose audio after a broadcast session is interrupted by a phone call. The workaround (restarting the broadcast session after a phone-call interruption) is no longer needed.</li> <li>Fixes an issue that prevented multiple stages from existing and all being able to play audio.</li> <li>When network loss happens suddenly, an ongoing Broadcast will now be stopped immediately instead of waiting for the connection to time out.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64-v8a	1.517 MB	4.761 MB	5.324 MB	15.028 MB
armeabi-v7a	1.340 MB	3.433 MB	4.370 MB	9.489 MB



Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
x86_64	1.653 MB	5.003 MB	5.802 MB	15.837 MB
x86	1.662 MB	5.287 MB	5.621 MB	15.964 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64	1.56 MB	3.84 MB	5.04 MB	10.85 MB

## June 28, 2023

### Viewer Session Revocation for Private Channels

IVS customers can now revoke the viewer session associated with an auth token, to prevent and stop playback using that token. For more information, see:

- [Setting Up Private Channels](#) – We changed the "Token Schema" section and added "Revoke Viewer Sessions."
- [IVS API Reference](#) – We added two endpoints (StartViewerSessionRevocation and BatchStartViewerSessionRevocation) and two objects (BatchStartViewerSessionRevocationError and BatchStartViewerSessionRevocationViewerSession).

# June 27, 2023

## Amazon IVS Broadcast SDK: iOS 1.9.1

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.9.1</a>	<p><b>Download without stages:</b> <a href="https://broadcast.live-video.net/1.9.1/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.9.1/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Download with stages:</b> <a href="https://broadcast.live-video.net/1.9.1/AmazonIVSBroadcast-Stages.xcframework.zip">https://broadcast.live-video.net/1.9.1/AmazonIVSBroadcast-Stages.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.9.1/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.9.1/ios/</a></p> <ul style="list-style-type: none"><li>Resolved an issue on iOS 16.5 and above where video bitrate gradually degrades after either: (1) approximately 20 minutes if not using auto-bitrate with b-frames turned off, or (2) approximately 20 minutes from reaching <code>IVSVideoConfiguration.maxBitrate</code> and the network connection has remained stable, with b-frames turned off.</li></ul> <p><b>Known issue:</b> Viewers may lose audio after a broadcast session is interrupted by a phone call. The workaround is to restart the broadcast session after a phone-call interruption.</p>

## Broadcast SDK Size: iOS

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64	1.55 MB	3.77 MB	5.01 MB	10.77 MB

## June 27, 2023

### Amazon IVS Broadcast SDK 1.7.5

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.7.5</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.7.5/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.7.5/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.5/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.5/ios/</a></p> <ul style="list-style-type: none"> <li>Resolved an issue on iOS 16.5 and above where video bitrate gradually degrades after either: (1) approximately 20 minutes if not using auto-bitrate with b-frames turned off, or (2) approximately 20 minutes from reaching <code>IVSVideoConfiguration.maxBitrate</code> and the network connection has remained stable, with b-frames turned off.</li> </ul>

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.48 MB	3.43 MB

## June 16, 2023

### Amazon IVS Broadcast SDK: Web 1.3.3

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.3.3</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>Fixed regressions in internal analytics used to support IVS multiple-hosts health.</li> </ul>

## June 2, 2023

### Advanced Channel Types

This release introduces two new channel types – ADVANCED\_SD and ADVANCED\_HD – in addition to the existing BASIC and STANDARD types. Channel type determines the allowable playback and recording resolution and bitrate.

- **ADVANCED\_SD:** Video is transcoded; multiple qualities are generated from the original input, to automatically give viewers the best experience for their devices and network conditions. Input resolution can be up to 1080p and bitrate can be up to 8.5 Mbps; output is capped at SD quality (480p). You can select an optional transcode preset (see below). Audio for all renditions is transcoded, and an audio-only rendition is available.
- **ADVANCED\_HD:** Video is transcoded; multiple qualities are generated from the original input, to automatically give viewers the best experience for their devices and network conditions. Input resolution can be up to 1080p and bitrate can be up to 8.5 Mbps; output is capped at HD

quality (720p). You can select an optional transcode preset (see below). Audio for all renditions is transcoded, and an audio-only rendition is available.

Optional *transcode presets* for the new channel types allow you to trade off available download bandwidth and video quality, to optimize the viewing experience. There are two presets:

- *Constrained bandwidth delivery* uses a lower bitrate for each quality level. Use it if you have low download bandwidth and/or simple video content (e.g., talking heads).
- *Higher bandwidth delivery* uses a higher bitrate for each quality level. Use it if you have high download bandwidth and/or complex video content (e.g., flashes and quick scene changes).

The [Document History](#) page lists related changes to the IVS User Guide and IVS API Reference.

## June 1, 2023

### Amazon IVS Broadcast SDK: Android 1.9.0 and iOS 1.9.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>• Improved stability of stages by reducing occurrences of rare crashes.</li> <li>• Enhanced automated recovery from recurrent network disruptions.</li> </ul>
<a href="#">Android Broadcast SDK 1.9.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.9.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.9.0/android/</a></p> <ul style="list-style-type: none"> <li>• Fixed a bug where disconnecting wired headphones would result in an error when the user tried to switch to a non-default microphone on some devices.</li> <li>• Fixed a bug on some devices where an incorrect microphone is attached when switching microphones during a broadcast ing session.</li> </ul>

Platform	Downloads and Changes
	<ul style="list-style-type: none"><li>• Fixed a crash when calling Presets methods before creating a Broadcast Session , Stage, or DeviceDiscovery object.</li></ul>

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.9.0</a>	<p><b>Download without stages:</b> <a href="https://broadcast.live-video.net/1.9.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.9.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Download with stages:</b> <a href="https://broadcast.live-video.net/1.9.0/AmazonIVSBroadcast-Stages.xcframework.zip">https://broadcast.live-video.net/1.9.0/AmazonIVSBroadcast-Stages.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.9.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.9.0/ios/</a></p> <ul style="list-style-type: none"><li>• When you have joined a Stage and have not attached a microphone to the Stage, the new default behavior when entering the background is to stay in the Stage instead of leaving automatically. This enables the use case of listening to a Stage as a viewer-only participant while in the background.</li><li>• Improved the handling of a stage's Bluetooth device connect/disconnect.</li><li>• Fixed an issue where audio is much lower when using <code>setGain</code> to mute and unmute.</li><li>• When attaching a camera to a <code>IVSBroadcastSession</code>, the camera now configures itself based on the <code>size</code> and <code>targetFrameRate</code> on the <code>IVSVideoConfiguration</code>.</li><li>• The iOS SDK now requires iOS 12.0 or higher. (iOS 11 is no longer supported.)</li></ul> <p><b>Known issue:</b> Viewers may lose audio after a broadcast session is interrupted by a phone call. The workaround is to restart the</p>

Platform	Downloads and Changes
	broadcast session after a phone-call interruption.

## Broadcast SDK Size: Android

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64-v8a	1.638 MB	4.846 MB	5.451 MB	14.778 MB
armeabi-v7a	1.461 MB	3.532 MB	4.506 MB	9.475 MB
x86_64	1.770 MB	5.082 MB	5.753 MB	15.904 MB
x86	1.781 MB	5.366 MB	5.919 MB	15.708 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64	1.55 MB	3.77 MB	5.00 MB	10.77 MB



# May 23, 2023

## Amazon IVS Player SDK 1.19.0

Platform	Downloads and Changes
<a href="#">Web player 1.19.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.19.0/amazon-ivs-player.min.js">https://player.live-video.net/1.19.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.19.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.19.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.19.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.19.0/web/</a></p>
<a href="#">Android player 1.19.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.19.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.19.0/android/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue in auto quality mode, where the player stayed in the lowest quality after rebuffering, even when there was enough bandwidth to switch up.</li> </ul>
<a href="#">iOS Player 1.19.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.19.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.19.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.19.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.19.0/ios/</a></p>

### Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.013 MB	2.866 MB

Architecture	Compressed Size	Uncompressed Size
armeabi-v7a	0.919 MB	2.272 MB
x86_64	1.084 MB	3.001 MB
x86	1.058 MB	2.702 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.41 MB	0.99 MB

## May 16, 2023

### Amazon IVS Broadcast SDK: iOS 1.8.1

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.8.1</a>	<p><b>Download without stages:</b> <a href="https://broadcast.live-video.net/1.8.1/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.8.1/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Download with stages:</b> <a href="https://broadcast.live-video.net/1.8.1/AmazonIVSBroadcast-Stages.xcframework.zip">https://broadcast.live-video.net/1.8.1/AmazonIVSBroadcast-Stages.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.8.1/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.8.1/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed a bitrate degradation issue on iOS 16.4.1, for both RTMP (without stages) and WebRTC (with stages). If you had implemented a workaround on your app (by enabling</li> </ul>

Platform	Downloads and Changes
	b frame), you can remove it after installing this update.

## Broadcast SDK Size: iOS

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64	1.53 MB	3.73 MB	5.00 MB	10.73 MB

## May 16, 2023

### Amazon IVS Broadcast SDK 1.7.4

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.7.4</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.7.4/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.7.4/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.4/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.4/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed a bitrate degradation issue on iOS 16.4. If you had implemented a workaround on your app (by enabling b frame), you can remove it after installing this update.</li> </ul>

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.48 MB	3.40 MB

## May 11, 2023

### Multiple Hosts Health

Now you can monitor the health of your IVS stages with multiple hosts. See:

- [Monitoring Stage Health](#) – This is a new *Amazon IVS User Guide* page.
- [Using Amazon EventBridge with Amazon IVS](#) – We added two Stage Update events.
- [IVS Service Quotas](#) – We added call-rate quotas for the new endpoints.
- [IVS Stage API Reference](#) – We added five endpoints (GetParticipant, ListParticipants, GetStageSession, ListStageSessions, ListParticipantEvents) and five objects (Event, Participant, ParticipantSummary, StageSession, StageSessionSummary).

## May 1, 2023

### Amazon IVS Web Broadcast SDK 1.3.2

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.3.2</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>• Fixed an issue where broadcasting a screenshare sometimes resulted in a black screen for live channels.</li> <li>• Fixed an issue where broadcasting a stage participant sometimes resulted in a black screen for live channels.</li> </ul>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>• Fixed an analytics issue where duplicate unpublish and publish events were reported.</li> <li>• Fixed an issue where <code>getStats</code> was not always updated for <code>RemoteStageStream</code> objects.</li> <li>• Fixed an <code>OverconstrainedError</code> when trying to broadcast stage participants.</li> <li>• Added an enhancement: subscribe-only participants are ignored when the stage strategy <code>shouldPublishParticipant</code> is set to <code>true</code>.</li> </ul>

## April 27, 2023

### Stage Participant Increase

The maximum number of participants who can be connected to a stage at once was increased from 12 to 1,000. At most 12 participants can be publishing to a stage at once and at most 1,000 can be subscribing at once. For more information see [Enabling Multiple Hosts on an Amazon IVS Stream](#) and [Amazon IVS Service Quotas](#).

## April 4, 2023

### Amazon IVS Player SDK 1.18.0

Platform	Downloads and Changes
<a href="#">Web player 1.18.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.18.0/amazon-ivs-player.min.js">https://player.live-video.net/1.18.0/amazon-ivs-player.min.js</a></p>

Platform	Downloads and Changes
	<p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.18.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.18.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.18.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.18.0/web/</a></p> <ul style="list-style-type: none"> <li>Fixed a Safari issue where after refresh, in the console tab, "HTTP Response Error" – "Load failed" was displayed.</li> </ul>
<p><a href="#">Android player 1.18.0</a></p>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.18.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.18.0/android/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue with video playback when the playback rate is greater than 1x.</li> </ul>
<p><a href="#">iOS Player 1.18.0</a></p>	<p><b>Download:</b> <a href="https://player.live-video.net/1.18.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.18.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.18.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.18.0/ios/</a></p>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.011 MB	2.854 MB
armeabi-v7a	0.916 MB	2.261 MB
x86_64	1.082 MB	2.990 MB
x86	1.055 MB	2.691 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.41 MB	0.99 MB

## March 30, 2023

### RTMP Support

Amazon IVS now supports RTMP (Real-Time Messaging Protocol) streaming, in addition to RTMPS. RTMPS is the secure version of RTMP. We recommend using RTMPS for secure ingest, unless you have specific and verified use cases that require RTMP.

RTMP streaming can be set up via:

- IVS console – Use the **Custom configuration** button during initial channel setup or the **Enable RTMP ingest** toggle when modifying an existing channel.
- API – Use the new `insecureIngest` field in `CreateChannel` or `UpdateChannel` requests. See the [IVS API Reference](#).

For information on RTMP ingest endpoints, see [Set Up Streaming Software](#), [Broadcast Android SDK Guide](#), and [Broadcast iOS SDK Guide](#).

**April 29, 2023 correction:** We changed [Broadcast Android SDK Guide](#) and [Broadcast iOS SDK Guide](#) to indicate that these SDKs support only RTMPS ingest (not insecure RTMP ingest).

## March 29, 2023

### Single-Use Tokens for Private Channels

In [Generate and Sign Playback Tokens](#), we added to the payload an optional field, `single-use-uuid`, for generating a single-use token.

## March 28, 2023

### Amazon IVS Web Broadcast SDK 1.3.1

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK 1.3.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"><li>• There were no customer-facing changes in this release.</li></ul>

## March 23, 2023

### Support for Multiple Hosts on a Stream (Stage Resource)

This is the first release of new functionality: you can now combine video from multiple participants into one live stream. A *stage* is a virtual space where participants can exchange audio and video in real time. You can then broadcast a stage to channels to reach a larger audience, and you can build applications where audience members can be brought "on stage" to contribute to the live conversation. For details, see:

- [Enabling Multiple Hosts on an IVS Stream](#) (new document)
- [Stage API Reference](#) (new document)
- [Service Quotas](#) (see "Amazon IVS Stage" endpoints and stage limits in Other Quotas > Amazon IVS)
- Documentation changes for the simultaneous release of [Amazon IVS Broadcast SDK 1.8.0](#)

## March 23, 2023

### Amazon IVS Broadcast SDK: Android 1.8.0, iOS 1.8.0, Web 1.3.0

In conjunction with adding support for multiple hosts on a stream, the Android and iOS Broadcast SDKs were updated to support the new stage functionality.



Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Added stage support. See <a href="#">Support for Multiple Hosts on a Stream (Stage Resource)</a>.</li> <li>In the <a href="#">Broadcast SDK overview</a>, added stage platform requirements.</li> </ul>
<a href="#">Android Broadcast SDK 1.8.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.8.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.8.0/android/</a></p> <ul style="list-style-type: none"> <li>Depending on how your gradle file is configured (using classifier), you can download the SDK without or without stage support. The SDK without stage support is smaller. For details, see <a href="#">Broadcast SDK: Android Guide</a>.</li> <li>In <a href="#">Broadcast SDK: Android Guide</a>, added "Add Multiple Hosts with the Stage SDK" and stage-related "Known Issues and Workarounds."</li> </ul>
<a href="#">iOS Broadcast SDK 1.8.0</a>	<p><b>Download without stages:</b> <a href="https://broadcast.live-video.net/1.8.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.8.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Download with stages:</b> <a href="https://broadcast.live-video.net/1.8.0/AmazonIVSBroadcast-Stages.xcframework.zip">https://broadcast.live-video.net/1.8.0/AmazonIVSBroadcast-Stages.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.8.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.8.0/ios/</a></p> <ul style="list-style-type: none"> <li>Depending on how your pod file is configured, you can download the SDK with or without stage support. The SDK without</li> </ul>

Platform	Downloads and Changes
	<p>stage support is smaller. For installation details, see <a href="#">Broadcast SDK: iOS Guide</a>.</p> <ul style="list-style-type: none"> <li>In <a href="#">Broadcast SDK: iOS Guide</a>, added "Add Multiple Hosts with the Stage SDK" and stage-related "Known Issues and Workarounds."</li> <li>Deprecated bitcode support from the SDK, as Apple has officially deprecated bitcode and no longer accepts it for App Store submissions. For more information, see the <a href="#">Xcode 14 release notes</a>.</li> </ul>
<a href="#">Web Broadcast SDK 1.3.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>In <a href="#">Broadcast SDK: Web Guide</a>, added "Add Multiple Hosts with the Stage SDK" and stage-related "Known Issues and Workarounds."</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64-v8a	1.767 MB	5.192 MB	5.886 MB	16.398 MB
armeabi-v7a	1.656 MB	4.263 MB	4.946 MB	10.924 MB
x86_64	1.967 MB	5.735 MB	6.316 MB	17.376 MB
x86	1.894 MB	5.196 MB	6.387 MB	16.730 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size without Stage Functionality	Uncompressed Size without Stage Functionality	Compressed Size with Stage Functionality	Uncompressed Size with Stage Functionality
arm64	1.53 MB	3.73 MB	5.03 MB	10.67 MB

## March 2, 2023

### Amazon IVS Broadcast SDK: Android 1.7.3

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.7.3</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.3/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.3/android/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue where custom image sources failed to work properly on devices with the MediaTek Dimensity 700 SoC.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.629 MB	4.688 MB
armeabi-v7a	1.520 MB	3.792 MB
x86_64	1.825 MB	5.218 MB
x86	1.629 MB	4.688 MB

# February 28, 2023

## Amazon IVS Player SDK 1.17.0

Platform	Downloads and Changes
<a href="#">Web player 1.17.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.17.0/amazon-ivs-player.min.js">https://player.live-video.net/1.17.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.17.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.17.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.17.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.17.0/web/</a></p> <ul style="list-style-type: none"><li>Implemented support for the <code>getLiveLatency</code> method for mobile Safari.</li></ul>
<a href="#">Android player 1.17.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.17.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.17.0/android/</a></p>
<a href="#">iOS Player 1.17.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.17.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.17.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.17.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.17.0/ios/</a></p> <ul style="list-style-type: none"><li>Deprecated bitcode support from the SDK, as Apple has officially deprecated bitcode and no longer accepts it for App Store submissions. For more information, see the <a href="#">Xcode 14 release notes</a>.</li></ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.009 MB	2.853 MB
armeabi-v7a	0.915 MB	2.260 MB
x86_64	1.081 MB	2.988 MB
x86	1.054 MB	2.690 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.41 MB	0.99 MB

## February 16, 2023

### Byte-Range Tags and Manifest Files for Auto-Record to S3

The auto-record-to-S3 feature now supports [byte-range playlist](#) generation, in addition to standard HLS playlists. For more information, see [Auto-Record to Amazon S3](#) ("Recording Contents," "Byte-Range Playlists," and new `byte_range_playlist` fields in JSON examples for `recording_started` and `recording_ended`).

## January 31, 2023

### Amazon IVS Chat Client Messaging SDK: Android 1.1.0

Platform	Downloads and Changes
<a href="#">Android Chat Client Messaging SDK 1.1.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-chat-messaging-sdk-android/1.1.0/">https://aws.github.io/amazon-ivs-chat-messaging-sdk-android/1.1.0/</a>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>To support Kotlin Coroutines, we added new IVS Chat Messaging APIs in the <code>com.amazonaws.ivs.chat.messaging.coroutines</code> package. Also see the new Kotlin Coroutines tutorial; part 1 (of 2) is <a href="#">Chat Rooms</a>.</li> </ul>

## Chat Client Messaging SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
All architectures (bytecode)	89 KB	92 KB

## January 17, 2023

## Amazon IVS Player SDK 1.16.0

Platform	Downloads and Changes
<a href="#">Web player 1.16.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.16.0/amazon-ivs-player.min.js">https://player.live-video.net/1.16.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.16.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.16.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.16.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.16.0/web/</a></p> <ul style="list-style-type: none"> <li>Updated SDK documentation to note which methods are not supported on iOS mobile browsers.</li> </ul>

Platform	Downloads and Changes
<a href="#">Android player 1.16.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.16.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.16.0/android/</a></p> <ul style="list-style-type: none"> <li>Added the <code>setOrigin</code> method to enable inclusion of an <code>Origin</code> request header with playback requests. Also see in <a href="#">Token Schema</a> for the new <code>strict-origin-enforcement</code> field.</li> </ul>
<a href="#">iOS Player 1.16.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.16.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.16.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.16.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.16.0/ios/</a></p> <ul style="list-style-type: none"> <li>Added the <code>setOrigin</code> method to enable inclusion of an <code>Origin</code> request header with playback requests. Also see in <a href="#">Token Schema</a> for the new <code>strict-origin-enforcement</code> field.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.009 MB	2.852 MB
armeabi-v7a	0.914 MB	2.258 MB
x86_64	1.054 MB	2.689 MB
x86	1.080 MB	2.987 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.41 MB	0.99 MB

## December 9, 2022

### Timestamp Added to Auto-Record to S3 Manifest Files

When [Auto-Record to Amazon S3](#) is enabled, HLS manifest files are created. Those files now contain HLS Program-Date-Time (PDT) tags indicating the wall-clock time for every HLS segment when produced, using the UTC ISO-8601 format.

## December 6, 2022

### Amazon IVS Broadcast SDK: Android 1.7.2

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.7.2</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.2/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.2/android/</a></p> <ul style="list-style-type: none"> <li>Fixed a bug where the <code>Device.Descriptor</code> returned by a non-Camera device subclassing <code>SurfaceSource</code> would provide a unique <code>deviceId</code> and <code>urn</code> on each call, making those properties unreliable for identifying devices.</li> <li>Fixed a bug where the preferred <code>AudioInput</code> property on a <code>BroadcastConfiguration.Mixer.Slot</code> was null when queried by <code>Mixer.getSlots()</code>, if the associated slot had a preferred <code>AudioInput</code> value of <code>Device.De</code></li> </ul>



Platform	Downloads and Changes
	<pre>scriptor.DeviceType.MICROPHONE</pre> when it was added.

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.628 MB	4.682 MB
armeabi-v7a	1.519 MB	3.786 MB
x86_64	1.701 MB	5.075 MB
x86	1.637 MB	4.605 MB

## November 17, 2022

### Chat Logging

This is the first release of new functionality. You can now create logging configurations to enable storage of messages sent to your chat rooms. For more information, see:

- [Chat Logging](#) - New page.
- [Getting Started with Chat](#) - Updated IAM permissions and added procedures for setting up chat logging.
- [Service Quotas](#) - new endpoints and logging configurations.
- CloudWatch – Added log-destination metrics.

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

Dec 28, 2023 update: Chat-related CloudWatch content was moved to [Monitoring Amazon IVS Chat](#).

- [Chat API Reference](#) – Added a LoggingConfiguration resource and several data types and endpoints. For details see [Document History](#).

## November 9, 2022

### Amazon IVS Chat Client Messaging SDK: JavaScript 1.0.2

Platform	Downloads and Changes
<a href="#">JavaScript Chat Client Messaging SDK 1.0.2</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-chat-messaging-sdk-js/1.0.2/">https://aws.github.io/amazon-ivs-chat-messaging-sdk-js/1.0.2/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue that affected Firefox: clients erroneously received a socket error when they were disconnected from a chat room using the DisconnectUser endpoint.</li> </ul>

## November 1, 2022

### Amazon IVS Player SDK 1.14.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Improved player stability by reducing occurrences of rare crashes.</li> </ul>
<a href="#">Web player 1.14.0 &amp; Video.js integration &amp; JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.14.0/amazon-ivs-player.min.js">https://player.live-video.net/1.14.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.14.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.14.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.14.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.14.0/web/</a></p>
<a href="#">Android player 1.14.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.14.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.14.0/android/</a></p>

Platform	Downloads and Changes
<a href="#">iOS Player 1.14.0</a>	<ul style="list-style-type: none"> <li>Added the <code>getErrorCode()</code> method to the <code>ErrorType</code> class.</li> </ul> <p><b>Download:</b> <a href="https://player.live-video.net/1.14.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.14.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.14.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.14.0/ios/</a></p> <ul style="list-style-type: none"> <li>Made public the IVS Player <code>setQuality:adaptive:</code> method.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.004 MB	2.840 MB
armeabi-v7a	0.909 MB	2.248 MB
x86_64	1.049 MB	2.678 MB
x86	1.075 MB	2.975 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.41 MB	0.99 MB

## October 18, 2022

### Amazon IVS Chat Client Messaging SDK: JavaScript 1.0.1

Platform	Downloads and Changes
<a href="#">JavaScript Chat Client Messaging SDK 1.0.1</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-chat-messaging-sdk-js/1.0.1/">https://aws.github.io/amazon-ivs-chat-messaging-sdk-js/1.0.1/</a>

## October 6, 2022

### Amazon IVS Broadcast SDK 1.7.1

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.7.1</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.7.1/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.7.1/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.1/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.1/ios/</a></p> <ul style="list-style-type: none"><li>• Fixed a linker error when directly linking against a few classes.</li><li>• Removed <code>init</code> and <code>new</code> functions on classes that should never be instantiated by the host application.</li><li>• Slots using the camera provided by the SDK and configured to a 9:16 portrait aspect ratio now correctly use the matching 9:16 camera ratio. (Previously they used a 3:4 camera ratio.) Slots using the FIT aspect mode now use the entire space. (Previously they were letterboxed.)</li></ul>

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.48 MB	3.40 MB

## September 22, 2022

### Amazon IVS Broadcast SDK 1.7.0

Note: There was no 1.6.0 release.

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Improved stability by reducing occurrences of rare crashes.</li> <li>Added an <code>AutomaticBitrateProfile</code> enum on <code>VideoConfiguration</code>. This controls the rate at which the ABR algorithm adjusts the video bitrate.</li> <li>Added the <code>onTransmissionStatsChanged</code> method. It contains more detailed transmission statistics than <code>onBroadcastQualityChanged</code> and <code>onNetworkHealthChanged</code>. We deprecated the latter two methods and we recommend you use <code>onTransmissionStatsChanged</code> instead.</li> </ul>
<a href="#">Android Broadcast SDK 1.7.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.0/android/</a>
<a href="#">iOS Broadcast SDK 1.7.0</a>	<b>Download:</b> <a href="https://broadcast.live-video.net/1.7.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.7.0/AmazonIVSBroadcast.xcframework.zip</a>

Platform	Downloads and Changes
	<p data-bbox="829 212 1487 296"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.7.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.7.0/ios/</a></p> <ul data-bbox="829 338 1487 611" style="list-style-type: none"> <li>• Added <code>IVSBroadcastSessionAudioSessionStrategy.PlayAndRecordDefaultToSpeaker</code>, which allows developers to specify whether devices with handsets (e.g., iPhones) prefer the speaker over the headset.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.628 MB	4.682 MB
armeabi-v7a	1.519 MB	3.786 MB
x86_64	1.824 MB	5.212 MB
x86	1.760 MB	4.742 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.47 MB	3.40 MB

# September 20, 2022

## Amazon IVS Player SDK 1.13.0

Platform	Downloads and Changes
<a href="#">Web player 1.13.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.13.0/amazon-ivs-player.min.js">https://player.live-video.net/1.13.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.13.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.13.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.13.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.13.0/web/</a></p> <ul style="list-style-type: none"><li>• Added support for the VideoJS seeking() function.</li><li>• Removed unused types (CaptureEventTypes) which caused development issues.</li><li>• Fixed intermittent MediaSource errors on network recovery.</li></ul> <p><b>Known issue:</b> The Sawmill Enabled log may appear when you open the console. This internal log is meant to be hidden, as it does not affect customers. If you see it, ignore it.</p>
<a href="#">Android player 1.13.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.13.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.13.0/android/</a></p> <ul style="list-style-type: none"><li>• Added additional guards to prevent playback crashes related to race conditions.</li><li>• Made stability improvements to ABR bandwidth estimation.</li></ul>

Platform	Downloads and Changes
<a href="#">iOS Player 1.13.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.13.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.13.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.13.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.13.0/ios/</a></p> <ul style="list-style-type: none"> <li>• Fixed a bug where audio-only playback could stop while playing in the background.</li> <li>• Added additional guards to prevent playback crashes related to race conditions.</li> <li>• Made stability improvements to ABR bandwidth estimation.</li> <li>• Clarified in the SDK Reference that <code>setAutoMaxQuality</code> filters qualities based on bitrate.</li> <li>• Changed the <code>setQuality:</code> method of the <code>IVSPlayer</code> class so it ignores invalid values.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.000 MB	2.829 MB
armeabi-v7a	0.904 MB	2.237 MB
x86_64	1.070 MB	2.962 MB
x86	1.045 MB	2.665 MB



## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.44 MB	1.06 MB

## September 15, 2022

### Vertical Video Improvement (Final Release)

Today we began rolling out the changes documented in [Vertical Video Improvement](#) for all Amazon IVS customers. It will take 2-3 days for the changes to propagate across all accounts.

## September 12, 2022

### Amazon IVS Broadcast SDK 1.5.2: iOS

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.5.2</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.5.2/AmazonIVSBroadcast.xcfrawork.zip">https://broadcast.live-video.net/1.5.2/AmazonIVSBroadcast.xcfrawork.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.5.2/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.5.2/ios/</a></p> <ul style="list-style-type: none"><li>• Fixed a rare crash when the network connection is lost very soon after a broadcast is stopped but before the broadcast shutdown has completed.</li><li>• Fixed a memory-growth issue when a retry loop repeatedly tries to restart a broadcast after a fatal error.</li></ul>

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.39 MB	3.20 MB

## September 8, 2022

## Amazon IVS Chat Client Messaging SDK: Android 1.0.0 and iOS 1.0.0

Platform	Downloads and Changes
<a href="#">Android Chat Client Messaging SDK 1.0.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-chat-messaging-sdk-android/1.0.0/">https://aws.github.io/amazon-ivs-chat-messaging-sdk-android/1.0.0/</a>
<a href="#">iOS Chat Client Messaging SDK 1.0.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-chat-messaging-sdk-ios/1.0.0/">https://aws.github.io/amazon-ivs-chat-messaging-sdk-ios/1.0.0/</a>

## Chat Client Messaging SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
All architectures (bytecode)	53 KB	58 KB

## Chat Client Messaging SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
ios-arm64_x86_64-simulator (bitcode)	484 KB	2.4 MB
ios-arm64_x86_64-simulator	484 KB	2.4 MB

Architecture	Compressed Size	Uncompressed Size
ios-arm64 (bitcode)	1.1 MB	3.1 MB
ios-arm64	233 KB	1.2 MB

## September 2, 2022

### Amazon IVS Web Broadcast SDK 1.2.0

Platform	Downloads and Changes
<a href="#">Web Broadcast SDK</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul style="list-style-type: none"> <li>Fixed bundle type definitions when the npm pack is run.</li> <li>Added a preset configuration to support 1080 BASIC channels.</li> </ul>

## August 30, 2022

### Merge Fragmented Streams

This is the first release of new functionality. If your stream is configured for Auto-Record to Amazon S3, you can now specify a window of time during which, if your stream is interrupted and a new stream is started, Amazon IVS tries to record to the same S3 prefix as the previous stream. In other words, if a broadcast disconnects and then reconnects within the specified interval, the multiple streams are considered a single broadcast and merged. For more information, see:

- Getting Started with Amazon IVS – We updated [Step 3: Create a Channel with Optional Recording](#), for console and CLI instructions.
- Auto-Record to S3 – See the new section, [Merge Fragmented Streams](#).

- EventBridge – In [Examples: Recording State Change](#), `recording_session_id` and `recording_session_stream_ids` fields were added.
- [IVS API Reference](#) – We added the `recordingReconnectWindowSeconds` field to the `CreateRecordingConfiguration` request and the `RecordingConfiguration` object. This affects three responses (`CreateRecordingConfiguration`, `GetRecordingConfiguration`, and `GetStreamSession`).

## August 9, 2022

### Amazon IVS Web Player SDK 1.12.0

Platform	Downloads and Changes
<a href="#">Web player 1.12.0 &amp; Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.12.0/amazon-ivs-player.min.js">https://player.live-video.net/1.12.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.12.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.12.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.12.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.12.0/web/</a></p> <ul style="list-style-type: none"> <li>• Added additional guards to prevent playback crashes related to race conditions.</li> </ul>

## July 28, 2022

### Amazon IVS iOS Broadcast SDK 1.5.1

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.5.1</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.5.1/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.5.1/AmazonIVSBroadcast.xcframework.zip</a></p>

Platform	Downloads and Changes
	<p data-bbox="829 212 1487 296"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.5.1/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.5.1/ios/</a></p> <ul data-bbox="829 338 1503 861" style="list-style-type: none"> <li data-bbox="829 338 1503 611">• Fixed compatibility issues with iOS 16 that prevented audio encoding, causing all broadcasts to fail. <i>This issue impacts all previous versions of the IVS Broadcast SDK for iOS. Version 1.5.1 is required to broadcast on iOS 16.</i></li> <li data-bbox="829 632 1503 861">• Fixed a memory leak when providing a delegate directly to the <code>IVSBroadcastSession</code>'s initializer. (A workaround was to set the delegate property afterwards.)</li> </ul>

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.36 MB	3.20 MB

## July 21, 2022

## Amazon IVS Web Broadcast SDK

Platform	Downloads and Changes
<p data-bbox="115 1604 399 1640"><a href="#">Web Broadcast SDK</a></p>	<p data-bbox="829 1604 1487 1734"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference">https://aws.github.io/amazon-ivs-web-broadcast/docs/sdk-reference</a></p> <ul data-bbox="829 1776 1487 1860" style="list-style-type: none"> <li data-bbox="829 1776 1487 1860">• This is the initial release of the Amazon IVS Web Broadcast SDK.</li> </ul>

## July 14, 2022

### Amazon IVS iOS Player SDK 1.8.3

Platform	Downloads and Changes
<a href="#">iOS Player 1.8.3</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.8.3/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.8.3/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.8.3/ios/">https://aws.github.io/amazon-ivs-player-docs/1.8.3/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue where the Player could not play recorded content served via a URL that includes a relative path.</li> <li>Fixed a memory-growth issue that could occur when the main thread is blocked.</li> </ul>

### Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.46 MB	1.10 MB

## June 28, 2022

### Amazon IVS Player Web SDK 1.11.0

Platform	Downloads and Changes
<a href="#">Web player 1.11.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.11.0/amazon-ivs-player.min.js">https://player.live-video.net/1.11.0/amazon-ivs-player.min.js</a></p>

Platform	Downloads and Changes
	<p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.11.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.11.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.11.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.11.0/web/</a></p> <ul style="list-style-type: none"> <li>• Made stability improvements to ABR bandwidth estimation.</li> <li>• Fixed an issue when playing recorded content on an iOS mobile browser using the Video.js integration: the replay button now works. The prior workaround (hiding the replay button when initializing Video.js) is no longer required.</li> </ul>

## June 22, 2022

### Amazon IVS Broadcast SDK 1.5.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>• Improved stability by reducing occurrences of rare crashes.</li> <li>• Improved stability for high-bitrate streams.</li> <li>• Broadcasts experiencing extremely high latency will be ended with error code 20401 and this message: "The broadcast has ended because the network got too far behind. Check that you have a stable connection or reduce the broadcast bitrate." The threshold latency value for this is likely to change over time; currently it is 45 seconds.</li> </ul>

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.5.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.5.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.5.0/android/</a></p> <ul style="list-style-type: none"><li>• Added a new configuration option to Video that enables transparency for the broadcast session: <code>enableTransparency(boolean)</code> and <code>isTransparencyEnabled()</code>. By default, transparency is disabled. Note that you must set <code>Video.enableTransparency</code> to <code>TRUE</code> for individual slot's <code>fillColor</code> or <code>transparency</code> values to work as expected. Enable transparency only when required, since it is more computationally intensive.</li></ul>



Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.5.0</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.5.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.5.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.5.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.5.0/ios/</a></p> <ul style="list-style-type: none"> <li>When using <code>IVSReplayKitBroadcastSession</code> for screen share, we recommend that you call <code>IVSReplayKitBroadcastSession::broadcastFinished</code> in <code>RPBroadcastSampleHandler::broadcastFinished</code> to ensure proper shutdown of the stream. Failure to do this might result in the stream staying live until it times out.</li> <li><code>IVSImagePreviewView</code> is no longer backed by <code>MTKView</code>, but instead a normal <code>UIView</code> that has a <code>AVSampleBufferDisplayLayer</code> based <code>CALayer</code>.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.539 MB	4.355 MB
armeabi-v7a	1.431 MB	3.483 MB
x86_64	1.729 MB	4.868 MB
x86	1.675 MB	4.436 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.36 MB	3.20 MB

## June 9, 2022

### Vertical Video Improvement

This release improves how Amazon IVS processes *vertical input*; e.g., content broadcast from mobile devices where the height is greater than the width. This change is being rolled out over time, as explained at the end of this post.

There are three changes:

1. (Affects Standard channels only) Vertical input is scaled based on the content's width, resulting in less downscaling and visually higher quality output. For example, here is how this change impacts 720x1280 input:

Name	Old Width x Height	New Width x Height
1280p	720 x 1280	720 x 1280
720p	404 x 720	720 x 1280
480p	268 x 480	480 x 852
360p	200 x 360	360 x 640
160p	88 x 160	160 x 284

2. (Affects Standard channels only) The only renditions that are generated are those with width less than or equal to your input width. For example, if your input is 720x1280, you get 720p, 480p, 360p, and 160p renditions. If your input width is between renditions, you get all renditions with lower widths than your input. For example, here is how this change impacts 540x960 input:

Name	Old Width x Height	New Width x Height
960p	540 x 960	â€”
720p	404 x 720	â€”
480p	268 x 480	480 x 852
360p	200 x 360	360 x 640
160p	88 x 160	160 x 284

3. (Affects Standard and Basic channels) Renditions for vertical input use a more conventional naming scheme based on width instead of height. For example, 360x640 input to a Basic channel has one output rendition named 360p.

This name appears in video playlists as the NAME attribute and in the user-facing quality selector ([example](#)). The name also is used as the Amazon S3 directory name for recorded assets. For example, for 360x640 input, the quality selector and Auto-Record to Amazon S3 directory name is 360p60 (the old value was 640p60).

We are rolling out this improvement over time:

- Now – Did you broadcast vertical input in the past six months? If not, we are enabling this change for your account now (specifically, over a 1-week period starting today). If yes, you will get a notification about this change in your account events section of the AWS Health Dashboard.
- September 15, 2022 – We will enable the change on all remaining accounts. If you broadcast vertical input in the past six months and want this change to be enabled on your account sooner, please submit an AWS support ticket.

**Important:** Make sure you do not have any code (e.g., post-processing of recordings) that depends on the old behavior. For instance, if you have a script with rendition width/height hardcoded, you must edit that or it may break after this change is applied.

# May 24, 2022

## Amazon IVS Web and Android Player SDK 1.10.0

Platform	Downloads and Changes
<a href="#">Web player 1.10.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.10.0/amazon-ivs-player.min.js">https://player.live-video.net/1.10.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.10.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.10.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.10.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.10.0/web/</a></p> <ul style="list-style-type: none"><li>• Fixed console errors related to pausing and playing with the Video.js plugin.</li><li>• In the reference documentation, removed from the TypeScript definitions file two types which should not have been exposed, <code>AutoplayOptions</code> and <code>PlayerEventType.STATE_CHANGED</code>.</li><li>• Fixed an issue where not all qualities were considered when using <code>setAutoMaxQuality</code> and <code>setAutoMaxVideoSize</code>.</li><li>• Exposed the <code>setAutoMaxVideoSize</code> method, with corresponding documentation.</li><li>• Clarified in the SDK Reference that <code>setAutoMaxQuality</code> filters qualities based on bitrate.</li></ul>

Platform	Downloads and Changes
<a href="#">Android player 1.10.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.10.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.10.0/android/</a></p> <ul style="list-style-type: none"> <li>• Corrected the end-of-stream behavior for VODs for web platforms.</li> <li>• Fixed an issue where not all qualities were considered when using <code>setAutoMaxQuality</code> and <code>setAutoMaxVideoSize</code>.</li> <li>• Added <code>getVolume()</code> to the <code>Player</code> class.</li> <li>• Clarified in the SDK Reference that <code>setAutoMaxQuality</code> filters qualities based on bitrate.</li> <li>• Corrected the end-of-stream behavior for VODs for web platforms.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	0.990 MB	2.805 MB
armeabi-v7a	0.895 MB	2.215 MB
x86_64	1.033 MB	2.643 MB
x86	1.058 MB	2.936 MB

## April 28, 2022

### Stream Health Updates

There are two updates to Amazon IVS Stream Health, for monitoring the health of your live streams in near real time:

- Charts of the high-resolution CloudWatch metrics are now available in the stream session details pages on the console.
- A new dimension (Health) was added to the ConcurrentStreams metric, to filter the results by channel health.

See [Monitoring Amazon IVS Live Stream Health](#) and [Monitoring Amazon IVS with Amazon CloudWatch](#).

Oct 12, 2023 update: These documents were combined into [Monitoring IVS Low-Latency Streaming](#).

## April 26, 2022

### Amazon IVS Chat

This is the initial release of Amazon IVS Chat, a managed, live-chat feature to go alongside live video streams. New documentation is accessible from the [Amazon IVS documentation landing page](#).

- Start with [Getting Started with Amazon IVS Chat](#).
- In the *Amazon IVS Chat User Guide*:
  - See [Chat Message Review Handler](#), a new page.
  - Search for "chat" changes in [Monitoring Amazon IVS with Amazon CloudWatch](#), [Amazon IVS Security](#), and [Amazon IVS Service Quotas](#).

Oct 12, 2023 update: The CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

Dec 28, 2023 update: All chat information was collected in a new [Chat User Guide](#).

- The new **Amazon IVS Chat** section of the documentation landing page has two API References:

- [Chat API Reference](#) – Control-plane API (HTTPS).
- [Chat Messaging API Reference](#) – Data-plane API (WebSocket).

As always, documentation changes are described in the Amazon IVS [Document History](#).

## April 22, 2022

### Amazon IVS iOS Player SDK 1.8.2

Platform	Downloads and Changes
<a href="#">iOS Player 1.8.2</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.8.2/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.8.2/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.8.2/ios/">https://aws.github.io/amazon-ivs-player-docs/1.8.2/ios/</a></p> <ul style="list-style-type: none"><li>• Added support for Picture in Picture on devices running iOS 15 and later. You can instantiate the <a href="#">AVPictureInPictureController</a> class directly with an instance of <code>IVSPlayerLayer</code>. Refer to the <a href="#">public sample app</a> for an example implementation.</li><li>• Fixed a deadlock issue that can occur while manipulating the <code>IVSPlayer</code> state from inside the completion handler of the <code>-seekTo:completionHandler:</code> method.</li><li>• Fixed an issue introduced by the 1.8.1 release in an attempt to resolve a memory growth issue that can occur when the main thread is blocked.</li></ul>

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.46 MB	1.10 MB

## April 19, 2022

## Amazon IVS Broadcast SDK 1.4.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Improved stability by reducing occurrences of rare crashes.</li> <li>Added a new page on <a href="#">Broadcast SDK: Custom Image Sources</a>.</li> </ul>
<a href="#">Android Broadcast SDK 1.4.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.4.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.4.0/android/</a></p> <ul style="list-style-type: none"> <li>Fixed a bug in <code>createServiceNotificationBuilder</code> to enable targeting Android 12.</li> <li>Fix issue on devices with a buggy main AVC profile by falling back to the baseline AVC profile.</li> <li>Adds some NonNull annotations to several public API method signatures to prevent unexpected exceptions from crashing the application.</li> </ul>
<a href="#">iOS Broadcast SDK 1.4.0</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.4.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.4.0/AmazonIVSBroadcast.xcframework.zip</a></p>



Platform	Downloads and Changes
	<p data-bbox="829 212 1487 296"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.4.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.4.0/ios/</a></p> <ul data-bbox="829 338 1487 919" style="list-style-type: none"> <li data-bbox="829 338 1487 520">• Improved performance on iOS throughout the entire SDK by better utilizing GCD and Darwin-optimized locks, and improving buffer reuse.</li> <li data-bbox="829 537 1487 667">• In BroadcastConfiguration, changed the Keyframe interval maximum value from 10 to 5 to be consistent with Android.</li> <li data-bbox="829 684 1487 919">• Added a new method to control the audio encoder quality. On IVSAudioC onfiguration , use the setQuality method. Reducing the encoder quality can have a large impact on CPU usage.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.526 MB	4.324 MB
armeabi-v7a	1.416 MB	3.442 MB
x86_64	1.657 MB	4.393 MB
x86	1.712 MB	4.827 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.33 MB	3.13 MB

# March 31, 2022

## Amazon IVS iOS Player SDK 1.8.1

Platform	Downloads and Changes
<a href="#">iOS Player 1.8.1</a>	<p><b>Download:</b> &lt;deprecated&gt;</p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.8.1/ios/">https://aws.github.io/amazon-ivs-player-docs/1.8.1/ios/</a></p> <ul style="list-style-type: none"> <li>Added support for Picture in Picture on devices running iOS 15 and later. You can instantiate the <a href="#">AVPictureInPictureController</a> class directly with an instance of <code>IVSPlayerLayer</code>. Refer to the <a href="#">public sample app</a> for an example implementation.</li> <li>Fixed a memory-growth issue that can occur when the main thread is blocked.</li> <li>Fixed a deadlock issue that can occur while manipulating the <code>IVSPlayer</code> state from inside the completion handler of the <code>-seekTo:completionHandler:</code> method.</li> </ul>

### Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.46 MB	1.10 MB

# March 3, 2022

## Amazon IVS Broadcast SDK 1.3.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"><li>• Improved stability by reducing occurrences of rare crashes.</li><li>• Added support for 32-bit signed integer and 64-bit floating point PCM audio.</li></ul>
<a href="#">Android Broadcast SDK 1.3.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.3.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.3.0/android/</a></p> <ul style="list-style-type: none"><li>• Fixed an intermittent issue where disconnecting a Bluetooth headset while streaming would lead to a crash.</li><li>• The <code>BroadcastSession.onBroadcastQuality</code> method now reports low initial broadcast-quality values.</li><li>• Added support for PCM buffers that include multiple <code>AudioBufferLists</code>. This is common for USB microphones.</li></ul> <p>Incorporates changes from the Android 1.2.1 release: new methods and a bug fix to properly support surface size and rotation changes:</p> <ul style="list-style-type: none"><li>• Fixed a bug where <code>SurfaceSource.setSize(...)</code> did not set a new size for the <code>SurfaceSource</code>.</li><li>• Added the <code>Device.setRotation(float rotation)</code> method to set the rotation on a device in radians.</li></ul>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>Added the <code>ImageDevice.rotateOnConfigurationChanges(boolean enable)</code> method to enable/disable automatic rotation of the <code>ImageDevice</code> when the physical handset is rotated.</li> <li>Added the <code>ImageDevice.willRotateOnConfigurationChanges()</code> method to return whether the <code>ImageDevice</code> is configured to automatically rotate when the physical handset rotates.</li> </ul>
<a href="#">iOS Broadcast SDK 1.3.0</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.3.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.3.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.3.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.3.0/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed some race conditions when using the <code>createAppBackgroundImageSource</code> method, which could have prevented the stream from resuming after the app returns to the foreground.</li> <li>Added support for the arm64 simulator.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.531 MB	4.411 MB
armeabi-v7a	1.420 MB	3.525 MB
x86_64	1.719 MB	4.877 MB
x86	1.659 MB	4.925 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.30 MB	3.06 MB

## March 1, 2022

## Amazon IVS Player SDK 1.8.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Reduced occurrences of freezing during quality switches when playing recorded content.</li> </ul>
<a href="#">Web player 1.8.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.8.0/amazon-ivs-player.min.js">https://player.live-video.net/1.8.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.8.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.8.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.8.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.8.0/web/</a></p> <ul style="list-style-type: none"> <li>Fixed an edge case where playback of recorded content could stall in some browsers.</li> <li>Fixed an issue where timed metadata events were not triggered after seeking forward and then backward on recorded video.</li> <li>Removed unnecessary, confusing warnings for the JW Player integration on <code>remove()</code>.</li> </ul>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>Enabled stricter type checking for cue types to support correct cue-type filtering.</li> </ul>
<a href="#">Android player 1.8.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.8.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.8.0/android/</a></p> <ul style="list-style-type: none"> <li>Removed the <code>ViewUtil</code> class, which is internal and was deprecated. Use <code>PlayerView</code> instead.</li> </ul>
<a href="#">iOS Player 1.8.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.8.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.8.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.8.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.8.0/ios/</a></p>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	0.975 MB	2.761 MB
armeabi-v7a	0.882 MB	2.177 MB
x86_64	1.020 MB	2.603 MB
x86	1.043 MB	2.890 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.46 MB	1.10 MB

# February 3, 2022

## Amazon IVS Broadcast SDK: Android 1.2.1

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.2.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.2.1/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.2.1/android/</a></p> <p>This release has new methods and a bug fix to properly support surface size and rotation changes. This is needed for use cases involving custom video input.</p> <ul style="list-style-type: none"><li>• Fixed a bug where <code>SurfaceSource.setSize(...)</code> did not set a new size for the <code>SurfaceSource</code>.</li><li>• Added the <code>Device.setRotation(float rotation)</code> method to set the rotation on a device in radians.</li><li>• Added the <code>ImageDevice.rotateOnConfigurationChanges(boolean enable)</code> method to enable/disable automatic rotation of the <code>ImageDevice</code> when the physical handset is rotated.</li><li>• Added the <code>ImageDevice.willRotateOnConfigurationChanges()</code> method to return whether the <code>ImageDevice</code> is configured to automatically rotate when the physical handset rotates.</li></ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.642 MB	4.536 MB
armeabi-v7a	1.468 MB	3.261 MB
x86_64	1.866 MB	5.225 MB
x86	1.809 MB	4.916 MB

## January 20, 2022

### Amazon IVS Player SDK 1.7.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Fixed stuttering when playing a stream from a source media playlist.</li> </ul>
<a href="#">Web player 1.7.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.7.0/amazon-ivs-player.min.js">https://player.live-video.net/1.7.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.7.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.7.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.7.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.7.0/web/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue where timed metadata events were not triggered after replaying an Amazon IVS recorded video.</li> <li>Fixed an issue where the ErrorNotAvailable error was not emitted when a streamâ€™s</li> </ul>



Platform	Downloads and Changes
	<p>playback URL is unavailable on iOS mobile web browsers.</p> <ul style="list-style-type: none"><li>• Fixed a console warning when calling <code>dispose()</code> using the Video.js wrapper.</li><li>• Fixed several null reference errors caused by attempting to access the player instance after it is destroyed.</li><li>• Updated <code>setQuality</code> documentation to more clearly specify that one should listen to the <code>QUALITY_CHANGED</code> to be notified of success.</li><li>• Updated <code>setInitialBufferDuration()</code> documentation to specify that it does not function on iOS mobile browsers.</li></ul> <p><b>Known issue:</b> When a viewer skips forward in recorded content, then skips backward, timed metadata within iOS browsers is not re-fired until after the skip-forward time. For example, if a viewer begins watching recorded content, skips forward to 60 seconds, then skips backward to 30 seconds, no timed metadata is triggered between 30 and 60 seconds. We expect to fix this issue in an upcoming release.</p>

Platform	Downloads and Changes
<a href="#">Android player 1.7.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.7.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.7.0/android/</a></p> <ul style="list-style-type: none"> <li>• Deprecated the <code>ViewUtil</code> class, which is internal; use <code>PlayerView</code> instead. This class will be removed completely in the next Amazon IVS Player release (1.8.0, tentatively planned for 2022Q1).</li> <li>• Added <code>PlayerView.setResizeMode(mode)</code> to control how the video is displayed in the view, allowing the video to be optionally zoomed in or to fill the view entirely ignoring the video aspect ratio.</li> </ul>
<a href="#">iOS Player 1.7.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.7.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.7.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.7.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.7.0/ios/</a></p> <ul style="list-style-type: none"> <li>• The iOS SDK now requires iOS 11.0 or higher.</li> <li>• The SDK no longer contains an arm64e slice. It will be re-enabled once Apple makes this a standard architecture.</li> <li>• Fixed rare crashes that could occur during app termination and media-service reset event.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.013 MB	2.820 MB

Architecture	Compressed Size	Uncompressed Size
armeabi-v7a	0.895 MB	2.012 MB
x86_64	1.119 MB	3.099 MB
x86	1.125 MB	2.970 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	0.46 MB	1.09 MB

## January 18, 2022

### R2S3 Thumbnail Configuration

This release allows you to enable/disable the recording of thumbnails for a live session and modify the interval at which thumbnails are generated for the live session. This is the first release of this new functionality. See:

- [Getting Started with Amazon IVS](#) – We updated "Step 3: Create a Channel with Optional Recording."
- [Auto-Record to Amazon S3](#) – We made several changes:
  - We added a note to "Recording Contents" about modifying the thumbnails folder.
  - We added a new "Thumbnails" section.
  - We changed the information about the thumbnails and path fields in "JSON Metadata Files."
- [Amazon IVS API Reference](#) – We made several changes:
  - New field (`thumbnailConfiguration`) in the `RecordingConfiguration` object. This in turn affects the `CreateRecordingConfiguration` request and response, `GetRecordingConfiguration` response, and `GetStreamSession` response.
  - New object: `ThumbnailConfiguration`.

# December 9, 2021

## Amazon IVS Broadcast SDK 1.2.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Improved stability by reducing occurrences of rare crashes.</li> <li>Added a new method, <code>onNetworkHealthChanged</code> (Android) and <code>broadcastSession:networkHealthChanged</code> (iOS). This provides updates when the instantaneous quality of the network changes. It can be used to provide feedback about when the broadcast might have temporary disruptions.</li> <li>Added methods to get/set <code>BroadcastConfiguration.mixer.canvasAspectMode</code>. This is used as the default aspect mode for slots when the slot's aspect mode is not set explicitly.</li> <li>Changed the <code>Mixer</code> (Android) and <code>IVSBroadcastMixer</code> (iOS) APIs:             <ul style="list-style-type: none"> <li>Added <code>getSlots()</code> which returns all added slots.</li> <li>Added <code>unbind</code>, which unbinds a device from a mixer slot.</li> <li>Updated <code>bind</code>, <code>unbind</code>, and <code>transition</code> to return a <code>bool</code> indicating success or failure.</li> </ul> </li> </ul>
<a href="#">Android Broadcast SDK 1.2.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.2.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.2.0/android/</a>

Platform	Downloads and Changes
	<ul style="list-style-type: none"> <li>Fixed a bug where, even if transparency was enabled, a slot's video or image was not blended with other slots beneath it (using zIndex values).</li> </ul>
<a href="#">iOS Broadcast SDK 1.2.0</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.2.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.2.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.2.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.2.0/ios/</a></p> <ul style="list-style-type: none"> <li>Improved the connection stability of Bluetooth and wired headsets.</li> <li>Added support to <code>IVSCustomImageSource</code> for the following pixel formats: <ul style="list-style-type: none"> <li><code>kCVPixelFormatType_Lossless_420YpCbCr8BiPlanarFullRange</code></li> <li><code>kCVPixelFormatType_Lossy_420YpCbCr8BiPlanarFullRange</code></li> <li><code>kCVPixelFormatType_Lossless_420YpCbCr8BiPlanarVideoRange</code></li> <li><code>kCVPixelFormatType_Lossy_420YpCbCr8BiPlanarVideoRange</code></li> <li><code>kCVPixelFormatType_Lossless_32BGRA</code></li> <li><code>kCVPixelFormatType_Lossy_32BGRA</code></li> </ul> </li> <li>Fixed two race conditions when using the <code>createAppBackgroundImageSource</code> method, which could have prevented the stream from resuming after the app returns to the foreground.</li> </ul>

## Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.639 MB	4.530 MB
armeabi-v7a	1.466 MB	3.255 MB
x86_64	1.863 MB	5.219 MB
x86	1.806 MB	4.910 MB

## Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	1.42 MB	3.30 MB

## November 23, 2021

### Amazon IVS Player SDK 1.6

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Added a new player function, <code>setInitialBufferDuration()</code>, which allows customers to set the initial buffer duration. This duration determines when playback can start. The allowable range is 0.1 to 5 seconds. This method has no effect on iOS browser platforms.</li> <li>Fixed a bug where a loaded stream could play without the <code>play</code> method being called, during a network reconnect.</li> </ul>

Platform	Downloads and Changes
<p data-bbox="115 443 764 527"><a href="#">Web player 1.6.1 &amp; Video.js integration</a> &amp; <a href="#">JW player integration</a></p>	<p data-bbox="829 212 1455 394"><b>Downloads and Changes</b></p> <ul data-bbox="829 212 1455 394" style="list-style-type: none"><li data-bbox="829 212 1455 289">• Fixed an issue where stale closed caption data was not cleared.</li><li data-bbox="829 310 1455 394">• Improved player stability by reducing occurrences of rare crashes.</li></ul> <p data-bbox="829 443 1471 527"><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p data-bbox="829 569 1495 653"><b>Script asset:</b> <a href="https://player.live-video.net/1.6.1/amazon-ivs-player.min.js">https://player.live-video.net/1.6.1/amazon-ivs-player.min.js</a></p> <p data-bbox="829 695 1484 779"><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.6.1/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.6.1/amazon-ivs-videojs-tech.min.js</a></p> <p data-bbox="829 821 1487 905"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.6.1/web/">https://aws.github.io/amazon-ivs-player-docs/1.6.1/web/</a></p> <ul data-bbox="829 947 1500 1339" style="list-style-type: none"><li data-bbox="829 947 1500 1087">• Added a note to <code>setQuality</code> documentation about how the video element's controls attribute impacts invocation.</li><li data-bbox="829 1108 1500 1192">• Improved how the player recovers from video decode and playlist network errors.</li><li data-bbox="829 1213 1500 1339">• Changed the default log level for the player from <i>warning</i> to <i>error</i>, to match other platforms.</li></ul>
<p data-bbox="115 1388 407 1430"><a href="#">Android player 1.6.0</a></p>	<p data-bbox="829 1388 1487 1472"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.6.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.6.0/android/</a></p>

Platform	Downloads and Changes
<a href="#">iOS Player 1.6.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.6.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.6.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.6.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.6.0/ios/</a></p> <ul style="list-style-type: none"> <li>iOS 10 support will be deprecated starting with the next IVS Player release (1.7.0, tentatively planned for 2022Q1).</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.01 MB	2.82 MB
armeabi-v7a	0.84 MB	2.16 MB
x86_64	1.13 MB	2.97 MB
x86	1.12 MB	3.09 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
universal	0.94 MB	2.02 MB
arm64	0.47 MB	1.11 MB
armv7	0.46 MB	0.89 MB



# November 18, 2021

## Stream Health

Amazon IVS Stream Health lets you monitor the health of your live streams in near real time thanks to new high resolution CloudWatch metrics. You also can gain insights into your stream's events and input configuration through two new API endpoints. This is the first release of this new functionality. See:

- [Monitoring Amazon IVS Live Stream Health](#) – This is a new *Amazon IVS User Guide* page.
- [Getting Started with Amazon IVS](#) – We updated the IAM policy in "Step 2: Set up IAM Permissions" with three more IVS permissions (`GetStream`, `GetStreamSession`, `ListStreamSessions`) and `cloudwatch:GetMetricData`.
- [Monitoring Amazon IVS with Amazon CloudWatch](#) – We added four new, high-resolution metrics (`IngestAudioBitrate`, `IngestFramerate`, `IngestVideoBitrate`, and `KeyframeInterval`).

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

- [Using Amazon EventBridge with Amazon IVS](#) – We added two events, `Session Created` and `Session Ended`.
- [Amazon IVS API Reference](#) – Many changes:
  - Two new endpoints: `GetStreamSession` and `ListStreamSessions`.
  - Seven new objects: `AudioConfiguration`, `IngestConfiguration`, `StreamEvent`, `StreamFilters`, `StreamSession`, `StreamSessionSummary`, and `VideoConfiguration`.
  - New field (`streamID`) in the `Stream` and `StreamSummary` objects. This in turn affects the `GetStream` and `ListStreams` responses.
  - New field (`filtersBy`) in the `ListStreams` request.

# October 20, 2021

## Amazon IVS Broadcast SDK 1.1.0: Android and iOS

Platform	Downloads and Changes
All	<ul style="list-style-type: none"><li>• Fixed a bug that could leave a mixer-slot configuration in an unexpected state when the slot configuration provided to the transition method had a name that did not match the target-slot name parameter.</li><li>• Improved stability by reducing occurrences of rare crashes.</li><li>• Rebalanced preset bitrates to better reflect the expected user experience. These are documented in the Broadcast SDK reference documentation.<ul style="list-style-type: none"><li>• Standard (portrait/landscape) – Initial: 2.1 Mbps. Maximum: 6 Mbps.</li><li>• Basic (portrait/landscape) – Initial: 1.2 Mbps. Maximum: 1.5 Mbps.</li><li>• Gaming (portrait/landscape) (Android only) – Initial: 2.1 Mbps. Maximum 6 Mbps.</li></ul></li><li>• Added support for mono audio. A broadcast session can now be configured with 1 or 2 audio channels (mono or stereo, respectively). Also, custom audio sources can be configured with 1 or 2 audio channels.</li><li>• Changed the Mixer canvas and slot origins to be top-left. This should be more natural for developers and provide more consistent usability. If you are using custom Mixer slots, you must update your code; see</li></ul>

Platform	Downloads and Changes
	<p data-bbox="862 212 1479 296"><a href="#">Broadcast SDK Mixer: Migrating from 1.0.0 to 1.1.0</a> below.</p> <ul data-bbox="829 317 1471 401" style="list-style-type: none"><li data-bbox="829 317 1471 401">• Added a new documentation page, <a href="#">Broadcasting to Amazon IVS: Mixer Guide</a>.</li></ul>
<p data-bbox="115 443 529 485"><a href="#">Android Broadcast SDK 1.1.0</a></p>	<p data-bbox="829 443 1487 579"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.1.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.1.0/android/</a></p> <ul data-bbox="829 621 1503 1451" style="list-style-type: none"><li data-bbox="829 621 1503 705">• Fixed a bug where device-orientation changes could crash the SDK.</li><li data-bbox="829 726 1503 1146">• Fixed a bug where <code>getPreviewView()</code> worked only the first time it was called. Now <code>getPreviewView()</code> returns a new <code>ImagePreviewView</code> every time it is called, so you can add multiple <code>ImagePreviewViews</code> of the same device or session to your view hierarchy at the same time. Note that using many <code>ImagePreviewViews</code> simultaneously can degrade performance.</li><li data-bbox="829 1167 1503 1293">• Added <code>stopSystemCapture()</code> to stop the system-capture service without releasing the entire broadcast session.</li><li data-bbox="829 1314 1503 1451">• Added an <code>attachDevice</code> override, to ignore mixer-slot preferred devices when attaching a device.</li></ul>

Platform	Downloads and Changes
<a href="#">iOS Broadcast SDK 1.1.0</a>	<p><b>Download:</b> <a href="https://broadcast.live-video.net/1.1.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.1.0/AmazonIVSBroadcast.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.1.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.1.0/ios/</a></p> <ul style="list-style-type: none"><li>• Setting size on an <code>IVSMixerSlotConfiguration</code> object now sets <code>matchCanvasSize</code> to false. Similarly, setting aspect on an <code>IVSMixerSlotConfiguration</code> object sets <code>matchCanvasAspectMode</code> to false.</li><li>• Added support for background audio with pre-encoded video. A new method, <code>createAppBackgroundImageSourceOnComplete</code>, changes the default behavior when backgrounding an app. Previously, the entire stream stopped because the SDK no longer had access to the camera or the GPU (which means no video input compositing or video encoding could be done).</li></ul> <p>The new method returns a subclass of <code>IVSCustomVideoSource</code>. Normally, <code>IVSCustomVideoSource</code> allows you to submit image samples to be broadcast. The subclass allows you to submit image samples to be pre-encoded for broadcast later, when your app is in the background.</p>

## Broadcast SDK Mixer: Migrating from 1.0.0 to 1.1.0

Version 1.1.0 of the Broadcast SDK changes how the mixer coordinate system works. In 1.0.0, the mixer used inconsistent origin points. In 1.1.0, the origin is the top-left corner. See the new [Amazon IVS Broadcast SDK Mixer Guide](#).

**Canvas Changes:** Horizontal (X-axis) positions are unchanged. Vertical positioning is inverted, compared to 1.0.0. A Y-axis value of 0 places the slot at the top of the canvas (rather than the bottom, as with 1.0.0). To keep a slot at the same position as in 1.0.0, subtract its current Y value from the height of the canvas; e.g., `config.video.size.height - y`

**Slot Changes:** Slots also have a top-left origin in 1.1.0. The orientation is unchanged from 1.0.0, but the origin has shifted from the center to the top-left. A slot aligned with the top left will be (0, 0), a slot aligned with the bottom right is:

`(canvas_width - slot_width, canvas_height - slot_height)`

To keep a slot in the same position as 1.0.0, subtract half its width from the X position and half its height from the Y position. Also, the slot's size is relative to the top-left corner. Therefore, to expand a slot from the center, you must change the position at the same time as the size; otherwise, the slot will appear to grow down and to the right.

### Broadcast SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	2.23 MB	5.75 MB
armeabi-v7a	2.07 MB	4.99 MB
x86_64	2.35 MB	5.78 MB
x86	2.55 MB	6.78 MB

### Broadcast SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
arm64	3.11 MB	6.74 MB

## September 29, 2021

### Amazon IVS Player SDK: Android 1.5.1

Platform	Downloads and Changes
<a href="#">Android player 1.5.1</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.5.1/android/">https://aws.github.io/amazon-ivs-player-docs/1.5.1/android/</a></p> <ul style="list-style-type: none"> <li>Fixed <code>getVersion()</code> , which now returns the correct version number.</li> </ul>

### Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.00 MB	2.80 MB
armeabi-v7a	0.83 MB	2.15 MB
x86_64	1.11 MB	3.07 MB
x86	1.12 MB	2.94 MB

## September 28, 2021

### Amazon IVS Player SDK 1.5.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Fixed an issue where a loaded stream could play without the play method being called during a network reconnect.</li> <li>Fixed an issue where the player stayed in the PLAYING state after a stream disconnect, instead of moving to the ENDED state.</li> </ul>

Platform	Downloads and Changes
	<ul style="list-style-type: none"><li>• Updated CEA-608 captions parsing to support more encoders.</li><li>• Improved the player's ability to play pass-through content; i.e., content from BASIC channels and the highest quality from STANDARD channels.</li></ul>
<a href="#">Web player 1.5.0 &amp; Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.5.0/amazon-ivs-player.min.js">https://player.live-video.net/1.5.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.5.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.5.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.5.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.5.0/web/</a></p> <ul style="list-style-type: none"><li>• Improved how the player recovers from video-decode and playlist-network errors.</li><li>• Fixed a bug where live streams did not resume (or resumed after a delay) when native HTML5 controls are enabled.</li><li>• Fixed an issue where the <code>getBuffered()</code> method returned <code>undefined</code> instead of the expected <code>{ start: 0, end: 0 }</code> when no content is loaded.</li><li>• Added support for picture-in-picture mode in Video.js.</li><li>• Changed the default log level for the player to error instead of warning.</li></ul>

Platform	Downloads and Changes
<a href="#">Android player 1.5.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.5.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.5.0/android/</a></p> <ul style="list-style-type: none"> <li>• Fixed zoom-in bug that happens on the Android SDK 30 emulator.</li> <li>• Improved the performance of PlayerView view layouts.</li> <li>• getVersion() returns 1.5.0-ivs.rc.2 instead of 1.5.0.</li> </ul>
<a href="#">iOS Player 1.5.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.5.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.5.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.5.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.5.0/ios/</a></p> <ul style="list-style-type: none"> <li>• Added support for the iOS Simulator on Apple Silicon Macs.</li> <li>• Fixed an issue in which the memory heap size of the player continues to increase during playback until the player is deallocated.</li> <li>• Improved playback behavior when there is bad data in the video by ignoring it and continuing playback rather than stopping playback.</li> </ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.00 MB	2.80 MB
armeabi-v7a	0.83 MB	2.15 MB



Architecture	Compressed Size	Uncompressed Size
x86_64	1.11 MB	3.07 MB
x86	1.12 MB	2.94 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
universal	0.92 MB	1.99 MB
arm64	0.47 MB	1.09 MB
armv7	0.46 MB	0.87 MB

## September 8, 2021

### Amazon IVS Player SDK 1.4.1

Platform	Downloads and Changes
All	Fixed the closed-caption decoder to handle captions that are inserted out of order.
<a href="#">Web player 1.4.1</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.4.1/amazon-ivs-player.min.js">https://player.live-video.net/1.4.1/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.4.1/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.4.1/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.4.1/web/">https://aws.github.io/amazon-ivs-player-docs/1.4.1/web/</a></p>

Platform	Downloads and Changes
<a href="#">Android player 1.4.1</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.4.1/android/">https://aws.github.io/amazon-ivs-player-docs/1.4.1/android/</a>
<a href="#">iOS Player 1.4.1</a>	<b>Download:</b> <a href="https://player.live-video.net/1.4.1/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.4.1/AmazonIVSPlayer.xcframework.zip</a> <b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.4.1/ios/">https://aws.github.io/amazon-ivs-player-docs/1.4.1/ios/</a>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.00 MB	2.79 MB
armeabi-v7a	0.83 MB	2.15 MB
x86_64	1.11 MB	3.06 MB
x86	1.11 MB	2.94 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
universal	0.89 MB	1.91 MB
arm64	0.45 MB	1.05 MB
armv7	0.44 MB	0.84 MB

## August 13, 2021

### ListTagsForResource API Endpoint

We removed support for pagination in this endpoint; i.e., the `maxResults` request field and `nextToken` request/response field. (Pagination did not work correctly.)

## August 10, 2021

### Amazon IVS Player SDK 1.4.0

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>• Fixed a rare issue in which VOD playback could stall if a seek happens right after a <code>DURATION_CHANGED</code> event or <code>READY</code> state update.</li> <li>• Fixed a memory leak when playing streams with ID3 metadata.</li> <li>• Fixed an edge case in which injected captions could be rendered incorrectly.</li> <li>• Improved the performance of the player's adaptive bitrate streaming algorithm.</li> <li>• Improved player stability by reducing occurrences of rare crashes.</li> <li>• Added a log warning message when the player is accessed from a different thread than it was created on.</li> <li>• Updated <code>getLiveLatency()</code> documentation to be more specific about how latency is calculated, from the server to the player.</li> </ul>
<a href="#">Web player 1.4.0</a> & <a href="#">Video.js integration</a> & <a href="#">JW player integration</a>	<b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a>

Platform	Downloads and Changes
	<p data-bbox="829 212 1495 296"><b>Script asset:</b> <a href="https://player.live-video.net/1.4.0/amazon-ivs-player.min.js">https://player.live-video.net/1.4.0/amazon-ivs-player.min.js</a></p> <p data-bbox="829 338 1484 422"><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.4.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.4.0/amazon-ivs-videojs-tech.min.js</a></p> <p data-bbox="829 464 1487 548"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.4.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.4.0/web/</a></p> <ul data-bbox="829 590 1511 1789" style="list-style-type: none"><li>• Fixed an edge case in which the <code>TIMED_METADATA</code> event did not fire on iOS Safari.</li><li>• Improved performance of the player's adaptive bitrate streaming algorithm when playing low-latency streams on Firefox.</li><li>• Fixed documentation for <code>getDuration()</code>, which always returns Infinity for live streams.</li><li>• Fixed a bug in which autoplaying on desktop Safari sometimes failed.</li><li>• Fixed an error in which "Cannot read property 'collectLogs' of undefined" is reported in the developer console.</li><li>• Video.js: Added support for picture-in-picture mode.</li><li>• Web: Added a new method, <code>setRequestCredentials</code>. This controls whether the player makes credentialed requests to cross-origin endpoints. The remote endpoint needs to respond with the appropriate CORS response headers (like <code>Access-Control-Allow-Origin</code>, matching the request's <code>Origin</code>) and <code>Access-Control-Allow-Credentials</code> must be true.</li></ul>

Platform	Downloads and Changes
	<p>This setting persists throughout the player instance’s lifecycle. Therefore , all subsequent <code>player.load()</code> calls with URL endpoints should respond with appropriate CORS headers.</p> <p>This method has no effect on iOS browser platforms. To allow credentialed cross-origin requests on iOS platforms, users must explicitly allow Cross-site Tracking and allow Cookies; these are in the settings for the device and the respective browser app.</p>

### [Android player 1.4.0](#)

**Reference documentation:** <https://aws.github.io/amazon-ivs-player-docs/1.4.0/android/>

- Fixed an issue in which high-resolution portrait video was considered as not supported video, even though the device can support it.
- Fixed an issue in which changing the playback rate failed on certain Android devices.
- Updated background-video handling to not decode content if the output surface is not set.
- Implemented additional checks to ignore SDK calls after the `player.release()` method is called. This improves player stability.
- Reduced Android library file size through optimization.

Platform	Downloads and Changes
<a href="#">iOS Player 1.4.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.4.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.4.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.4.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.4.0/ios/</a></p> <ul style="list-style-type: none"><li>• Fixed an issue in which the memory heap size of the player continues to increase during playback until the player is deallocated.</li><li>• Fixed a potential deadlock when there is animation on top of video playback (e.g., a UI animation or GIF image).</li><li>• Fixed a potential crash during media-services reset events.</li><li>• Resolved a memory leak of <code>CMFormatDescriptionRef</code> that could occur during quality switches.</li><li>• Added an error message that is logged if IVS-specific properties of the <code>IVSPlayerView</code> and <code>IVSPlayerLayer</code> classes are accessed on a thread other than the main thread.</li><li>• Updated background-video handling to not decode content if the output surface is not set.</li><li>• Improved documentation coverage in the IOS SDK Reference.</li><li>• Reduced iOS library file size through optimization.</li></ul>

## Mobile SDK Size: Android

Architecture	Compressed Size	Uncompressed Size
arm64-v8a	1.00 MB	2.79 MB
armeabi-v7a	0.83 MB	2.15 MB
x86_64	1.11 MB	3.06 MB
x86	1.11 MB	2.93 MB

## Mobile SDK Size: iOS

Architecture	Compressed Size	Uncompressed Size
universal	0.89 MB	1.91 MB
arm64	0.45 MB	1.05 MB
armv7	0.44 MB	0.84 MB

## July 27, 2021

## Amazon IVS Broadcast SDK: Android 1.0.0 and iOS 1.0.0

Platform	Downloads and Changes
<a href="#">Android Broadcast SDK 1.0.0</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.0.0/android/">https://aws.github.io/amazon-ivs-broadcast-docs/1.0.0/android/</a>
<a href="#">iOS Broadcast SDK 1.0.0</a>	<b>Download:</b> <a href="https://broadcast.live-video.net/1.0.0/AmazonIVSBroadcast.xcframework.zip">https://broadcast.live-video.net/1.0.0/AmazonIVSBroadcast.xcframework.zip</a> <b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-broadcast-docs/1.0.0/ios/">https://aws.github.io/amazon-ivs-broadcast-docs/1.0.0/ios/</a>

## June 1, 2021

### Amazon IVS Player SDK: Android 1.3.3 and iOS 1.3.3

Platform	Downloads and Changes
Android and iOS	Fixed an issue where high-resolution portrait video was considered as not supported, although the device could support it.
<a href="#">Android player 1.3.3</a>	<b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.3.3/android/">https://aws.github.io/amazon-ivs-player-docs/1.3.3/android/</a>
<a href="#">iOS Player 1.3.3</a>	<b>Download:</b> <a href="https://player.live-video.net/1.3.3/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.3.3/AmazonIVSPlayer.xcframework.zip</a>  <b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.3.3/ios/">https://aws.github.io/amazon-ivs-player-docs/1.3.3/ios/</a> <ul style="list-style-type: none"><li>• Fixed a potential deadlock when there is animation on top of video playback (e.g., a UI animation or GIF image).</li><li>• Fixed a potential crash during media-services reset events.</li></ul>

## May 19, 2021

### Amazon IVS Player SDK: Android 1.3.2

**Reference documentation:** <https://aws.github.io/amazon-ivs-player-docs/1.3.2/android/>

To improve player stability, additional checks were implemented to ignore API calls after the `player.release()` method is called.



# May 5, 2021

## Amazon IVS Player SDK 1.3

Platform	Downloads and Changes
All	<ul style="list-style-type: none"> <li>Updated SDK documentation for using TextCue usage documentation. See the latest the Player SDK References on the <a href="#">Amazon IVS documentation landing page</a>.</li> <li>Fixed an issue with audio playback of malformed mono input streams.</li> <li>Fixed a rare playback error that could occur when playing content outside of the live HLS window.</li> <li>Improved the player's ability to play standard HLS live and recorded streams.</li> <li>Improved the accuracy of <code>getLiveLatency</code>, notably ensuring it is reset to zero when loading a new stream.</li> <li>Improved the ABR (adaptive bitrate streaming) algorithm to increase video quality more quickly when network connections improve.</li> <li>Improved player stability by reducing occurrences of rare crashes.</li> </ul>
<a href="#">Web player 1.3.1 &amp; Video.js integration</a> & <a href="#">JW player integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.3.1/amazon-ivs-player.min.js">https://player.live-video.net/1.3.1/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.3.1/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.3.1/amazon-ivs-videojs-tech.min.js</a></p>

Platform	Downloads and Changes
	<p data-bbox="829 212 1490 296"><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.3.1/web/">https://aws.github.io/amazon-ivs-player-docs/1.3.1/web/</a></p> <ul data-bbox="829 338 1503 1604" style="list-style-type: none"><li>• Fixed an issue where seek calls executed immediately after load were sometimes ignored, causing the player to begin at the wrong position.</li><li>• Fixed several issues with seeking within recorded content (also known as VOD).</li><li>• Fixed an issue where playback could fail in suboptimal network conditions.</li><li>• Added support for IVS Timed Metadata on iOS mobile web browsers.</li><li>• Fixed a bug where autoplating in desktop Safari sometimes failed.</li><li>• The Web SDK <code>getVersion</code> function no longer appends a hash to the player version.</li><li>• Fixed an issue where seeking to the exact start of a buffered range may result in another seek forward.</li><li>• Enabled low-latency ABR (adaptive bitrate streaming) in macOS Safari 14 and later.</li><li>• Fixed an issue with loading the player in a server context, by removing an unsafe import side effect.</li><li>• Changed the <code>amazon-ivs-player</code> NPM package so it exports the <code>LogLevel</code> enum, which is used by <code>setLogLevel</code>.</li></ul> <p data-bbox="829 1682 1479 1856"><b>Note:</b> The Web Player 1.3.0 NPM package exists but does not work. It is marked as deprecated on NPM. Use Web Player 1.3.1 or newer, as documented.</p>

Platform	Downloads and Changes
<a href="#">Android player 1.3.0</a>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.3.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.3.0/android/</a></p> <ul style="list-style-type: none"><li>• Fixed an issue where the player SDK crashed if the app targeted Android 11 (API level 30) and the user was running Android 11 on a cellular network.</li><li>• Fixed a network recovery issue. Playback is now automatically paused when the network connection is lost, and it is resumed when the connection is restored. Use the <code>onNetworkUnavailable</code> callback in <code>Player.Listener</code> to observe network state changes.</li><li>• Fixed an issue where player controls could not be hidden with <code>setControlsEnabled(false)</code> while playing VODs.</li><li>• Fixed an issue where the SDK could crash if the client app uses an old (pre-4.0) version of <code>OkHttp</code>.</li><li>• The Amazon IVS Android player library moved from a JCenter repository to Maven Central.</li><li>• Removed <code>BuildConfig</code> version properties from the library.</li></ul>

Platform	Downloads and Changes
<a href="#">iOS Player 1.3.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.3.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.3.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.3.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.3.0/ios/</a></p> <ul style="list-style-type: none"><li>• Fixed an issue where if there was an audio sample-rate change within a single HLS media segment, the iOS SDK could not handle it properly. This could result in an unexpected memory increase and playback failure or a crash, due to bad media.</li><li>• Fixed a network-recovery issue. Playback is now automatically paused when the network connection is lost, and it is resumed when the connection is restored. Use the <code>playerNetworkDidBecomeUnavailable</code> delegate method to observe network state changes.</li><li>• Fixed an issue which caused an iOS memory increase that could happen over time.</li><li>• Added graceful handling of audio hardware problems. Playback is now automatically paused in the event of a media-services reset notification (<code>AVAudioSessionMediaServicesWereResetNotification</code>). Note that a playback error may still occur if media is playing when the reset occurs.</li><li>• Added audio-session interruption handling. Playback is now automatically paused when an audio-session interruption begins. When the interruption ends, playback automatically resumes if the player was</li></ul>

Platform	Downloads and Changes
	previously playing and the interruption options indicate that the app should resume playback.

## April 26, 2021

### Service Quotas Integration with CloudWatch Usage Metrics

You can use CloudWatch to proactively manage your service quotas, via CloudWatch *usage metrics*. See [Amazon IVS Service Quotas](#).

## April 13, 2021

### New CloudWatch Metrics

CloudWatch metrics were added for concurrent views and concurrent streams. See [Monitoring Amazon IVS with Amazon CloudWatch](#).

Oct 12, 2023 update: This CloudWatch document was deleted and the content was moved to [Monitoring IVS Low-Latency Streaming](#).

The names of related service quotas were updated to match the new metrics. See [Service Quotas \(Low-Latency Streaming\)](#).

For a full definition of "view," see the [Amazon IVS Glossary](#).

## April 7, 2021

### Auto-Record to S3 (R2S3)

Amazon IVS now enables you to save your live video content to Amazon S3. Saved video is available later for actions like editing or replaying as a VOD.

When you enable recording for a channel, all live broadcasts of the channel are stored to an S3 bucket of your choice. All available quality renditions and thumbnails images are saved. Your recording configuration also is saved, so it can be easily re-used for additional channels.

You can set up a recording configuration and enable/disable recording through the Amazon IVS console or API. For details, see [Getting Started with IVS](#) and the [Amazon IVS API Reference](#).

## January 28, 2021

### Amazon IVS Player SDK: JW Player Integration 1.2.0

The Amazon IVS player now integrates with JW Player. See [JW Player Integration](#).

**Known Issue:** In some cases, the duration of the video appears to be 00:00 and the playhead does not seek if dragged on the seekbar. This happens only when watching an ad-free playlist with a mixture of Amazon IVS live streams and VODs, using Safari on an iPhone.

## December 16, 2020

### Amazon IVS Player: SDK for Android 1.2.1

**Reference documentation:** <https://aws.github.io/amazon-ivs-player-docs/1.2.1/android/>

This release includes an Android Player patch which fixes an issue: in prior Android player SDK releases, the SDK crashes if the app targets Android 11 (API level 30) and the user is running Android 11 on a cellular network.

## November 23, 2020

### Amazon IVS Player SDK 1.2.0

Platform	Downloads and Changes
All	Improved detection of Amazon IVS streams, so metrics are more accurate.
<a href="#">Web player 1.2.0</a> & <a href="#">Video.js integration</a>	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.2.0/amazon-ivs-player.min.js">https://player.live-video.net/1.2.0/amazon-ivs-player.min.js</a></p>

Platform	Downloads and Changes
	<p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.2.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.2.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.2.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.2.0/web/</a></p> <ul style="list-style-type: none"><li>• If the master playlist for a stream is unavailable, we now emit ErrorNotAvailable for all web playback sources.</li><li>• Updated reference documentation with respect to errors related to reaching the concurrent-viewers (CCV) limit.</li></ul>
<p><a href="#">Android player 1.2.0</a></p>	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.2.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.2.0/android/</a></p> <ul style="list-style-type: none"><li>• Fixed an issue where the getSessionId function crashed on Android.</li><li>• Updated reference documentation with respect to errors related to reaching the concurrent-viewers (CCV) limit.</li></ul> <p><b>Known Issue:</b> The player SDK will crash if the app targets Android 11 (API level 30) and the user is running Android 11 on a cellular network. This will be fixed in the next release. In the meantime, we recommend targeting a previous Android API level (29 or lower).</p>

Platform	Downloads and Changes
<a href="#">iOS Player 1.2.0</a>	<p><b>Download:</b> <a href="https://player.live-video.net/1.2.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.2.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.2.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.2.0/ios/</a></p> <ul style="list-style-type: none"><li>• Fixes a potential source of memory corruption when switching the stream URL or closing the player.</li><li>• Resolves an issue that could cause playback to fail when the best audio pitch correction could not be enabled before starting playback. Pitch correction improves audio quality at playback speeds faster or slower than normal. If pitch correction cannot be enabled or the highest quality correction algorithm is unavailable, a message is logged but playback continues.</li></ul> <p><b>Known Issue:</b> If there is an audio sample-rate change within a single HLS media segment, the iOS SDK cannot handle it properly. This can result in an unexpected memory increase and playback failure or a crash, due to bad media. This will be fixed in the next major iOS player release.</p>

## November 12, 2020

### New Event Field, `stream_id`

The `stream_id` field was added to several events. See [Using Amazon EventBridge with IVS](#).



## November 9, 2020

### Add Metadata Viewing to Console

Timed metadata can now be viewed from the Amazon IVS console. In the *Amazon IVS User Guide*, see the new section on [Viewing Timed Metadata](#) in *Embedding Metadata within a Video Stream*.

## October 30, 2020

### CloudFormation Support

Amazon IVS now supports AWS CloudFormation. This enables Amazon IVS customers to create and manage channels, stream keys, and playback key pairs with AWS CloudFormation.

Amazon IVS support for CloudFormation is available in all [AWS regions](#) where Amazon IVS is available. To get started, see the [Amazon IVS product page](#) or the [Amazon IVS information](#) in the *AWS CloudFormation User Guide*.

## October 27, 2020

### Higher Limits for Channels, CCV, and CCB

We increased three service-quota limits:

- The maximum number of *channels* that users can create, per AWS region, increased from 500 to 5,000.
- The maximum number of *concurrent viewers* allowed to play back a live channel, across all channels in an AWS region, increased from 3,000 to 15,000.
- The maximum number of *concurrent broadcasts* (channels that can be streamed simultaneously), per AWS region, increased from 30 to 100.

These increases are available in [all regions](#) where Amazon IVS is available. To learn more, see [Service Quotas \(Low-Latency Streaming\)](#) in the *Amazon IVS User Guide*.

## October 9, 2020

### New Service Quotas and EventBridge Event

There are now service quotas and EventBridge events related to ingest resolution. See [Service Quotas \(Low-Latency Streaming\)](#) and [Using Amazon EventBridge with IVS](#).

### Amazon IVS Player: SDK for Web 1.1.2

**NPM Package:** <https://www.npmjs.com/package/amazon-ivs-player>

**Script asset:** <https://player.live-video.net/1.1.2/amazon-ivs-player.min.js>

**Video.js tech asset:** <https://player.live-video.net/1.1.2/amazon-ivs-videojs-tech.min.js>

**Reference documentation:** <https://aws.github.io/amazon-ivs-player-docs/1.1.2/web/>

This release includes a Web Player patch which fixes an issue that affected viewers using Microsoft Edge. For those viewers, if auto-quality mode is turned on for the stream (i.e., ABR is in effect), low-latency playback does not work; under these circumstances, streams played back with higher latency.

## October 7, 2020

### Amazon IVS Player SDK 1.1.0

The Amazon Interactive Video Service (IVS) Player SDKs use [semantic versioning](#).

Platform	Downloads and Changes
All	<ul style="list-style-type: none"><li>Fixed an issue where the player's adaptive bitrate algorithm could incorrectly drop quality to 160p.</li><li>The player now throws an error if there are no playable video qualities.</li><li>Updated VOD seek behavior: when attempting to seek beyond the end, the</li></ul>

Platform	Downloads and Changes
	<p>player seeks to the end instead of returning an error.</p> <ul style="list-style-type: none"><li>• The player now throws a fatal error after exhausting all available qualities during error recovery.</li></ul>

Platform	Downloads and Changes
Web Player 1.1.0	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.1.0/amazon-ivs-player.min.js">https://player.live-video.net/1.1.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.1.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.1.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.1.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.1.0/web/</a></p> <p><b>Known Issues:</b></p> <ul style="list-style-type: none"><li>• If <code>Video.js</code> is not available, <code>registerIVSQualityPlugin</code> now throws an exception instead of writing to <code>console.error</code>.</li><li>• If <code>registerIVSTech</code> or <code>registerIVSQualityPlugin</code> is called more than once, calls after the first one now do nothing (instead of attempting to re-register).</li><li>• The type of the first parameter to <code>registerIVSQualityPlugin</code> has changed from <code>VideoJS</code> to <code>any</code>.</li><li>• Removed dependencies on browser context to enable server-side rendering.</li><li>• If the browser autopauses in response to unmuting, the player now fires the <code>AUDIO_BLOCKED</code> event and resumes muted playback.</li><li>• Added network connectivity recovery. A network timeout will not result in an error</li></ul>

Platform	Downloads and Changes
	<p>state being sent to the client app. Instead, when network connectivity is lost:</p> <ul style="list-style-type: none"><li>• If the app is playing, the player library sends the <code>NETWORK_UNAVAILABLE</code> event to the app and the player enters the IDLE state. When connectivity is restored, the player library resumes playing and the app receives a <code>PLAYING</code> event.</li><li>• If the app is paused, the <code>NETWORK_UNAVAILABLE</code> event is not sent to the app and the player library remains in the IDLE state. When connectivity is restored, the player library stays in the IDLE state.</li><li>• At any time, if the app tries to play, the player library attempts a normal play. The <code>NETWORK_UNAVAILABLE</code> event is sent to the app and the player enters the IDLE state.</li></ul>
Android Player 1.1.0	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.1.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.1.0/android/</a></p> <p><b>Known Issue:</b> The player SDK will crash if the app targets Android 11 (API level 30) and the user is running Android 11 on a cellular network. This will be fixed in the next release. In the meantime, we recommend targeting a previous Android API level (29 or lower).</p>

Platform	Downloads and Changes
iOS Player 1.1.0	<p><b>Download:</b> <a href="https://player.live-video.net/1.1.0/AmazonIVSPlayer.xcframework.zip">https://player.live-video.net/1.1.0/AmazonIVSPlayer.xcframework.zip</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.1.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.1.0/ios/</a></p> <ul style="list-style-type: none"> <li>Fixed an issue that could cause crashes, with this message from UIKit: "Modifications to the layout engine must not be performed from a background thread after it has been accessed from the main thread." This could occur when backgrounding and foregrounding the application.</li> </ul>

## September 14, 2020

### New Event Field, channel\_name

The channel\_name field was added to several events. See [Using Amazon EventBridge with IVS](#).

## August 19, 2020

### Playback Authorization (Private Channels)

Amazon IVS now offers customers the ability to create private channels, allowing customers to restrict which viewers can watch their streams. Customers control access to video playback by enabling playback authorization on channels and generating signed JSON Web Tokens (JWTs) for authorized playback requests. For details, see [Setting Up Private Channels](#).

A new authorized field in the Channel object indicates whether the channel is private. See the [Amazon IVS API Reference](#).

## August 11, 2020

### Amazon IVS Player: SDK for iOS 1.0.6

**Download:** <deprecated>

**Reference documentation:** <https://aws.github.io/amazon-ivs-player-docs/1.0.6/ios/>

This release includes an iOS Player patch which fixes an issue that had prevented some iOS Player apps from being added to the Apple App Store. Specifically, apps built with bitcode enabled would fail App Store Connect validation after uploading.

## August 5, 2020

### Using Amazon EventBridge with Amazon IVS

Amazon IVS EventBridge events are now available through the Amazon EventBridge console. See the section on [Creating Amazon EventBridge Rules for Amazon IVS](#) in *Using Amazon EventBridge with Amazon IVS*, in the *Amazon IVS User Guide*.

## July 15, 2020

### Player Version 1.0

The Amazon Interactive Video Service (IVS) Player SDKs use [semantic versioning](#).

Platform	Downloads and Changes
All	<b>Known Issue:</b> For the <code>setAutoMaxQuality</code> and <code>setQuality</code> functions, the quality you provide is applied correctly to the current stream but is not applied correctly if you load a new stream. To avoid this, if you load a new stream, call this with a quality for the new stream after <code>PlayerState.READY</code> .

Platform	Downloads and Changes
Web Player 1.0.0	<p><b>NPM Package:</b> <a href="https://www.npmjs.com/package/amazon-ivs-player">https://www.npmjs.com/package/amazon-ivs-player</a></p> <p><b>Script asset:</b> <a href="https://player.live-video.net/1.0.0/amazon-ivs-player.min.js">https://player.live-video.net/1.0.0/amazon-ivs-player.min.js</a></p> <p><b>Video.js tech asset:</b> <a href="https://player.live-video.net/1.0.0/amazon-ivs-videojs-tech.min.js">https://player.live-video.net/1.0.0/amazon-ivs-videojs-tech.min.js</a></p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.0.0/web/">https://aws.github.io/amazon-ivs-player-docs/1.0.0/web/</a></p> <p><b>Known Issues:</b></p> <ul style="list-style-type: none"><li>• When playing a VOD on an iOS mobile browser (e.g. Safari or Chrome), seeking backwards will mute the player. To avoid this, call <code>player.setMuted(false)</code> after seeking.</li><li>• When playing a VOD on an iOS mobile browser, seeking backwards works intermittently when directly selecting the desired position. To avoid this, drag the seek bar to the desired position.</li><li>• When playing a VOD on an iOS mobile browser using the Video.js integration, the replay button does not work properly. To avoid this, hide the replay button when initializing Video.js: <a href="https://docs.videojs.com/tutorial-components.html#play-toggle">https://docs.videojs.com/tutorial-components.html#play-toggle</a>.</li></ul>



Platform	Downloads and Changes
Android Player 1.0.0	<p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.0.0/android/">https://aws.github.io/amazon-ivs-player-docs/1.0.0/android/</a></p> <p><b>Known Issue:</b> Backgrounding and foregrounding can cause audio/video de-synchronization for VOD playback on Android.</p>
iOS Player 1.0.0	<p><b>Download:</b> &lt;deprecated&gt;</p> <p><b>Reference documentation:</b> <a href="https://aws.github.io/amazon-ivs-player-docs/1.0.0/ios/">https://aws.github.io/amazon-ivs-player-docs/1.0.0/ios/</a></p> <p><b>Known Issues:</b></p> <ul style="list-style-type: none"><li>• Backgrounding and foregrounding cause live and VOD playback failure. To avoid this, pause the stream when the <code>UIApplicationDidEnterBackgroundNotification</code> is received and resume play on the <code>UIApplicationDidBecomeActiveNotification</code>.</li><li>• iOS 10 devices may experience a crash when returning from background. To avoid this, set the <code>layer.player</code> property to <code>nil</code> before entering the background.</li></ul>