aws

User Guide

AWS Elemental MediaPackage

AWS Elemental MediaPackage: User Guide

Table of Contents

What is AWS Elemental MediaPackage?

AWS Elemental MediaPackage (MediaPackage) is a just-in-time video packaging and origination service that runs in the AWS Cloud. With MediaPackage, you can deliver highly secure, scalable, and reliable video streams to a wide variety of playback devices and content delivery networks (CDNs).

MediaPackage offers a broadcast-grade viewing experience for viewers, while allowing you the flexibility to control and protect your content. Additionally, the built-in resiliency and scalability of MediaPackage means that you have the right amount of resources at the right time, with no manual intervention required.

Topics

Are you a first-time user of MediaPackage?

If you're a first-time user of MediaPackage, we recommend that you begin by reading the following sections:

- Concepts and terminology
- How MediaPackage works
- Features of AWS Elemental MediaPackage
- Getting started with AWS Elemental MediaPackage

Concepts and terminology

AWS Elemental MediaPackage (MediaPackage) includes the following components:

Just-in-time packaging

MediaPackage performs *just-in-time packaging* (JITP). When a playback device requests content, MediaPackage dynamically customizes the live video streams and creates a manifest in a format that's compatible with the requesting device.

Origination service

MediaPackage is considered an *origination service* because it's the point of distribution for media content delivery.

Packager

A *packager* prepares output streams for access by different types of players. The packager type specifies the streaming format that MediaPackage delivers from the endpoint (either Apple HLS, DASH-ISO, Microsoft Smooth Streaming, or Common Media Application Format [CMAF]). Additional packager settings include buffer and update durations and manifest tag handling instructions.

A packager is a part of an endpoint. Each endpoint must have one, and only one, packager. To use different packager types for the same content, create multiple endpoints on the channel.

Source Content

Source contents are live streams and video files that MediaPackage ingests.

- For live video, source content comes from an upstream encoder, such as AWS Elemental MediaLive. MediaPackage supports HLS source content.
- For video on demand (VOD), source content resides in an Amazon Simple Storage Service (Amazon S3) bucket in your AWS account. MediaPackage supports HLS and MP4 (.smil manifest format) content.

Stream

A *stream* refers to the content input and output of MediaPackage.

For live workflows, an upstream encoder sends a live stream as an input to MediaPackage to the channel. When a downstream device requests playback of the content, MediaPackage

dynamically packages the stream (including specifying the packager type, adding encryption, and configuring track outputs) and delivers it to the requesting device as an output of the endpoint. An endpoint can produce multiple streams.

For VOD workflows, MediaPackage pulls file-based content from Amazon S3. As with live workflows, when a downstream device requests playback of the content, MediaPackage dynamically packages the stream and delivers it to the requesting device as an output of the asset resource.

Track

Tracks make up the output content stream. MediaPackage includes selected video, audio, and subtitles or captions tracks in the output stream. The stream delivers the tracks to the player (either directly or through a CDN), and the player plays back the tracks based on player logic or network conditions (such as available bandwidth).

Live components

The following components apply to live workflows in MediaPackage:

Channel

A *channel* represents the entry point for a content stream into MediaPackage. Upstream encoders such as AWS Elemental MediaLive send content to the channel. When MediaPackage receives a content stream, it packages the content and outputs the stream from an endpoint that you create on the channel. There's one channel for each incoming set of adaptive bitrate (ABR) streams.

Endpoint

An *endpoint* is part of a channel and represents the packaging aspect of MediaPackage. When you create an endpoint on a channel, you indicate what streaming format, packaging parameters, and features the output stream will use. Downstream devices request content from the endpoint. A channel can have multiple endpoints.

Harvest Job

A *harvest job* is a task that you create to extract a VOD asset from a live content stream. A harvest job defines the start and end times of the VOD asset, and where MediaPackage stores the asset. When the job runs, MediaPackage creates an HLS clip for the times that you

indicated. This clip is stored as a VOD asset in an Amazon S3 bucket of your choosing. You can use the VOD functionality in MediaPackage to serve the asset to end users.

VOD Components

The following components apply to VOD workflows in MediaPackage:

Asset

An *asset* represents the entry point for file-based content into MediaPackage. MediaPackage uses the information in the asset to locate and ingest your source content from Amazon S3. When you create an asset in MediaPackage, you associate it with a *packaging group*, which holds one or more *packaging configurations*. Each asset and packaging configuration combination provides a URL for playback of repackaged content. Each asset is associated with all the packaging configurations within one packaging group.

Packaging Configuration

A *packaging configuration* defines how MediaPackage formats, encrypts, and delivers source content to viewers. The packaging configuration includes settings such as stream selection, encryption, segment duration and combining, and one or more HLS, DASH, MSS, or CMAF manifest definitions.

Packaging Group

A *packaging group* is a set of one or more packaging configurations. Because you can associate the group to more than one asset, the group provides an efficient way to associate multiple packaging configurations with multiple assets.

Supported inputs and outputs

This section describes the input types, input codecs, and output codecs that AWS Elemental MediaPackage supports for live and video on demand (VOD) content.

Topics

Live supported codecs and input types

The following sections describe supported input types, input codecs, and output codecs for live streaming content.

Supported input types

These are the input types that MediaPackage supports for live content.

MediaPackage input type	Use case
HLS	Push an HLS stream from an external source or encoder (such as AWS Elemental MediaLive) using the HTTPS protocol. Additional requirements: • Inputs must be over WebDAV and with digest authentication. • Media segments must not be encrypted. • Streams can contain either muxed video and audio tracks, or unmuxed tracks. • The input must contain at least one video track. MediaPackage doesn't support inputs that contain no video track.

Supported input codecs

These are the video, audio, and subtitles codecs that MediaPackage supports for source content streams.

Media container	Video codecs	Audio codecs	Subtitles/captions format
• Video: TS • Audio: TS, AAC, AC3, or EC3	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital	• WebVTT

Media container	Video codecs	Audio codecs	Subtitles/captions format
		• Dolby Digital Plus	• CEA-608 and CEA-708 closed captions

Supported output codecs

These are the video, audio, and subtitles codecs that MediaPackage supports when delivering live content.

Endpoint type	Manifest format	Media container	Video codecs	Audio codecs	Subtitles /captions format
Apple HLS	HLS	• Video: TS • Audio: TS or AAC	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• WebVTT • CEA-608 and CEA-708 closed captions
DASH-ISO	MPEG-DASH	MP4	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• EBU-TT • CEA-608 and CEA-708 closed captions
Microsoft Smooth	MSS	MP4	• H.264 (AVC) • H.265 (HEVC) with	• AAC • Dolby Digital	DFXP

Endpoint type	Manifest format	Media container	Video codecs	Audio codecs	Subtitles /captions format
			HDR-10 support	• Dolby Digital Plus	
CMAF	HLS	CMAF	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• WebVTT • CEA-608 and CEA-708 closed captions

Live-to-VOD supported codecs and input types

The following sections describe supported input types, input codecs, and output codecs for live-to-VOD assets that are harvested from streaming content in AWS Elemental MediaPackage.

Supported input types

These are the input types that MediaPackage supports for live-to-VOD assets.

MediaPackage input type	Use case
HLS	Extract a portion of a live HLS or DASH endpoint in MediaPackage and save it as a live-to-VOD asset. Additional requirements: • The endpoint must have a defined startover window, which determines the maximum length of the live-to-VOD asset that can be harvested. • Streams can contain either muxed video and audio tracks, or unmuxed tracks.

MediaPackage input type	Use case
	• The input must contain at least one video track. MediaPackage doesn't support inputs that contain no video track.

Supported input codecs

These are the video, audio, and subtitles codecs that MediaPackage supports for live-to-VOD assets.

Media container	Video codecs	Audio codecs	Subtitles/captions format
• Video: TS • Audio: TS, AAC, AC3, or EC3	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• WebVTT • CEA-608 and CEA-708 closed captions

Supported output codecs

These are the video, audio, and subtitles codecs that MediaPackage supports when saving a live-to-VOD asset to an Amazon S3 bucket. The endpoint must serve either clear (unencrypted) or encrypted DASH or HLS content.

Endpoint type	Manifest format	Media container	Video codecs	Audio codecs	Subtitles /captions format
Apple HLS	HLS	• Video: TS • Audio: TS or AAC	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• WebVTT • CEA-608 and CEA-708 closed captions

Endpoint type	Manifest format	Media container	Video codecs	Audio codecs	Subtitles /captions format
DASH-ISO	MPEG-DASH	MP4	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• EBU-TT • CEA-608 and CEA-708 closed captions

VOD supported codecs and input types

The following sections describe supported input types, input codecs, and output codecs for file-based video on demand (VOD) content.

Supported input types

These are the input types that MediaPackage supports for VOD content.

MediaPackage input type	Use case
HLS	Pull an HLS stream set from an Amazon S3 bucket, with or without a secure connection. Additional requirements: • Media segments must not be encrypted. • Streams can contain either muxed video and audio tracks, or unmuxed tracks. • The input must contain at least one video track. MediaPackage doesn't support inputs that contain no video track.
SMIL	Pull an MP4 stream set referenced by a .smil manifest from an Amazon S3 bucket, with or

MediaPackage input type	Use case
	without a secure connection. For information about the .smil manifest, see Requirements for .smil manifests. Additional requirements: • MP4 container must not be fragmented. • Media segments must not be encrypted. • Streams can contain either muxed video and audio tracks, or only video tracks. • Streams must have an equal time base.

Supported input codecs

These are the video, audio, and subtitles codecs that MediaPackage supports for file-based source content.

Input type	Media container	Video codecs	Audio codecs	Subtitles/ captions format
HLS	• Video: TS • Audio: TS, AAC, AC3, or EC3	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• WebVTT • CEA-608 and CEA-708 closed captions
SMIL	MP4 (non-frag mented)	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	SRT

Supported output codecs

These are the video, audio, and subtitles codecs that MediaPackage supports for delivering VOD content.

Endpoint type	Manifest format	Media container	Video codecs	Audio codecs	Subtitles /captions format
Apple HLS	HLS	• Video: TS • Audio: TS, AAC, AC3, or EC3	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• WebVTT • CEA-608 and CEA-708 closed captions
DASH-ISO	MPEG-DASH	MP4	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	• EBU-TT • CEA-608 and CEA-708 closed captions
Microsoft Smooth	MSS	MP4	• H.264 (AVC) • H.265 (HEVC) with HDR-10 support	• AAC • Dolby Digital • Dolby Digital Plus	DFXP
CMAF	HLS	CMAF	• H.264 (AVC) • H.265 (HEVC)	• AAC • Dolby Digital	• WebVTT • CEA-608 and CEA-708

Endpoint type	Manifest format	Media container	Video codecs	Audio codecs	Subtitles /captions format
			with HDR-10 support	• Dolby Digital Plus	closed captions

Requirements for .smil manifests

When sending a VOD MP4 asset to AWS Elemental MediaPackage, a .smil manifest must be included. The .smil manifest is an XML file that acts as a wrapper for all the files in the asset, letting MediaPackage know which MP4s are part of a single asset.

Resources

- For guidance on creating a .smil manifest, see .smil using AWS Elemental MediaPackage VOD (blog).
- For general information about Synchronized Multimedia Integration Language (SMIL), see the SMIL 3.0 specification.

MediaPackage supports the following attributes in a .smil manifest.

Attributes

- `audioName` - The name of the audio track, such as `English 2`.
- `includeAudio` - A Boolean value indicating if the audio tracks should be included. This attribute should contain as many values as there are languages defined. If not specified, all tracks default to `true`.
- `src` or `name` - Either the name or the source of the text stream or video file relative to the manifest location.
- `subtitleName` - The subtitle name, such as `English`.
- `systemLanguage` or `language` - The system language, such as `eng`.

Example.smil manifest

The following is an example of a `.smil` manifest.

```
<?xml version="1.0" encoding="utf-8"?>
<smil>
    <body>
        <alias value="Example"/>
        <switch>
            <video name="example_360.mp4" systemLanguage="eng,fra,spa"
 audioName="English,French,Spanish" includeAudio="true,true,true"/>
            <video name="example_480.mp4" systemLanguage="eng" audioName="English 2"
 includeAudio="false"/>
            <textstream src="example_subs_eng.srt" systemLanguage="eng"
 subtitleName="English" includeAudio="false"/>
            <textstream src="example_subs_fra.srt" systemLanguage="fra"
 subtitleName="French" includeAudio="false"/>
            <textstream src="example_subs_spa.srt" systemLanguage="spa"
 subtitleName="Spanish" includeAudio="false"/>
    </switch>
</body>
</smil>
```

How MediaPackage works

AWS Elemental MediaPackage (MediaPackage) uses just-in-time format conversion to deliver over-the-top (OTT) video from a single source to a wide variety of playback devices or content delivery networks (CDNs).

The following sections describe how MediaPackage works.

Topics

Live content processing

In the processing flow for live content, encoders send live HLS streams to MediaPackage. MediaPackage then packages the content, formatting it in response to playback requests from downstream devices.

The following sections describe the live processing flows.

Topics

General MediaPackage live processing flow

The following outlines the general flow of live content in MediaPackage:

1. An upstream encoder (such as AWS Elemental MediaLive) sends an HLS live stream with digest authentication over WebDAV to the MediaPackage channel input URL, and includes the channel's access credentials (as supplied in MediaPackage). If you're using input redundancy, the encoder sends two identical HLS live streams to MediaPackage, one to each input URL on the channel. MediaPackage uses the stream from one input URL as the source content. If MediaPackage stops receiving content on the active input URL, it automatically switches to the other input URL for source content. Additionally, AWS scales resources up and down to handle the incoming traffic.

   For more information, see Live input redundancy AWS Elemental MediaPackage processing flow.

   > **Note**
   >
   > To allow support for features like time-shifted viewing, MediaPackage stores all received content for a limited time. This stored content is only available for playback if it falls within the **startover window** that's defined on the endpoint. Stored content isn't available for playback if it's outside the startover window, or if you haven't defined a window on the endpoint. For more information, see Time-shifted viewing reference in AWS Elemental MediaPackage.

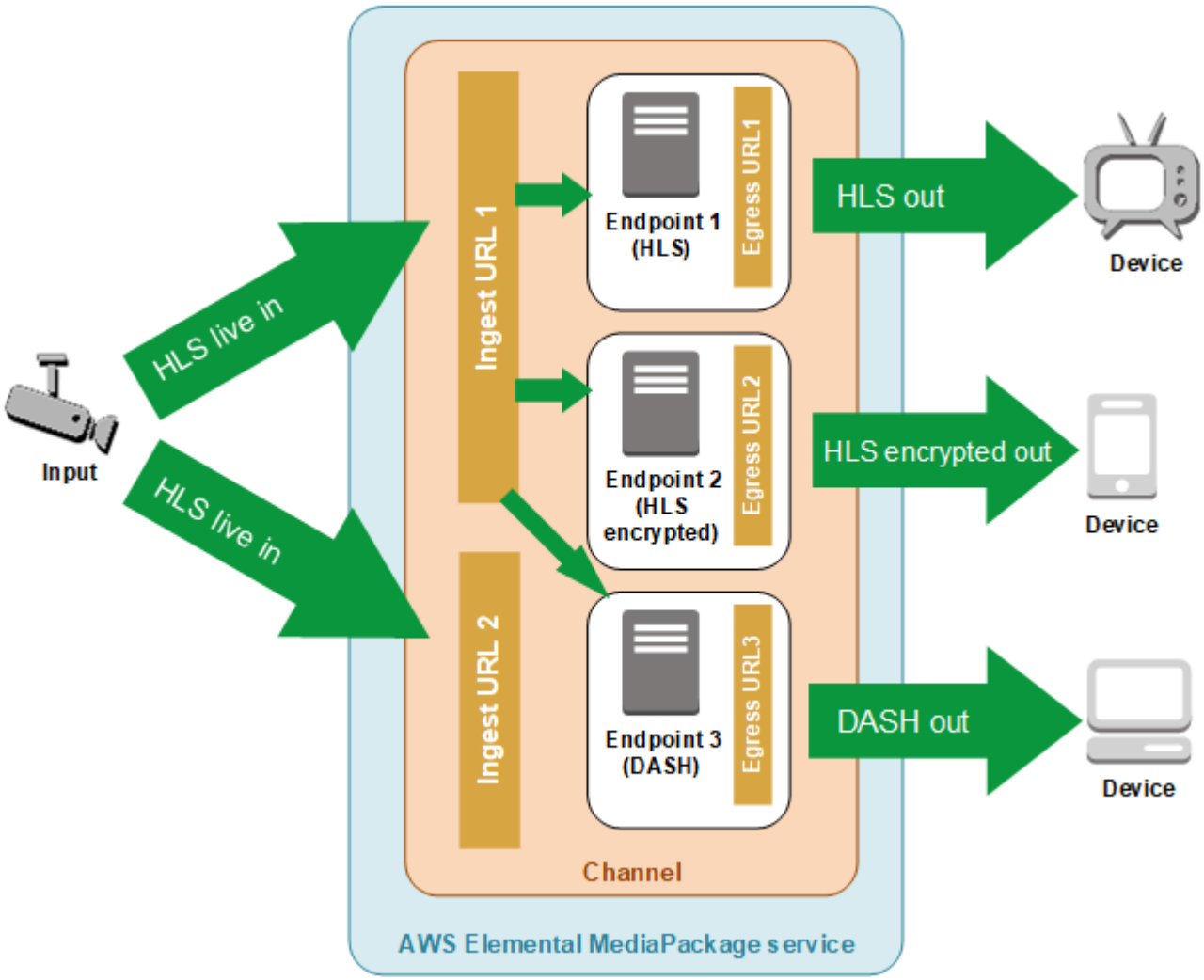
2. A downstream device requests content from MediaPackage through the endpoint output URL. A downstream device is either a video player or a CDN. The output URL is associated with an endpoint for a specific streaming format (either Apple HLS, DASH-ISO, Microsoft Smooth Streaming, or CMAF).
3. When MediaPackage receives the playback request from the downstream device, it dynamically packages the stream according to the settings that you specified on the endpoint. Packaging can include adding encryption and configuring audio, video, and subtitles or captions track outputs.

Be sure to order your inputs so that your preferred audio rendition is listed first in the audio section of the parent manifest. Do the same for the subtitles or captions. When packaging audio and subtitles or captions tracks, MediaPackage designates the first audio and captions or subtitles track as `DEFAULT=YES` and `AUTO-SELECT=YES`. This packaging overrides default and auto-select settings from the input.

4. MediaPackage delivers the output stream over HTTPS to the requesting device. As with input, AWS scales resources up and down to handle changes in traffic.
5. MediaPackage logs activity through Amazon CloudWatch. You can view information such as the number of content requests and amount of content that MediaPackage has received or delivered. For information about viewing MediaPackage metrics in CloudWatch, see Monitoring AWS Elemental MediaPackage with Amazon CloudWatch metrics.

Throughout the content input and output processes, MediaPackage detects and mitigates potential infrastructure failures before they become a problem for viewers.

The following illustration shows the overall process.



Live input redundancy AWS Elemental MediaPackage processing flow

Achieve input redundancy in AWS Elemental MediaPackage by sending two streams to separate input URLs on a channel in MediaPackage. One of the streams becomes the primary, active source of content for the endpoints, while the other continues to passively receive content. If MediaPackage stops receiving content from the active stream, it switches over to the other input stream so that content playback isn't interrupted.

If you use MediaPackage with AWS Elemental MediaLive (for example), here's the flow of input redundancy:

1. You create a channel in MediaPackage, as described in Creating a channel. When MediaPackage provisions the channel, it creates two input URLs for the channel. If you're not using input redundancy, you can send a stream to either input URL. There's no requirement that you send content to both URLs.

> **Note**
>
> When input redundancy became available, MediaPackage added a second input URL to existing channels and updated the existing URL to a new format. You can use either the existing URL or the new URLs for content input.

2. You create an endpoint in MediaPackage as described in Creating an endpoint.

> **Important**
>
> If you use short output segments, depending on your playback device, you might see buffering when MediaPackage switches inputs. You can reduce buffering by using the time delay feature on the endpoint. Be aware that using a time delay introduces latency to end-to-end delivery of the content. For information about enabling time delay, see Creating an endpoint.

3. You create an input and channel in AWS Elemental MediaLive, and you add a MediaPackage output group to the channel in MediaLive. For more information, see Creating a Channel from Scratch in the *AWS Elemental MediaLive User Guide*.

   If you use an HLS output group in AWS Elemental MediaLive, the input loss action on the HLS group's settings must be set to pause the output if the service doesn't receive input. If MediaLive sends a black frame or some other filler frame when it's missing input, then MediaPackage can't tell when segments are missing, and subsequently can't perform failover. For more information about setting the input loss action in MediaLive, see Fields for the HLS Group in the *AWS Elemental MediaLive User Guide*.

> **Important**
>
> If you use a different encoder (not AWS Elemental MediaLive) and you send two separate streams to the same channel in MediaPackage, the streams must have identical encoder settings and manifest names. Otherwise, input redundancy might not work correctly and playback could be interrupted if the inputs switch.

4. You start the channel in AWS Elemental MediaLive to send the streams to MediaPackage.
5. MediaPackage receives content on both of the input URLs, but only one of the streams is used for source content at a time. If the active stream is missing any segments, then MediaPackage

automatically fails over to the other stream. MediaPackage continues to use this stream until failover is needed again.

The formula that's used to determine if an input is missing segments is based on the segment lengths on the inputs and the endpoints. If an input is missing segments and quickly recovers, an endpoint with longer segment lengths won't switch inputs. This might result in different endpoints on the channel using different inputs (if one endpoint switches and the other doesn't). This is expected behavior and should not affect the content workflow.

VOD Content Processing

In the processing flow for VOD content, AWS Elemental MediaPackage ingests file-based video content from Amazon S3. MediaPackage then packages the content, formatting it in response to playback requests from downstream devices.

Here is the general processing flow for VOD content in MediaPackage:

1. From the MediaPackage asset, you initiate ingest of the source content from an Amazon S3 bucket. This process can take several minutes. You receive an Amazon CloudWatch event when ingest is complete and the playback URLs are live.
2. A downstream device requests content from MediaPackage through the packaging configuration URL on the asset. A downstream device is either a video player or a CDN. The URL is associated with a configuration for a specific streaming format (either Apple HLS, DASH-ISO, Microsoft Smooth Streaming, or CMAF).
3. When MediaPackage receives the playback request from the downstream device, it dynamically packages the stream according to the settings that you specified in the packaging configuration. Packaging can include adding encryption and configuring audio, video, and subtitles or captions track outputs.

   Be sure to order your inputs so that your preferred audio rendition is listed first in the audio section of the parent manifest. Do the same for the subtitles or captions. When packaging audio and subtitles or captions tracks, MediaPackage designates the first audio and captions or subtitles track as `DEFAULT=YES` and `AUTO-SELECT=YES`. This packaging overrides default and auto-select settings from the input.
4. MediaPackage delivers the output stream over HTTPS to the requesting device. As with input, AWS scales resources up and down to handle changes in traffic.

5. MediaPackage logs activity through Amazon CloudWatch. You can view information like the number of content requests and amount of content that MediaPackage has delivered. For information about viewing MediaPackage VOD metrics in CloudWatch, see Monitoring AWS Elemental MediaPackage with Amazon CloudWatch metrics.

Throughout the content input and output processes, MediaPackage detects and mitigates potential infrastructure failures before they become a problem for viewers.

Live and VOD manifest reference

AWS Elemental MediaPackage delivers live and video on demand (VOD) manifests to requesting devices. A live manifest indicates that the content isn't complete. New content continually becomes available through the playback endpoint. Alternatively, a VOD manifest indicates that the program is complete, or will be complete at a specified time in the future.

This section describes the differences in live and VOD manifests, and explains when MediaPackage delivers each manifest type.

Manifest properties

These are the main properties in a manifest that determine if it's live or VOD:

- For HLS and CMAF VOD manifests, `EXT-X-ENDLIST` is at the end of the bitrate manifests. In live manifests, this tag isn't present.
- For MPEG-DASH VOD manifests, `type="static"` is in the `MPD` properties. In live manifests, `type=dynamic`.
- For Microsoft Smooth VOD manifests, `IsLive` isn't present in the `SmoothStreamingMedia` properties. In live manifests, `IsLive=TRUE`.

For VOD, the scrub bar on playback devices also often shows that the program has a limited duration. This duration is equal to the length of the current manifest. If a playback request defines a specific playback window, this duration is equal to the length of that playback window.

To determine if the manifest is live or VOD, see Live and VOD manifest reference.

When a manifest is VOD

MediaPackage delivers a VOD manifest when the content of the program is complete. MediaPackage considers a program complete under the following conditions:

There's an end parameter in the past.

When a playback request includes an end parameter that's set in the past, the content is complete. No new content is added to it. MediaPackage delivers a static, VOD manifest to downstream devices.

For information about start and end parameters in playback requests, see Time-shifted viewing reference in AWS Elemental MediaPackage.

The manifest that the upstream encoder delivers to MediaPackage includes an `EXT-X-ENDLIST` tag.

When you stop the output from your encoder, the manifest that it sends to MediaPackage includes an `EXT-X-ENDLIST` tag. This tag tells MediaPackage that the content is complete, and no new content will be added. MediaPackage delivers a static, VOD manifest to downstream devices.

> **Note**
>
> If you manually stop an AWS Elemental MediaLive channel when one or both pipelines to MediaPackage are stopped, MediaLive doesn't include `EXT-X-ENDLIST` in the HLS manifest to MediaPackage. MediaPackage continues to produce a live manifest.
> If both pipelines are active when you stop the channel, MediaLive includes `EXT-X-ENDLIST`. MediaPackage delivers a VOD manifest to downstream devices.

If you restart the output from the encoder, the manifest from MediaPackage becomes live again. Playback devices might need to refresh to resume content playback.

If you're using input redundancy and the active stream ends, MediaPackage fails over to the other incoming stream for input. The manifest isn't marked as complete unless both incoming streams end.

Features of AWS Elemental MediaPackage

MediaPackage supports the following features:

Accessibility support

MediaPackage supports audio and subtitles accessibility signaling for HLS, CMAF, and DASH VOD assets that are created from an HLS source.

- Audio accessibility signaling supports functionality like Descriptive Voice Services (DVS) to help make media accessible to people who are blind or visually impaired. For example, an audio track might be used to provide an audio description of the scene.
- Subtitles accessibility signaling helps make media accessible to people who are deaf or hard of hearing. For example, a subtitles track might be used to provide description of music and sound effects in the video.

To enable players to provide accessibility signaling, MediaPackage passes through the `EXT-X-MEDIA` tag and attributes from the source playlist.

> **Important**
>
> The `EXT-X-MEDIA` tag must include a `CHARACTERISTICS` attribute with an appropriate value for accessibility signalling to work.
>
> - For audio accessibility, the value must be `public.accessibility.describes-video`.
> - For subtitles accessibility, the value must include one or both of `public.accessibility.describes-music-and-sound` and `public.accessibility.transcribes-spoken-dialog`.

Example EXT-X-MEDIA tag with accessibility caption attribute

```
#EXT-X-MEDIA:TYPE=SUBTITLES,GROUP-ID="captions-
group",NAME="accessibility-captions1",LANGUAGE="eng",
CHARACTERISTICS="public.accessibility.transcribes-
spoken-dialog,public.accessibility.describes-music-and-
sound",AUTOSELECT=YES,DEFAULT=YES,URI="caption-accessibility-eng.m3u8"
```

Allow listing

Allow listing is available with only live workflows in MediaPackage.

MediaPackage supports restricting network access to the endpoint. To take advantage of this feature, you must enter the allowed IP addresses on the endpoint. For more information about adding allow listing information, see Access control settings fields.

Audio

MediaPackage supports multi-language audio inputs and the following audio codecs:

- AAC stereo
- Dolby AC3 and E-AC3 (Dolby Digital and Dolby Digital+)

MediaPackage accepts these codecs from the input source and passes them through to the output stream.

Be sure to order your inputs so that your preferred audio rendition is listed first in the audio section of the parent manifest. When packaging audio and subtitles or captions tracks, MediaPackage designates the first audio track as `DEFAULT=YES` and `AUTO-SELECT=YES`. This packaging overrides default and auto-select settings from the input.

> **Important**
>
> MediaPackage doesn't support audio-only inputs. The stream configuration from the encoder must include at least one video track.

Captions

Your embedded source captions can be CEA-608 captions, CEA-708 captions, or both CEA-608 and CEA-708. MediaPackage will pass through these captions in the media segments and parent manifest on HLS, CMAF, and DASH endpoints, and generate the appropriate manifest signaling.

Be sure to order your inputs so that your preferred captions rendition is listed first in the captions section of the parent manifest. When packaging captions tracks, MediaPackage designates the first captions track as `DEFAULT=YES` and `AUTO-SELECT=YES`. This packaging overrides default and auto-select settings from the input.

> **Important**
>
> Your input HLS playlist must include captions signaling tags. If not present, MediaPackage will not be able to generate the corresponding output manifest signaling.

CDN Authorization

MediaPackage supports content delivery network (CDN) authorization. For information, see CDN authorization in AWS Elemental MediaPackage.

DRM

MediaPackage supports content protection through digital rights management (DRM). For information, see Content encryption and DRM in AWS Elemental MediaPackage.

HLS Rendition Groups

MediaPackage supports rendition groups for incoming and outgoing HLS content. For information about output rendition groups, see Rendition groups reference in AWS Elemental MediaPackage.

Live to VOD

Use the harvest job resource to extract a live-to-VOD (video on demand) asset from a live content stream. MediaPackage creates the asset and stores it in an Amazon S3 bucket. You can use the VOD functionality in MediaPackage to deliver the asset to end users.

Input Redundancy

Input redundancy is available with only live workflows in MediaPackage.

MediaPackage creates two input URLs on every channel so that you can create input redundancy by sending two identical streams to the same channel. For information about how input redundancy works, see Live input redundancy AWS Elemental MediaPackage processing flow.

Subtitles

MediaPackage supports input WebVTT text-based subtitles. MediaPackage translates the subtitles to the appropriate format based on the packager that's used on the endpoint:

- For HLS and CMAF: WebVTT is passed through
- For DASH: subtitles are translated to EBU-TT
- For Microsoft Smooth Streaming: subtitles are translated to DFXP

Be sure to order your inputs so that your preferred subtitles rendition is listed first in the subtitles section of the parent manifest. When packaging subtitles tracks, MediaPackage

designates the first subtitles track as `DEFAULT=YES` and `AUTO-SELECT=YES`. This packaging overrides default and auto-select settings from the input.

Time-shift Viewing

Time-shift viewing is available with only live workflows in MediaPackage.

MediaPackage allows playback of a stream at a time earlier than the current time. Start-over, catch-up TV, and time delay are all supported. For more information about setting up time-shift capabilities, see Time-shifted viewing reference in AWS Elemental MediaPackage.

Video

MediaPackage supports the input H.264 video codec and passes it through to the output stream. CMAF endpoints in MediaPackage also support H.265/HEVC and HDR-10, following the Apple specification to applicable playback devices.

> **Important**
>
> MediaPackage requires at least one video track to be present in the stream configuration from the encoder. The service doesn't support audio-only ingest.

Related services

- **Amazon CloudFront** is a global content delivery network (CDN) service that securely delivers data and videos to your viewers. Use CloudFront to deliver content with the best possible performance. For more information, see Amazon CloudFront.
- **Amazon CloudWatch** is a monitoring service for AWS Cloud resources and the applications that you run on AWS. Use CloudWatch to track metrics such as content input and output request counts. For more information, see Amazon CloudWatch.
- **AWS Elemental MediaLive (MediaLive)** is a live video processing service that encodes high-quality live video streams for broadcast television and multi-screen devices. Use MediaLive to encode content streams and send them to MediaPackage for packaging. For more information about how encoders (such as MediaLive) work with MediaPackage, see How MediaPackage works.
- **AWS Identity and Access Management (IAM)** is a web service that helps you securely control access to AWS resources for your users. Use IAM to control who can use your AWS resources (authentication) and what resources users can use in which ways (authorization). For more information, see *Setting up*.

- **AWS Elemental MediaTailor (MediaTailor)** is a scalable ad insertion service that runs in the AWS Cloud. Use MediaTailor to serve targeted ads to viewers. For more information, see AWS Elemental MediaTailor.
- **Amazon Simple Storage Service (Amazon S3)** is a storage service. Pull video on demand (VOD) assets from Amazon S3, or store live-to-VOD assets in the bucket of your choice. For more information, see Getting started with VOD content delivery in MediaPackage and Getting started with live-to-VOD content delivery in MediaPackage.

Accessing MediaPackage

You can access MediaPackage using any of the following methods.

- **AWS Management Console** - The procedures throughout this guide explain how to use the AWS Management Console to perform tasks for MediaPackage.

```
https://console.aws.amazon.com/mediapackage/
```

- **AWS Command Line Interface** - For more information, see the AWS Command Line Interface User Guide.

```
aws mediapackage
```

- **MediaPackage API** - For information about API actions and about how to make API requests, see the AWS Elemental MediaConnect API Reference.

```
https://mediapackage.region.amazonaws.com
```

- **AWS SDKs** - If you're using a programming language that AWS provides an SDK for, you can use an SDK to access MediaPackage. SDKs simplify authentication, integrate easily with your development environment, and provide easy access to MediaPackage commands. For more information, see Tools for Amazon Web Services.
- **AWS Tools for Windows PowerShell** - For more information, see the AWS Tools for PowerShell User Guide.

Pricing for MediaPackage

As with other AWS products, there are no contracts or minimum commitments for using MediaPackage. You are charged only for AWS resources that your account uses. Pricing is pay-as-you-go and consists of the following:

- A per GB charge for received content
- A per GB charge for content that's streamed out of MediaPackage

  Content that's cached and served from a content delivery network (CDN) doesn't incur this per GB charge.

For detailed pricing information, see MediaPackage Pricing.

Regions for MediaPackage

To reduce latency in your applications, MediaPackage offers a regional endpoint for your requests. To view the list of AWS Regions where MediaPackage is available, see MediaPackage Regions.

MediaPackage control plane APIs support IPv6 in all supported regions. For more information, see *IPv6 support*.

AWS opt-in Regions

Although most AWS Regions are active by default for your AWS account, certain Regions are activated only when you manually select them. This document refers to those Regions as *opt-in Regions*. In contrast, Regions that are active by default, as soon as your AWS account is created, are referred to as *commercial Regions*, or simply, *Regions*.

The term *opt-in* has a historical basis. Any AWS Regions introduced after March 20, 2019 are considered to be opt-in Regions. Opt-in Regions have higher security requirements than commercial Regions, regarding the sharing of IAM data through accounts that are active in opt-in Regions. All of the data managed through the IAM service is considered identity data, including users, groups, roles, policies, identity providers, their associated data (for example, X.509 signing certificates or context-specific credentials), and other account-level settings, such as password policy and the account alias.

You can activate opt-in Regions automatically during channel setup, by selecting them. Your channel becomes active in all selected Regions.

If you choose to select an opt-in Region as for your MediaPackage resources, enable it first by following the steps in Enabling a Region, when signed in to the AWS Management Console.

MediaPackage is available in the following AWS opt-in Regions:

- Middle East (UAE) Region, me-central-1
- Asia Pacific (Hyderabad) Region, ap-south-2
- Asia Pacific (Melbourne) Region, ap-southeast-4

Setting up MediaPackage

Before you start using AWS Elemental MediaPackage (MediaPackage), you must sign up for AWS (if you don't already have an AWS account) and create IAM users and roles to allow access to MediaPackage. This includes creating an IAM role for yourself. If you want to use encryption to protect your content, you also must store your encryption keys in AWS Secrets Manager, and then give MediaPackage permission to obtain the keys from your Secrets Manager account.

This section guides you through the steps required to configure users and roles to access MediaPackage. For background and additional information about identity and access management for MediaPackage, see the section called "Identity and Access Management".

Topics

Sign up for an AWS account

To get started with AWS, you need an AWS account. For information about creating an AWS account, see Getting started with an AWS account in the *AWS Account Management Reference Guide*.

Creating policies and non-administrative roles

By default, users and roles don't have permission to create or modify MediaPackage resources. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by MediaPackage, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for AWS Elemental MediaPackage in the *Service Authorization Reference*.

This section describes how you can create policies and create non-administrative roles so that users can create or modify MediaPackage resources. This section also describes how your users can assume that role to grant secure and temporary credentials.

Topics

(Optional) Step 1: Create an IAM policy for Amazon CloudFront

If you or your users will create Amazon CloudFront distributions from the AWS Elemental MediaPackage live console, create an IAM policy that allows access to CloudFront.

For more information about using CloudFront with MediaPackage, see Working with CDNs.

To use the JSON policy editor to create a policy

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 on the left, choose **Policies**.

   If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.
3. At the top of the page, choose **Create policy**.
4. In the **Policy editor** section, choose the **JSON** option.
5. Enter the following JSON policy document:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "cloudfront:GetDistribution",
                "cloudfront:CreateDistributionWithTags",
                "cloudfront:UpdateDistribution",
                "cloudfront:CreateDistribution",
                "cloudfront:TagResource",
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
}
```

6. Choose **Next**.

> **Note**
>
> You can switch between the **Visual** and **JSON** editor options anytime. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

7. On the **Review and create** page, enter a **Policy name** and a **Description** (optional) for the policy that you are creating. Review **Permissions defined in this policy** to see the permissions that are granted by your policy.
8. Choose **Create policy** to save your new policy.

(Optional) Step 2: Create an IAM policy for MediaPackage VOD

If you or your users will be using video on demand (VOD) functionality in MediaPackage, create an IAM policy that allows access to resources for the `mediapackage-vod` service.

The following sections describe how to create a policy that allows all actions, and one that allows read-only rights. You can customize the policies by adding or removing actions to fit your workflows.

Policy for full VOD access

This policy allows the user to perform all actions on all VOD resources.

To use the JSON policy editor to create a policy

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 on the left, choose **Policies**.

   If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.
3. At the top of the page, choose **Create policy**.
4. In the **Policy editor** section, choose the **JSON** option.
5. Enter the following JSON policy document:

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

6. Choose **Next**.

> **Note**
>
> You can switch between the **Visual** and **JSON** editor options anytime. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

7. On the **Review and create** page, enter a **Policy name** and a **Description** (optional) for the policy that you are creating. Review **Permissions defined in this policy** to see the permissions that are granted by your policy.
8. Choose **Create policy** to save your new policy.

Policy for read-only VOD access

This policy allows the user to view all VOD resources.

To use the JSON policy editor to create a policy

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 on the left, choose **Policies**.

   If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.
3. At the top of the page, choose **Create policy**.
4. In the **Policy editor** section, choose the **JSON** option.
5. Enter the following JSON policy document:

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

6. Choose **Next**.

> **Note**
>
> You can switch between the **Visual** and **JSON** editor options anytime. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

7. On the **Review and create** page, enter a **Policy name** and a **Description** (optional) for the policy that you are creating. Review **Permissions defined in this policy** to see the permissions that are granted by your policy.
8. Choose **Create policy** to save your new policy.

Step 3: Create a role in the IAM console

Create a role in the IAM console for each policy that you create. This allows users to assume a role rather than attaching individual policies to each user.

To create a role in the IAM console

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 of the IAM console, choose **Roles**, and then choose **Create role**.
3. Under **Select trusted entity**, choose **AWS account**.
4. Under **An AWS account**, select the account with the users that will be assuming this role.
   - If a third-party will be accessing this role, it's best practice to select **Require external ID**. For more information about external IDs, see Using an external ID for third-party access in the *IAM User Guide*.
   - It's best practice to require multi-factor authentication (MFA). You can select the check box next to **Require MFA**. For more information about MFA, see Multi-factor authentication (MFA) in the *IAM User Guide*.
5. Choose **Next**.
6. Under **Permissions policies**, search for and add the policy with the appropriate MediaPackage permissions level.
   - For access to live functionality, choose one of the following options:
     - Use **AWSElementalMediaPackageFullAccess** to allow the user to perform all actions on all live resources in MediaPackage.
     - Use **AWSElementalMediaPackageReadOnly** to provide the user read-only rights for all live resources in MediaPackage.
   - For access to video on demand (VOD) functionality, use the policy that you created in (Optional) Step 2: Create an IAM policy for MediaPackage VOD.

7. Add policies to allow the MediaPackage console to make calls to Amazon CloudWatch on the user's behalf. Without these policies, the user is able to use the service's API only (not the console). Choose one of the following options:

   - Use **ReadOnlyAccess** to allow MediaPackage to communicate with CloudWatch, and also provide the user read-only access to all AWS services on your account.
   - Use **CloudWatchReadOnlyAccess**, **CloudWatchEventsReadOnlyAccess**, and **CloudWatchLogsReadOnlyAccess** to allow MediaPackage to communicate with CloudWatch, and limit the user's read-only access to CloudWatch.

8. (Optional) If this user will create Amazon CloudFront distributions from the MediaPackage console, attach the policy that you created in (Optional) Step 1: Create an IAM policy for Amazon CloudFront.

9. (Optional) Set a permissions boundary. This is an advanced feature that is available for service roles, but not service-linked roles.

   1. Expand the **Permissions boundary** section and choose **Use a permissions boundary to control the maximum role permissions**. IAM includes a list of the AWS managed and customer managed policies in your account.
   2. Select the policy to use for the permissions boundary or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see Creating IAM policies in the *IAM User Guide*.
   3. After you create the policy, close that tab and return to your original tab to select the policy to use for the permissions boundary.

10. Verify that the correct policies are added to this group, and then choose **Next**.

11. If possible, enter a role name or role name suffix to help you identify the purpose of this role. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both **`PRODROLE`** and **`prodrole`**. Because various entities might reference the role, you cannot edit the name of the role after it has been created.

12. (Optional) For **Description**, enter a description for the new role.

13. Choose **Edit** in the **Step 1: Select trusted entities** or **Step 2: Select permissions** sections to edit the use cases and permissions for the role.

14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see Tagging IAM resources in the *IAM User Guide*.

15. Review the role and then choose **Create role**.

Step 4: Assume the role from the IAM console or AWS CLI

View the following resources for learning about granting permissions for users to assume the role and how users can switch to the role from the IAM console or AWS CLI.

- For more information about granting a user permissions to switch roles, see Granting a user permissions to switch roles in the *IAM User Guide*.
- For more information about switching roles (console), see Switching to a role (console) in the *IAM User Guide*.
- For more information about switching roles (AWS CLI), see Switching to an IAM role (AWS CLI) in the *IAM User Guide*.

Allowing AWS Elemental MediaPackage to access other AWS services

Some features require you to allow MediaPackage to access other AWS services, such as Amazon S3 and AWS Secrets Manager (Secrets Manager). To allow this access, create an IAM role and policy with the appropriate permissions. The following steps describe how to create roles and policies for MediaPackage features.

Topics

Step 1: Create a policy

The IAM policy defines the permissions that AWS Elemental MediaPackage (MediaPackage) requires to access other services.

- For video on demand (VOD) workflows, create a policy that allows MediaPackage to read from the Amazon S3 bucket, verify the billing method, and retrieve content. For the billing method, MediaPackage must verify that the bucket *does not* require the requester to pay for requests. If the bucket has **requestPayment** enabled, MediaPackage can't ingest content from that bucket.

- For live-to-VOD workflows, create a policy that allows MediaPackage to read from the Amazon S3 bucket and store the live-to-VOD asset in it.
- For content delivery network (CDN) authorization, create a policy that allows MediaPackage to read from a secret in Secrets Manager.

The following sections describe how to create these policies.

Policy for Amazon S3 access for VOD workflows

If you're using MediaPackage to ingest a VOD asset from an Amazon S3 bucket and to package and deliver that asset, you need a policy that allows you to do these things in Amazon S3:

- `GetObject` - MediaPackage can retrieve the VOD asset from the bucket.
- `GetBucketLocation` - MediaPackage can retrieve the Region for the bucket. The bucket must be in the same region as the MediaPackage VOD resources.
- `GetBucketRequestPayment` - MediaPackage can retrieve the payment request information. MediaPackage uses this information to verify that the bucket doesn't require the requester to pay for the content requests.

If you also use MediaPackage for live-to-VOD asset harvesting, add the `PutObject` action to the policy. For more information the required policy for live-to-VOD workflows, see Policy for live-to-VOD workflows.

To use the JSON policy editor to create a policy

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 on the left, choose **Policies**.

   If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.
3. At the top of the page, choose **Create policy**.
4. In the **Policy editor** section, choose the **JSON** option.
5. Enter the following JSON policy document:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:GetBucketRequestPayment",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::bucket_name/*",
                "arn:aws:s3:::bucket_name"
            ],
            "Effect": "Allow"
        }
    ]
}
```

6. Choose **Next**.

> **Note**
>
> You can switch between the **Visual** and **JSON** editor options anytime. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

7. On the **Review and create** page, enter a **Policy name** and a **Description** (optional) for the policy that you are creating. Review **Permissions defined in this policy** to see the permissions that are granted by your policy.
8. Choose **Create policy** to save your new policy.

Policy for live-to-VOD workflows

If you use MediaPackage to harvest a live-to-VOD asset from a live stream, you need a policy that allows you to do these things in Amazon S3:

- `PutObject`: MediaPackage can save the VOD asset in the bucket.
- `GetBucketLocation`: MediaPackage can retrieve the Region for the bucket. The bucket must be in the same AWS Region as the MediaPackage VOD resources.

If you also use MediaPackage for VOD asset delivery, add these actions to the policy: `GetObject` and `GetBucketRequestPayment`. For more information about the required policy for VOD workflows, see Policy for Amazon S3 access for VOD workflows.

To use the JSON policy editor to create a policy

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 on the left, choose **Policies**.

   If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.
3. At the top of the page, choose **Create policy**.
4. In the **Policy editor** section, choose the **JSON** option.
5. Enter the following JSON policy document:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::bucket_name/*",
                "arn:aws:s3:::bucket_name"
            ],
            "Effect": "Allow"
        }
    ]
}
```

6. Choose **Next**.

> **Note**
>
> You can switch between the **Visual** and **JSON** editor options anytime. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to

optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

7. On the **Review and create** page, enter a **Policy name** and a **Description** (optional) for the policy that you are creating. Review **Permissions defined in this policy** to see the permissions that are granted by your policy.
8. Choose **Create policy** to save your new policy.

Policy for Secrets Manager access for CDN authorization

If you use content delivery network (CDN) authorization headers to restrict access to your endpoints in MediaPackage, you need a policy that allows you to do these things in Secrets Manager:

- `GetSecretValue` - MediaPackage can retrieve the encrypted authorization code from a version of the secret.
- `DescribeSecret` - MediaPackage can retrieve the details of the secret, excluding encrypted fields.
- `ListSecrets` - MediaPackage can retrieve a list of secrets in the AWS account.
- `ListSecretVersionIds`: MediaPackage can retrieve all of the versions that are attached to the specified secret.

Note

You don't need a separate policy for each secret that you store in Secrets Manager. If you create a policy like the one described in the following procedure, MediaPackage can access all secrets in your account in this Region.

To use the JSON policy editor to create a policy

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 on the left, choose **Policies**.

If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.

3. At the top of the page, choose **Create policy**.
4. In the **Policy editor** section, choose the **JSON** option.
5. Enter the following JSON policy document:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret",
        "secretsmanager:ListSecrets",
        "secretsmanager:ListSecretVersionIds"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:account-id:secret:secret-name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
         "iam:GetRole",
         "iam:PassRole"
       ],
       "Resource": "arn:aws:iam::account-id:role/role-name"
     }
  ]
}
```

6. Choose **Next**.

> **Note**
>
> You can switch between the **Visual** and **JSON** editor options anytime. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to

optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

7. On the **Review and create** page, enter a **Policy name** and a **Description** (optional) for the policy that you are creating. Review **Permissions defined in this policy** to see the permissions that are granted by your policy.
8. Choose **Create policy** to save your new policy.

Step 2: Create a role

An IAM role is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. Create a role that AWS Elemental MediaPackage assumes when ingesting source content from Amazon S3.

When you create the role, you choose Amazon Elastic Compute Cloud (Amazon EC2) as the trusted entity that can assume the role because MediaPackage isn't available for selection. In Step 3: Modify the trust relationship, you change the trusted entity to MediaPackage.

For information about creating a service role, see Creating a Role to Delegate Permissions to an AWS Service in the *IAM User Guide*.

Step 3: Modify the trust relationship

The trust relationship defines what entities can assume the role that you created in the section called “Step 2: Create a role”. When you created the role and established the trusted relationship, you chose Amazon EC2 as the trusted entity. Modify the role so that the trusted relationship is between your AWS account and AWS Elemental MediaPackage.

To change the trust relationship to MediaPackage

1. Access the role that you created in Step 2: Create a role.

   If you're not already displaying the role, in the navigation pane of the IAM console, choose **Roles**. Search for and choose the role that you created.

2. On the **Summary** page for the role, choose **Trust relationships**.
3. Choose **Edit trust relationship**.
4. On the **Edit Trust Relationship** page, in the **Policy Document**, change `ec2.amazonaws.com` to `mediapackage.amazonaws.com`.

   The policy document should now look like this:

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "mediapackage.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

   If you're using MediaPackage and related services in an opt-in Region, the Region must be listed in the `Service` section of the policy document. For example, if you're using services in the Asia Pacific (Melbourne) Region, the policy document looks like this:

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "mediapackage.amazonaws.com",
          "mediapackage.ap-southeast-4.amazonaws.com"
```

```
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

5. Choose **Update Trust Policy**.
6. On the **Summary** page, make a note of the value in **Role ARN**. You use this ARN when you ingest source content for video on demand (VOD) workflows. The ARN looks like this:

   `arn:aws:iam::111122223333:role/role-name`

   In the example, *111122223333* is your AWS account number.

(Optional) Setting up encryption

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

To encrypt content, you must have a DRM solution provider and be set up to use encryption. For more information, see the section called “Content encryption and DRM”.

(Optional) Installing the AWS CLI

To use the AWS CLI with AWS Elemental MediaPackage, install the latest AWS CLI version. For information about installing the AWS CLI or upgrading it to the latest version, see Installing the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.

IPv6 support for AWS Elemental MediaPackage control plane

AWS Elemental MediaPackage control plane APIs support dual-stack (IPv4 and IPv6) endpoints for both Live and VOD services. This enables you to make API requests using either IPv4 or IPv6 protocols for management operations such as creating channels, endpoints, packaging groups, packaging configurations, and assets.

> **Note**
>
> IPv6 support applies to control plane operations only. Content ingest and delivery endpoints continue to use IPv4.

IPv6 endpoints

MediaPackage provides the following dual-stack endpoints:

- **Live**: `mediapackage.region.api.aws`
- **VOD**: `mediapackage-vod.region.api.aws`

Your existing applications continue to work with the original IPv4-only endpoints (`mediapackage.region.amazonaws.com` and `mediapackage-vod.region.amazonaws.com`). For a complete list of available endpoints, see MediaPackage endpoints and quotas in the *AWS General Reference*.

Using IPv6 endpoints

To use the IPv6 endpoints, specify the dual-stack endpoint URL when making API calls.

AWS CLI example (Live):

```
aws mediapackage list-channels \
    --endpoint-url https://mediapackage.us-east-1.api.aws
```

AWS CLI example (VOD):

```
aws mediapackage-vod list-packaging-groups \
    --endpoint-url https://mediapackage-vod.us-east-1.api.aws
```

SDK example (Python):

```
import boto3

# Live
live_client = boto3.client(
    'mediapackage',
    endpoint_url='https://mediapackage.us-east-1.api.aws'
)

# VOD
vod_client = boto3.client(
    'mediapackage-vod',
    endpoint_url='https://mediapackage-vod.us-east-1.api.aws'
)
```

Getting started with AWS Elemental MediaPackage

The following sections describe how to quickly get started receiving and sending content with AWS Elemental MediaPackage.

Topics

Getting started with live content delivery in AWS Elemental MediaPackage

This Getting Started tutorial shows you how to use the AWS Elemental MediaPackage (MediaPackage) console to create a channel and endpoints for streaming live videos.

Prerequisites

Before you can use MediaPackage, you need an AWS account and the appropriate permissions to access, view, and edit MediaPackage components. Make sure that your system administrator has completed the steps in *Setting up*, and then return to this tutorial.

For supported live inputs and codecs, see Live supported codecs and input types.

Step 1: Access MediaPackage

Using your IAM credentials, sign in to the MediaPackage console:

```
https://console.aws.amazon.com/mediapackage/
```

Step 2: Create a channel

The channel is the first component in MediaPackage. It represents the input to MediaPackage for incoming live content from an encoder such as AWS Elemental MediaLive.

MediaPackage does not require that you supply any customer data. There are no fields in channels where there is an expectation that you will provide customer data.

To create a channel

1. On the MediaPackage **Channels** page, choose **Create channel**.
2. For **ID**, enter a name that describes the channel, such as **`channel1HLS1`**. The ID is the primary identifier for the channel, and must be unique for your account in the AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
3. Keep the defaults for the remaining fields, and then choose **Create**.

   MediaPackage displays the new channel's details page.
4. On the details page for the channel, note the values for **URL**, **Username**, and **Password**. If you're using input redundancy, you need this information for both input URLs. If you're sending only one stream to the channel, you can note the information for either input URL.

   MediaPackage securely generates the WebDAV user names and passwords when it creates the channel. If you need to change these credentials, see Rotating credentials on an input URL.

   Provide the information from these fields to the person in charge of the upstream encoder. In the stream configuration in the encoder, this person must enter the destination as the input URL, and the WebDAV credentials as the channel's user name and password. The upstream encoder must use digest authentication and push WebDAV over HTTPS to MediaPackage, and include these credentials. If you're using input redundancy, the input streams to this channel must have identical encoder settings. For more information about setting up source streams for input redundancy, see Live input redundancy AWS Elemental MediaPackage processing flow.

Step 3: Create endpoints

The endpoint is attached to a channel, and represents the output of the live content. You can associate multiple endpoints to a single channel. Each endpoint gives players and downstream CDNs (such as Amazon CloudFront) access to the content for playback.

MediaPackage does not require that you supply any customer data. There are no fields in endpoints where there is an expectation that you will provide customer data.

To create an endpoint

1. On the **Channels** page, choose the channel that the endpoint will be associated with.
2. On the details page for the channel, under **Origin endpoints**, choose **Manage endpoints**.

3. For **ID**, enter a name that describes the endpoint, such as **`HLSendpoint1`**. The ID is the primary identifier for the endpoint, and must be unique for your account in the AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
4. Keep the defaults for the remaining fields, and then choose **Save**.

   MediaPackage displays the channel's details page, including the endpoint that you just created.
5. On the details page for the channel, note the value in the **URL** field for the endpoint. Provide this information to the person in charge of the downstream device (CDN or player). In the downstream device, this person must enter the request destination as the endpoint's URL.

(Optional) Step 4: Monitor MediaPackage activity

Use Amazon CloudWatch to track MediaPackage activity, such as the counts of bytes that MediaPackage has received and sent, response times, and request counts. Metrics are grouped first by the service namespace, and then by the various dimension combinations within each namespace.

To view metrics using the CloudWatch console

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. Under **All metrics**, choose the **AWS/MediaPackage** namespace.
4. Select the metric dimension to view the metrics (for example, choose `channel` to view metrics per channel).

For a list of MediaPackage metrics, see AWS Elemental MediaPackage live content metrics.

Step 5: Clean up

To avoid extraneous charges, be sure to delete all unnecessary channels and endpoints. You must delete all endpoints on a channel before the channel can be deleted.

To delete an endpoint

1. On the MediaPackage **Channels** page, choose the channel that the endpoint is associated with.

2. On the details page for the channel, under **Origin endpoints**, select the origin endpoint that you want to delete.
3. Select **Delete**.
4. In the **Delete endpoints** confirmation dialog box, choose **Delete**.

To delete a channel

1. On the **Channels** page, choose the channel you want to delete.
2. Choose **Delete**.
3. In the **Channel delete** confirmation dialog box, choose **Delete**.

   MediaPackage removes the channel and all associated endpoints.

Getting started with live-to-VOD content delivery in MediaPackage

This Getting Started tutorial shows you how to use the AWS Elemental MediaPackage console to create a live-to-VOD (video on demand) asset and make it available for playback.

To deliver live-to-VOD content, you do these three main things:

- Ingest a live HLS content stream into MediaPackage
- Extract a VOD asset from the stream
- Make the asset available for playback

> **Note**
>
> You're not required to use MediaPackage to deliver your live-to-VOD asset to viewers. This tutorial is meant as an illustration of how you can use MediaPackage to complete the live-to-VOD workflow.

The following sections are a guided tutorial for becoming familiar with these three things and other supporting actions.

Prerequisites

Before you can use AWS Elemental MediaPackage, you need an AWS account and the appropriate permissions to access, view, and edit MediaPackage components. Make sure that your system administrator has completed the following steps in *Setting up*, and then return to this tutorial:

- To create an AWS account, see ???.
- To allow non-administrative roles access to MediaPackage, see Creating policies and non-administrative roles.
- To allow MediaPackage to access your Amazon S3 bucket to save and retrieve the live-to-VOD asset, see Allowing AWS Elemental MediaPackage to access other AWS services.

Step 1: Access MediaPackage

Using your IAM credentials, sign in to the AWS Elemental MediaPackage console:

```
https://console.aws.amazon.com/mediapackage/
```

Step 2: Ingest live content

To ingest a live content stream into AWS Elemental MediaPackage and extract a video on demand (VOD) asset from it, create a channel and endpoint. The channel is the entry point to MediaPackage, and the endpoint provides MediaPackage access to the stream so that it can extract the VOD asset. The following sections describe how to use the MediaPackage console to create a channel and endpoint.

Create a channel

The channel is the first component in MediaPackage. It represents the input to MediaPackage for incoming live content from an encoder such as AWS Elemental MediaLive.

MediaPackage does not require that you supply any customer data. There are no fields in channels where there is an expectation that you will provide customer data.

To create a channel

1. On the MediaPackage **Channels** page, choose **Create channel**.

2. For **ID**, enter a name that describes the channel, such as **`channel1HLS1`**. The ID is the primary identifier for the channel, and must be unique for your account in the AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
3. Keep the defaults for the remaining fields, and then choose **Create**.

   MediaPackage displays the new channel's details page.
4. On the details page for the channel, note the values for **URL**, **Username**, and **Password**. If you're using input redundancy, you need this information for both input URLs. If you're sending only one stream to the channel, you can note the information for either input URL.

   MediaPackage securely generates the WebDAV user names and passwords when it creates the channel. If you need to change these credentials, see Rotating credentials on an input URL.

   Provide the information from these fields to the person in charge of the upstream encoder. In the stream configuration in the encoder, this person must enter the destination as the input URL, and the WebDAV credentials as the channel's user name and password. The upstream encoder must use digest authentication and push WebDAV over HTTPS to MediaPackage, and include these credentials. If you're using input redundancy, the input streams to this channel must have identical encoder settings. For more information about setting up source streams for input redundancy, see Live input redundancy AWS Elemental MediaPackage processing flow.

Create an endpoint

The endpoint is attached to a channel, and represents the output of the live content. When you create a harvest job to extract a VOD asset from the live content, you have to indicate what endpoint you're extracting from. You can harvest assets from clear (unencrypted) or encrypted HLS and DASH endpoints, and the endpoint must have a startover window defined. If you have only encrypted endpoints, see the Creating live-to-VOD assets with AWS Elemental MediaPackage feature reference.

MediaPackage does not require that you supply any customer data. There are no fields in endpoints where there is an expectation that you will provide customer data.

To create an endpoint

1. On the **Channels** page, choose the channel that the endpoint will be associated with.
2. On the details page for the channel, under **Origin endpoints**, choose **Manage endpoints**.

3. For **ID**, enter a name that describes the endpoint, such as **`HLSendpoint1`**. The ID is the primary identifier for the endpoint, and must be unique for your account in the AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
4. Keep the defaults for the remaining fields, and then choose **Save**.

   MediaPackage displays the channel's details page, including the endpoint that you just created.
5. On the details page for the channel, note the value in the **URL** field for the endpoint. Provide this information to the person in charge of the downstream device (CDN or player). In the downstream device, this person must enter the request destination as the endpoint's URL.

Step 3: Extract a VOD asset

To extract a live-to-VOD asset from a live content stream, create a harvest job. The harvest job identifies what endpoint the asset is being harvested from, the start and end of the asset, and where MediaPackage saves the asset after it's been harvested.

To create a harvest job

1. On the **Harvest jobs** page, choose **Create harvest job**.
2. For **ID**, enter a name that describes the harvest job, such as **`gamehighlights`**. The ID is the primary identifier for the job. You can reuse the ID after the harvest job expires from your account. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
3. For **Origin endpoint**, select the endpoint for the live content stream that you're extracting a VOD asset from. The endpoint must serve clear (unencrypted) or encrypted DASH or HLS content. If you want to extract from encrypted live content, see Creating live-to-VOD assets with AWS Elemental MediaPackage.
4. For **Date and time format**, keep the default.
5. For **When the live-to-VOD asset begins** and **When the live-to-VOD asset ends**, enter the start and end dates and times for the extracted VOD asset. We recommend that the start time be after the live stream has started and before the current time ("now"). The end time must be in the past.

> **Note**
>
> "Now" is the current time according to the `EXT-X-PROGRAM-DATE-TIME`, when it's present in the source content from the encoder. Therefore, we recommend that the upstream encoder provides an `EXT-X-PROGRAM-DATE-TIME` tag in the source.

6. For **IAM role ARN**, enter the IAM role that allows MediaPackage to write your live-to-VOD asset to your Amazon S3 bucket. For help with the role, see Allowing AWS Elemental MediaPackage to access other AWS services.
7. For **Amazon S3 bucket name**, select the Amazon S3 bucket where you want MediaPackage to store the live-to-VOD asset.
8. For **Manifest key**, enter the path in the Amazon S3 bucket and identifier for the parent manifest for the live-to-VOD asset. MediaPackage creates a directory based on the path that you enter.

   > **Important**
   >
   > The manifest key must be unique. When you use the same manifest key for multiple harvest jobs, the newest playlist for the asset overwrites existing playlists. The only time you should reuse a manifest key is when you are harvesting the same content, such as if there was a problem with a previous harvest of the content.

9. Choose **Create**.

When MediaPackage processes the harvest job, it sends a CloudWatch event when the job fails or succeeds. The event includes the details of the harvest job. If the job fails, the event includes information about why. This information is available only in the CloudWatch event. For example events, see Harvest job notification events.

(Optional) Step 4: Output VOD content

To use MediaPackage to make the live-to-VOD asset available for playback, create a packaging group, packaging configuration, and asset resource. The asset ingests the live-to-VOD asset from the Amazon S3 bucket. A packaging group holds one or more packaging configurations, which define the output format and settings.

Create a packaging group

A packaging group holds one or more packaging configurations. The packaging configurations enable you to define what kind of VOD outputs you want. To apply these output definitions, associate a packaging group to multiple assets.

Example

You have 15 pieces of source content. You want to serve them all as DASH, HLS, and encrypted HLS outputs. To do this, you define one packaging group with DASH, HLS, and encrypted HLS packaging configurations. You then associate that group to the asset resources that represent these pieces of content. You don't have to create new configurations for each asset.

MediaPackage doesn't require that you supply any customer data. There are no fields in packaging groups where there is an expectation that you will provide customer data.

To create a packaging group

1. On the **Packaging groups** page, choose **Create group**.
2. For **ID**, enter a name that describes the group, such as **`gamehighlights`**. The ID is the primary identifier for the group, and must be unique for your account in this AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
3. Choose **Create**.

Create a packaging configuration

A packaging configuration specifies how the output manifest is configured, such as stream selection limitations and ordering.

MediaPackage does not require that you supply any customer data. There are no fields in packaging configurations where there is an expectation that you will provide customer data.

To create a packaging configuration

1. On the **Packaging groups** page, choose the group that you just created.
2. On the details page for the packaging group, under **Packaging configurations** choose **Manage configurations**.
3. On the **Manage packaging configurations** page, choose **Add**, and then choose **New configuration**.

4. For **ID**, enter a name that describes the configuration, such as **`hls_highlights`**. The ID is the primary identifier for the configuration, and must be unique for your account in this AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
5. Keep the defaults for the remaining fields, and then choose **Save**.

Create an asset

An asset resource is how AWS Elemental MediaPackage ingests, packages, and serves VOD content. The asset is associated with one or more packaging configurations. Downstream devices send playback requests to specific packaging configurations on the asset.

MediaPackage doesn't require customer data from you, so assets don't include those fields.

To create an asset and ingest source content

1. From your Amazon S3 buckets, determine what file you're using as source content. Make note of the following:
   - The name of the Amazon S3 bucket where the file is stored
   - The full path for the file, such as *S3://bucket/path/source-file-name*
   - The IAM role that allows MediaPackage to read from Amazon S3
2. On the MediaPackage console, go to the **Assets** page, and then choose **Ingest assets**.
3. For **Amazon S3 bucket name**, choose the bucket where your source content is stored.
4. For **IAM role**, choose **Use existing role** and select the IAM role that allows MediaPackage to read from Amazon S3.
5. For **Filename**, enter the full path to either the .smil manifest (MP4) or the .m3u8 parent playlist (HLS) within your Amazon S3 bucket, including the name of the source content. For example, if your content is called `lion_movie.m3u8` and is in a subdirectory called `thursday_night` in a bucket called `movies`, you would enter the following in the **Filename** field:

   ```
   thursday_night/lion_movie.m3u8
   ```

   You don't need to enter the bucket name because you chose it in **Amazon S3 bucket name** field.

6. For **Packaging group**, choose the group that you created in Create a packaging group.
7. Choose **Ingest assets**.

Provide playback URLs

After creating the asset resource, AWS Elemental MediaPackage prepares to serve the packaged manifests to viewers. This happens in the background and might take some time depending on the size and complexity of the source content, but is usually less than a few minutes. The URLs of the manifests are available immediately on the asset's details page, but content is not yet available for playback.

After the processing for each manifest is complete, MediaPackage sends an Amazon CloudWatch event to your account.

On the asset, MediaPackage provides a URL for each packaging configuration. This URL is how downstream devices (CDN or playback device) request VOD content from MediaPackage.

To get playback URLs

1. On the MediaPackage console, go to the **Assets** page and choose the **ID** of the asset that you created in Step 4: Create an asset.
2. On the asset's detail page, get the URL for each packaging configuration.
3. Provide the URLs to the person in charge of the downstream device (CDN or player). In the downstream device, this person must enter the request destination as the URL from the corresponding packaging configuration.

Each URL is stable. It never changes during the lifetime of the combination of this asset and packaging configuration. Provide the URL to the person in charge of the downstream device (CDN or player). In the downstream device, this person must use the asset's URL as the request destination.

(Optional) Step 5: Monitor MediaPackage activity

Use Amazon CloudWatch to track MediaPackage activity, such as the counts of bytes that MediaPackage has received and sent, response times, and request counts. Metrics are grouped first by the service namespace, and then by the various dimension combinations within each namespace.

To view metrics using the CloudWatch console

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. Under **All metrics**, choose the **AWS/MediaPackage** namespace.
4. Select the metric dimension to view the metrics (for example, choose `channel` to view metrics per channel).

For a list of MediaPackage metrics, see Monitoring AWS Elemental MediaPackage with Amazon CloudWatch metrics.

Step 6: Clean up

To avoid incurring extra charges, delete the resources that you're no longer using.

> **Note**
>
> Harvest jobs automatically expire off your account and can't be manually deleted.

Delete live resources

When you're done ingesting, serving, and harvesting from live content, delete the channel and endpoint. You must delete all endpoints on a channel before you can delete the channel.

To delete an endpoint

1. On the MediaPackage **Channels** page, choose the channel that the endpoint is associated with.
2. On the details page for the channel, under **Origin endpoints**, select the origin endpoint that you want to delete.
3. Select **Delete**.
4. In the **Delete endpoints** confirmation dialog box, choose **Delete**.

To delete a channel

1. On the **Channels** page, choose the channel you want to delete.

2. Choose **Delete**.
3. In the **Channel delete** confirmation dialog box, choose **Delete**.

   MediaPackage removes the channel and all associated endpoints.

Delete VOD resources

When you're done ingesting and serving VOD content, delete the extra resources. If you want to make a specific output unavailable, delete the packaging configuration from the packaging group. If you want to make an asset no longer available for playback from any outputs, delete the asset.

To delete an asset

1. On the MediaPackage console, go to the **Assets** page, and then choose the **ID** of the asset.
2. On the asset's details page, choose **Delete**.
3. In the confirmation dialog box, choose **Delete**.

To delete a packaging configuration

1. On the MediaPackage console, go to the **Packaging groups** page.
2. Choose the **ID** of the group that has the configuration that you want to delete.
3. On the packaging group's details page, in the **Packaging configurations** section, locate the configuration and choose its **ID**.
4. On the packaging configuration's details page, choose **Delete**.
5. In the confirmation dialog box, choose **Delete**.

Getting started with VOD content delivery in MediaPackage

This Getting Started tutorial shows you how to use the AWS Elemental MediaPackage console to ingest video on demand (VOD) content and make it available for playback.

Prerequisites

Before you can use AWS Elemental MediaPackage VOD capability, you must meet the following conditions:

- You have an AWS account and the appropriate permissions to access, view, and edit MediaPackage components. Make sure that your system administrator has completed the steps in *Setting up*, and then return to this tutorial.
- You have file-based source content in one or more Amazon S3 buckets.

For supported VOD inputs and codecs, see VOD supported codecs and input types.

Step 1: Access MediaPackage

Using your IAM credentials, sign in to the AWS Elemental MediaPackage console:

```
https://region.console.aws.amazon.com/mediapackage/home
```

Step 2: Create a packaging group

A packaging group holds one or more packaging configurations. The packaging configurations enable you to define what kind of VOD outputs you want. To apply these output definitions, associate a packaging group to multiple assets.

Example

You have 15 pieces of source content. You want to serve them all as DASH, HLS, and encrypted HLS outputs. To do this, you define one packaging group with DASH, HLS, and encrypted HLS packaging configurations. You then associate that group to the asset resources that represent these pieces of content. You don't have to create new configurations for each asset.

MediaPackage doesn't require that you supply any customer data. There are no fields in packaging groups where there is an expectation that you will provide customer data.

To create a packaging group

1. On the **Packaging groups** page, choose **Create group**.
2. For **ID**, enter a name that describes the group, such as **`gamehighlights`**. The ID is the primary identifier for the group, and must be unique for your account in this AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
3. Choose **Create**.

Step 3: Create a packaging configuration

A packaging configuration specifies how the output manifest is configured, such as stream selection limitations and ordering.

MediaPackage does not require that you supply any customer data. There are no fields in packaging configurations where there is an expectation that you will provide customer data.

To create a packaging configuration

1. On the **Packaging groups** page, choose the group that you just created.
2. On the details page for the packaging group, under **Packaging configurations** choose **Manage configurations**.
3. On the **Manage packaging configurations** page, choose **Add**, and then choose **New configuration**.
4. For **ID**, enter a name that describes the configuration, such as **`hls_highlights`**. The ID is the primary identifier for the configuration, and must be unique for your account in this AWS Region. Supported characters are letters, numbers, underscores (_), and dashes (-). You can't use spaces in the ID.
5. Keep the defaults for the remaining fields, and then choose **Save**.

Step 4: Create an asset

An asset resource is how MediaPackage ingests, packages, and serves VOD content. The asset is associated with one or more packaging configurations. Downstream devices send playback requests to specific packaging configurations on the asset.

MediaPackage doesn't require customer data from you, so assets don't include those fields.

To create an asset

1. From your Amazon S3 buckets, determine what file you're using as source content. Make note of the following:

   - The name of the Amazon S3 bucket where the file is stored
   - The full path for the file, such as *S3://bucket/path/source-file-name*
   - The IAM role that allows MediaPackage to read from Amazon S3

2. On the MediaPackage console, go to the **Assets** page, and then choose **Ingest assets**.
3. For **Amazon S3 bucket name**, choose the bucket where your source content is stored.
4. For **IAM role**, choose **Use existing role** and select the IAM role that allows MediaPackage to read from Amazon S3.
5. For **Filename**, enter the full path to either the .smil manifest (MP4) or the .m3u8 parent playlist (HLS) within your Amazon S3 bucket, including the name of the source content. You don't need to enter the bucket name because you chose it in **Amazon S3 bucket name** field. For example, if your content is called `lion_movie.m3u8` and is in a subdirectory called `thursday_night` in a bucket called `movies`, you would enter the following in the **Filename** field:

   ```
   thursday_night/lion_movie.m3u8
   ```

   For more information about using .smil manifests with MediaPackage, see Requirements for .smil manifests.
6. For **Packaging group**, choose the group that you created in Step 2: Create a packaging group.
7. Choose **Ingest assets**.

Step 5: Provide playback URLs

After creating the asset resource, AWS Elemental MediaPackage prepares to serve the packaged manifests to viewers. This happens in the background and might take some time depending on the size and complexity of the source content, but is usually less than a few minutes. The URLs of the manifests are available immediately on the asset's details page, but content is not yet available for playback.

After the processing for each manifest is complete, MediaPackage sends an Amazon CloudWatch event to your account.

On the asset, MediaPackage provides a URL for each packaging configuration. This URL is how downstream devices (CDN or playback device) request VOD content from MediaPackage.

To get playback URLs

1. On the MediaPackage console, go to the **Assets** page and choose the **ID** of the asset that you created in Step 4: Create an asset.
2. On the asset's detail page, get the URL for each packaging configuration.

3. Provide the URLs to the person in charge of the downstream device (CDN or player). In the downstream device, this person must enter the request destination as the URL from the corresponding packaging configuration.

Each URL is stable. It never changes during the lifetime of the combination of this asset and packaging configuration. Provide the URL to the person in charge of the downstream device (CDN or player). In the downstream device, this person must use the asset's URL as the request destination.

(Optional) Step 6: Monitor MediaPackage activity

Use Amazon CloudWatch to track MediaPackage activity, such as the counts of bytes that MediaPackage has received and sent, response times, and request counts. Metrics are grouped first by the service namespace, and then by the various dimension combinations within each namespace.

To view metrics using the CloudWatch console

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. Under **All metrics**, choose the **AWS/MediaPackage** namespace.
4. Select the metric dimension to view the metrics (for example, choose `channel` to view metrics per channel).

For a list of MediaPackage metrics, see AWS Elemental MediaPackage VOD content metrics.

Step 7: Clean up

To avoid incurring extra charges, delete your VOD resources. If you want to make a specific output unavailable, delete the packaging configuration from the packaging group. If you want to make an asset no longer available for playback from any outputs, delete the asset.

To delete an asset

1. On the MediaPackage console, go to the **Assets** page, and then choose the **ID** of the asset.
2. On the asset's details page, choose **Delete**.
3. In the confirmation dialog box, choose **Delete**.

To delete a packaging configuration

1. On the MediaPackage console, go to the **Packaging groups** page.
2. Choose the **ID** of the group that has the configuration that you want to delete.
3. On the packaging group's details page, in the **Packaging configurations** section, locate the configuration and choose its **ID**.
4. On the packaging configuration's details page, choose **Delete**.
5. In the confirmation dialog box, choose **Delete**.

Delivering live content from AWS Elemental MediaPackage

AWS Elemental MediaPackage uses the following resources for live content:

- *Channels* are the entry point for your live streams from upstream encoders.

  For supported live inputs and codecs, see Live supported codecs and input types.
- *Endpoints* tell MediaPackage how to package outbound content. Endpoints are associated with channels and hold encryption, stream, and packaging settings.

The following sections describe how to use these resources to manage live content in MediaPackage.

Topics

Working with channels in AWS Elemental MediaPackage

A channel holds all the information that AWS Elemental MediaPackage (MediaPackage) requires to receive a live content stream from a source such as AWS Elemental MediaLive or another encoder. The channel receives content, and after packaging it, outputs it through an endpoint to downstream devices (such as video players or CDNs) that request the content.

After you create a channel, MediaPackage provides a pair of input URLs that are fixed for the lifetime of the channel, regardless of any failures or upgrades that might happen over time. The output of the upstream encoder points to the URLs for stream delivery to MediaPackage.

For supported live inputs and codecs, see Live supported codecs and input types.

Topics

- Rotating credentials on an input URL
- Deleting a channel
- Adding an endpoint to a channel

Creating a channel

Create a channel to start receiving content streams. Later, you add an endpoint to the channel. This endpoint is the access point for content playback requests.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to create a channel. For information about creating a channel through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

When you're creating a channel, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with MediaPackage using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

To create a channel (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Live**, choose **Channels**.
3. On the **Channels** page, choose **Create channel**.
4. For **ID**, type a name that describes the channel. The ID is the primary identifier for the channel, and must be unique for your account in the region.
5. (Optional) For **Description**, enter any descriptive text that helps you to identify the channel.
6. For **Input type**, choose **Apple HLS**.
7. Choose **Create**.

   MediaPackage displays the new channel's details page.

   The channel is active and can start receiving content as soon as it's created. MediaPackage scales resources up and down to allow the right amount of capacity for your traffic. If you're using input redundancy and one of the inputs stops sending content, then MediaPackage automatically switches to the other input for the source content. For more information about

how input redundancy works, see Live input redundancy AWS Elemental MediaPackage processing flow.

When you're creating a channel, you will receive an error if you exceed the quotas on the account. An error similar to Too many requests, please try again. Resource limit exceeded means that either you have exceeded the API request quotas, or you have already reached the maximum number of channels allowed on your account. If this is your first channel, or if you think you received this error wrongfully, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

Viewing channel details

View all channels that are configured in AWS Elemental MediaPackage, or view the details of a specific channel, including the endpoints that are associated with it.

You can use the MediaPackage console, the AWS CLI, or the MediaPackage API to view channel details. For information about viewing details about a channel through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To view channels (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. If the **Channels** page doesn't appear, on the MediaPackage home page, choose **Skip and go to console**.

   All existing channels are displayed on the console.
3. (Optional) Choose **Preferences** to adjust your viewing preferences (such as page size and properties that are displayed).
4. To view more information about a specific channel, choose the name of the channel that you want to view.

   MediaPackage displays important information such as the values for **Input URL** and the WebDAV **Username** and **Password** for each input URL. Provide this information for the upstream encoder stream destination settings. If you're using input redundancy, provide the information for both input URLs. If you're sending only one stream to the channel, you can provide the information for either input URL. For information about how input redundancy works, see Live input redundancy AWS Elemental MediaPackage processing flow.

> **Note**
>
> All channels have two input URLs. For channels that existed before input redundancy, MediaPackage created two new input URLs. You can use either the old or new URLs for inputs to the channel. The parent manifest should be named **`channel.m3u8`**.

If you created an Amazon CloudFront distribution from the MediaPackage console, you will also see the high-level distribution information (such as status and ID) from the channel. When you add an endpoint in MediaPackage, an origin is also added to the distribution, and you will see the CloudFront CDN URL from the channel's details page as well.

Editing a channel

Edit a channel's description for easier identification later.

You can edit the description on a channel or enable Amazon CloudFront distribution creation from the AWS Elemental MediaPackage console.

> **Note**
>
> To make changes to an existing distribution (even if it was created from MediaPackage), go to the Amazon CloudFront console.

You can use the MediaPackage console, the AWS CLI, or the MediaPackage API to edit a channel. For information about editing a channel through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

When you're editing a channel, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with MediaPackage using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

To edit a channel (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.

2. If the **Channels** page doesn't appear, on the MediaPackage home page, choose **Skip and go to console**.
3. On the **Channels** page, choose the name of the channel that you want to edit.
4. On the channel's details page, choose **Edit**.
5. Make the changes that you want.
6. Choose **Update**.

Rotating credentials on an input URL

Rotate credentials on an input URL to generate a new WebDAV user name and password.

You can use the AWS Elemental MediaPackage console or the MediaPackage API to rotate credentials. For information about rotating credentials through the MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To rotate credentials (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. If the **Channels** page doesn't appear, on the MediaPackage home page, choose **Skip and go to console**.
3. On the **Channels** page, choose the name of the channel that holds the input URL that you're rotating the credentials for.
4. On the channel's details page, choose the input URL that you're rotating credentials for, and then choose **Rotate credentials**.
5. To confirm that you want to generate a new user name and password, choose **Rotate**.

   MediaPackage displays the new credentials.

Deleting a channel

Delete a channel to stop AWS Elemental MediaPackage from receiving further content. You must delete the channel's endpoints (as described in Deleting an endpoint) before you can delete the channel.

You can use the MediaPackage console, the AWS CLI, or the MediaPackage API to delete a channel. For information about deleting a channel through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To delete a channel (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. If the **Channels** page doesn't appear, on the MediaPackage home page, choose **Skip and go to console**.
3. On the **Channels** page, choose the name of the channel that you want to delete.
4. Choose **Delete**.

   If there's an Amazon CloudFront distribution associated with the channel, select the CloudFront link in the confirmation dialog box to go to the CloudFront console to delete the distribution. MediaPackage will not delete the distribution when the channel is deleted. For help deleting in CloudFront, see Deleting a distribution in the *Amazon CloudFront Developer Guide*.
5. In the confirmation dialog box in MediaPackage, choose **Delete** to proceed with the channel deletion.

Adding an endpoint to a channel

Add an endpoint to a channel to allow downstream video players and content delivery networks (CDNs) to start requesting content playback.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to add an endpoint to a channel. For information about adding through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

For instructions on adding endpoints to a channel from the MediaPackage console, see the section called “Working with endpoints”.

Working with endpoints in AWS Elemental MediaPackage

An endpoint defines a single delivery point of a channel. The endpoint holds all the information that's needed for AWS Elemental MediaPackage to integrate with a player or content delivery network (CDN) such as Amazon CloudFront. Configure the endpoint to output content in one of the available stream formats:

- Apple HLS – Packages content to Apple HTTP Live Streaming (HLS)
- Microsoft Smooth Streaming – Packages content for Microsoft Smooth Streaming players
- DASH-ISO – Packages content for the DASH-ISO ABR streaming protocol
- CMAF – Packages content to devices that support Apple HLS fragmented MP4 (fMP4)

Additionally, the endpoint holds information about digital rights management (DRM) and encryption integration, stream bitrate presentation order, and more.

Topics

Creating an endpoint

Create an endpoint on a channel to define how AWS Elemental MediaPackage prepares content for delivery. Content can't be served from a channel until it has an endpoint. If you're using input redundancy, each endpoint receives content from one input URL at a time. If MediaPackage performs a failover on the inputs for one input URL, the endpoints automatically start receiving content from the other input URL. For more information about input redundancy and failover, see Live input redundancy AWS Elemental MediaPackage processing flow.

When you create an endpoint, MediaPackage assigns it a public URL that's fixed for the lifetime of the endpoint, regardless of any failures or upgrades that might happen over time. This URL is how the player or CDN accesses the stream from the endpoint.

You can use the MediaPackage console, the AWS CLI, or the MediaPackage API to create an endpoint. For information about creating an endpoint through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

MediaPackage doesn't require customer data from you, so endpoints don't include those fields.

Topics

- Creating an HLS endpoint
- Creating a DASH endpoint
- Creating a Microsoft Smooth Streaming endpoint
- Creating a CMAF endpoint

Creating an HLS endpoint

Create an endpoint that formats content for devices that support Apple HLS.

To create an Apple HLS endpoint (console)

1. Access the channel that the endpoint will be associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, choose **Manage endpoints**.
3. Complete the fields as described in the following topics:
   - New endpoint fields
   - Packager settings fields
   - Package encryption fields
   - Access control settings fields
   - Stream selection fields
4. Choose **Save**.

   If you enabled Amazon CloudFront distribution creation from the AWS Elemental MediaPackage console and this is your first endpoint on the channel, MediaPackage adds an origin to the distribution. You can view the CloudFront CDN URL and endpoint information in the endpoints section of the channel's details page.

   The endpoint is active and can deliver content as soon as requests are sent to its URL endpoints. MediaPackage scales resources up and down to allow the right amount of capacity for your traffic.

   When you're creating an endpoint, you will receive an error if you exceed the quotas on the account. An error similar to Too many requests, please try again. Resource limit exceeded means that either you've exceeded the API request quotas, or you've already reached the maximum number of endpoints allowed on this channel. If you think you received this error

wrongfully, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

New endpoint fields

When you're creating an endpoint, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with AWS Elemental MediaPackage using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

1. For **ID**, enter a name that describes the endpoint. The ID is the primary identifier for the endpoint and must be unique for your account in the AWS Region.
2. (Optional) For **Description**, enter any descriptive text that helps you to identify the endpoint.
3. For **Manifest name**, enter a short string that will be appended to the end of the endpoint URL. The manifest name helps to create a unique path to this endpoint.
4. (Optional) For **Startover window**, enter the size of the window (in seconds) to create a window of the live stream that's available for on-demand viewing. Viewers can start-over or catch-up on content that falls within the window. For more information about implementing start-over and catch-up TV, see Time-shifted viewing reference in AWS Elemental MediaPackage.
5. (Optional) For **Time delay**, enter the duration (in seconds) to delay when content is available to players. The minimum time is 5 seconds. The maximum time is 86,400 seconds (24 hours).

   Use time delay to redefine the live point and make content available at a time that equals "now" minus the delay specified. With a 60-second time delay, content that MediaPackage receives at 12:20 isn't available until 12:21. Requests for playback at 12:20 will be served with content from 12:19. Likewise, if you're serving content across time zones, you can set a time delay equal to the time zone difference to make content available at, for example, 8:00 local time.

   When you use time delay in conjunction with a startover window, the time delay duration must be less than the startover window duration.

> **Tip**
>
> Use a time delay to help reduce buffering during input switching when you're using input redundancy with short output segments. Note that the delay can increase latency in content playback.

Packager settings fields

The Packager settings fields hold general information about the endpoint.

1. For **Packaging type**, choose **Apple HLS**.
2. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.
3. (Optional) For **Live playlist window duration**, enter the total duration (in seconds) of the parent manifest.
4. (Optional) Select **Use audio rendition group** to group all audio tracks into a single HLS rendition group. For more information about rendition groups, see Rendition groups reference in AWS Elemental MediaPackage.
5. (Optional) Select **Include DVB subtitles** to passthrough DVB subtitles into the output.
6. (Optional) Select **Include IFrame only stream** to include an additional I-frame only stream along with the other tracks in the manifest. MediaPackage generates an I-frame only stream from the first rendition in the manifest. The service inserts `EXT-I-FRAMES-ONLY` tags in the output manifest, and then generates and includes an I-frames only playlist in the stream. This playlist enables player functionality like fast forward and rewind.
7. (Optional) For **Program date/time interval**, enter the interval (in seconds) for MediaPackage to insert the `EXT-X-PROGRAM-DATE-TIME` tags in the manifest. Default is 0 (`EXT-X-PROGRAM-DATE-TIME` tags aren't inserted).

   The `EXT-X-PROGRAM-DATE-TIME` tag holds the time of the segment. When program date time (PDT) information is available in the source content, MediaPackage uses this same information on the output content. Otherwise, MediaPackage uses Coordinated Universal Time (UTC) for the PDT.

The PDT information helps downstream players to synchronize the stream to the wall clock, enabling functionality like viewer seek in the playback timeline and time display on the player.

> **Tip**
>
> `EXT-X-PROGRAM-DATE-TIME` tags help to mitigate synchronization issues that could break time-shifted viewing or live-to-VOD workflows.

8. (Optional) For **Playlist type**, choose **Event** or **VOD**. When specified as either event or VOD, a corresponding `EXT-X-PLAYLIST-TYPE` entry is included in the media playlist. Indicates if the playlist is live to VOD content.

SCTE-35 Options

The following fields dictate how MediaPackage processes SCTE-35 messages from the input stream. For more information, see SCTE-35 message options in AWS Elemental MediaPackage.

1. (Optional) For **Ad markers**, choose how ad markers are included in the packaged content.

   Choose from the following:

   - **None** – Omit all SCTE-35 ad markers from the output.
   - **Passthrough** – Copy the SCTE-35 ad markers directly from the input HLS input stream to the output.
   - **SCTE-35 enhanced** – Generate ad markers and blackout tags in the output based on the SCTE-35 input messages from the input stream.
   - **Daterange** – Emit `EXT-X-DATERANGE` tags in HLS and CMAF manifests to signal ads and program transitions.

2. (Optional) For **Ad triggers**, choose the SCTE-35 message types that you want to be treated as ad markers in the output. If you don't make a selection here, MediaPackage inserts ad markers in the output manifest based on these message types:

   - Splice insert
   - Provider advertisement
   - Distributor advertisement
   - Provider placement opportunity
   - Distributor placement opportunity

3. (Optional) For **Ads on delivery restrictions**, choose what ad insertion action MediaPackage takes based on delivery restriction flags in the segmentation descriptors of SCTE-35 messages.
   - **None** – MediaPackage doesn't insert any ad markers in the output manifest.
   - **Restricted** – MediaPackage inserts ad markers when there *are* delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.
   - **Unrestricted** – MediaPackage inserts ad markers when there *aren't* delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.
   - **Both** – MediaPackage inserts ad markers whether or not there are delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.

Package encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM provider, and be set up to use encryption. For information, see the section called “Content encryption and DRM”.

Define the encryption values.

1. To serve content without copyright protection, keep **No encryption** selected.
2. To serve content with copyright protection, choose **Encrypt content** and complete the additional fields as follows:
   a. For **Resource ID**, enter an identifier for the content. The service sends this to the key server to identify the current endpoint. How unique you make this depends on how fine-grained you want access controls to be. The service does not allow you to use the same ID for two simultaneous encryption processes. The resource ID is also known as the content ID.

      The following example shows a resource ID.

```
MovieNight20171126093045
```

b. For **System ID**, enter unique identifiers for your streaming protocol and DRM system. Provide one system ID. If you do not know your ID, ask your DRM provider.

c. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

   The following example shows a URL.

```
https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
```

d. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

   The following example shows a role ARN.

```
arn:aws:iam::444455556666:role/SpekeAccess
```

e. **Certificate ARN** – (Optional) Enter a 2048 RSA certificate ARN to use for content key encryption. Use this option only if your DRM key provider supports content key encryption. If you use this and your key provider doesn't support it, the event fails.

   To enter a certificate ARN here, you must have already imported the corresponding certificate into AWS Certificate Manager. Then enter the certificate ARN from ACM here.

   For information about content key encryption, see Preparing and managing certificates for use with content keys.

f. For **Encryption method**, choose **Sample-AES** for Apple HLS FairPlay or choose **AES-128** for Apple HLS AES-128.

g. (Optional) For **Constant initialization vector** enter a 128-bit, 16-byte hex value represented by a 32-character string, to be used with the key for encrypting content.

h. (Optional) For **Key rotation interval**, enter the frequency, in seconds, of key changes for live workflows, in which content is streamed real time. The service retrieves content keys before the live content begins streaming, and then retrieves them as needed over the lifetime of the workflow. By default, key rotation is set to 60 seconds, which is equivalent to setting it to `60`. To disable key rotation, set this interval to `0` (zero).

   The following example setting causes the service to rotate keys every thirty minutes.

```
1800
```

For information about key rotation, see Understanding key rotation behavior.

i. (Optional) Select **Repeat EXT-X-KEY** if you want the service to repeat the key before every segment of the manifest. By default, the key is written just once, after the header and before the segments. If you select **Repeat EXT-X-KEY**, the manifest is written as header, key, segment, key, segment, key, and so on, with every segment preceded by the key. Set this according to the needs of the player. Selecting this option might result in an increase in client requests to the DRM server.

Access control settings fields

Define the access control values.

1. Select **Allow origination** to enable this endpoint to serve content to requesting devices. It is uncommon to disallow origination on an endpoint.

   Typically, the only reason that you won't allow an endpoint to serve content is if it's only being used to harvest VOD content from the live stream. For more information, see Creating live-to-VOD assets with AWS Elemental MediaPackage.
2. Choose **Allow all incoming clients** to serve content to all requesting IP addresses and ranges or choose **Restrict by IP address** to limit the IP addresses that this endpoint serves. If you restrict by IP address, for **IP allowlist**, enter the IP addresses and ranges that this endpoint serves content to. One CIDR block per line.

   > **Note**
   >
   > Only IPv4 addresses are allowed.

3. Select **Use CDN authorization** to require that content requests to this endpoint include a valid authorization code. Complete the remaining fields:

   a. For **Secrets role ARN**, enter the ARN for the IAM role that grants MediaPackage access to AWS Secrets Manager. The Secrets role ARN must be in this format: `arn:aws:iam::accountID:role/name`

   b. For **CDN identifier secret ARN**, enter the ARN for the authorization code secret in Secrets Manager that your CDN uses for authorization to access

your endpoint. The CDN identifier secret ARN must be in this format: `arn:aws:secretsmanager:region:accountID:secretguid`.

For information about how this authorization works, see CDN authorization in AWS Elemental MediaPackage.

Stream selection fields

Define the streams to include.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

1. (Optional) For **Stream order**, choose from the following:
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Video bitrate ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Video bitrate descending** to sort the output streams starting with the highest bitrate and ending with the lowest.
2. (Optional) For **Min video bitrate**, enter the minimum bitrate (in bits per second) that video tracks must be at or above to be available for playback from this endpoint.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate (in bits per second) that video tracks must be at or below to be available for playback from this endpoint.

Creating a DASH endpoint

Create an endpoint that formats content for devices that support MPEG-DASH.

To create an MPEG-DASH endpoint (console)

1. Access the channel that the endpoint will be associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, choose **Manage endpoints**.
3. Choose **Manage endpoints**.
4. Complete the fields as described in the following topics:

- New endpoint fields
- Packager settings fields
- Package encryption fields
- Access control settings fields
- Stream selection fields

5. Choose **Save**.

   If you enabled Amazon CloudFront distribution creation from the AWS Elemental MediaPackage console and this is your first endpoint on the channel, MediaPackage adds an origin to the distribution. You can view the CloudFront CDN URL and endpoint information in the endpoints section of the channel's details page.

   The endpoint is active and can deliver content as soon as requests are sent to its URL endpoints. MediaPackage scales resources up and down to allow the right amount of capacity for your traffic.

   When you're creating an endpoint, you will receive an error if you exceed the quotas on the account. An error similar to Too many requests, please try again. Resource limit exceeded means that either you've exceeded the API request quotas, or you've already reached the maximum number of endpoints allowed on this channel. If you think you received this error wrongfully, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

New endpoint fields

When you're creating an endpoint, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with AWS Elemental MediaPackage using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

1. For **ID**, enter a name that describes the endpoint. The ID is the primary identifier for the endpoint and must be unique for your account in the AWS Region.
2. (Optional) For **Description**, enter any descriptive text that helps you to identify the endpoint.
3. For **Manifest name**, enter a short string that will be appended to the end of the endpoint URL. The manifest name helps to create a unique path to this endpoint.

4. (Optional) For **Startover window**, enter the size of the window (in seconds) to create a window of the live stream that's available for on-demand viewing. Viewers can start-over or catch-up on content that falls within the window. For more information about implementing start-over and catch-up TV, see Time-shifted viewing reference in AWS Elemental MediaPackage.
5. (Optional) For **Time delay**, enter the duration (in seconds) to delay when content is available to players. The minimum time is 5 seconds. The maximum time is 86,400 seconds (24 hours).

   Use time delay to redefine the live point and make content available at a time that equals "now" minus the delay specified. With a 60-second time delay, content that MediaPackage receives at 12:20 isn't available until 12:21. Requests for playback at 12:20 will be served with content from 12:19. Likewise, if you're serving content across time zones, you can set a time delay equal to the time zone difference to make content available at, for example, 8:00 local time.

   When you use time delay in conjunction with a startover window, the time delay duration must be less than the startover window duration.

   > **Tip**
   >
   > Use a time delay to help reduce buffering during input switching when you're using input redundancy with short output segments. Note that the delay can increase latency in content playback.

Packager settings fields

1. For **Packaging type**, choose **DASH-ISO**.
2. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.

   > **Important**
   >
   > If you enable **Number with duration** in **Segment template format**, you can't change the segment duration after you've created the endpoint.

3. (Optional) For **Manifest window duration**, enter the total duration (in seconds) of the manifest.

4. (Optional) For **Profile**, choose a DASH profile that determines the segment and manifest formats of the output.
   - **None** – the output doesn't use a DASH profile.
   - **Hbbtv 1.5** – the output is compliant with HbbTV v1.5. For information about HbbTV v1.5, see the HbbTV specification website.
   - **Hybridcast** – the output is compliant with Hybridcast. For more information about Hybridcast, see the IPTV Forum Japan Hybridcast specification. If you enable the Hybridcast profile on your packaging configuration, you can't use DASH **Period triggers**.
   - **DVB-DASH 2014** – the output is compliant with DVB-DASH 2014. For more information about DVB-DASH 2014, see the DVB-DASH specification.
5. (Optional) For **Manifest layout**, choose if you want MediaPackage to serve a full or compact manifest in response to playback requests.
   - If you choose **Full**, MediaPackage presents the `SegmentTemplate` and `SegmentTimeline` tags for every `Representation` in the manifest.
   - If you choose **Compact**, MediaPackage combines duplicate `SegmentTemplate` tags and presents them at the start of the manifest. This shortens the manifest and makes it easier for some devices to process it.

   For more information about the manifest layout options, see Compacted DASH manifests.
6. (Optional) For **Min update period**, enter the minimum amount of time (in seconds) that the player should wait before requesting manifest updates. A lower value means that manifests are updated more frequently, but a lower value also contributes to request and response network traffic.
7. (Optional) For **Min buffer time**, enter the minimum amount of time (in seconds) that a player must keep in the buffer. If network conditions interrupt playback, the player will have additional buffered content before playback fails, allowing for recovery time before the viewer's experience is affected.
8. (Optional) For **Suggested presentation delay**, enter the amount of time (in seconds) that the player should be from the end of the manifest. This sets the content start point back x seconds from the end of the manifest (the point where content is live). For example, with a 35-second presentation delay, requests at 5:30 receive content from 5:29:25. When used with time delay, MediaPackage adds the suggested presentation delay to the time delay duration.
9. (Optional) For **Segment template format**, choose how MediaPackage and playback requests refer to each segment.

- If you choose **Number with timeline**, MediaPackage uses the `$Number$` variable to refer to the segment in the `media` attribute of the `SegmentTemplate` tag. The value of the variable is the sequential number of the segment. `SegmentTimeline` is included in each segment template.
- If you choose **Number with duration**, MediaPackage uses the `$Number$` variable and replaces the `SegmentTimeline` objects with a `duration` attribute in the segment template.

  > **Note**
  >
  > This option isn't supported in combination with multi-period DASH.

- If you choose **Time with timeline**, MediaPackage uses the `$Time$` variable to refer to the segment. The value of the variable is the timestamp of when on the manifest timeline the segment starts. `SegmentTimeline` is included in each segment template.

For more information about the formatting options of the `SegmentTemplate` tag, see DASH manifest segment template format.

10. (Optional) For **UTC timing**, select the method that the player uses to synchronize to coordinated universal time (UTC) wall clock time. This enables the player and MediaPackage to run on the same UTC wall clock time. This is a requirement, otherwise playback timing or synchronization issues can occur.

    The options are `HTTP-HEAD`, `HTTP-ISO`, `HTTP-XSDATE`, and `NONE`. This value will be set as the `@schemeIdURI` attribute for the `UTCTiming` element in the outbound Media Presentation Description. For information about `UTCTiming`, see DASH clock synchronization.

11. (Optional) For **UTC timing URI**, specify a URI to use for UTC synchronization. This is the URI used to fetch the timing data according to the scheme defined by **UTC timing**. This value is only valid if **UTC timing** is not `NONE`. This value will be set as the `@value` attribute for the `UTCTiming` element.

12. (Optional) Select **Include IFrame only stream** to include an additional I-frame only stream along with the other tracks in the manifest. MediaPackage generates an I-frame only stream from the first rendition in the manifest. The service inserts an `<EssentialProperty schemeIdUri="http://dashif.org/guidelines/trickmode" value="`*X*`"/>` descriptor, where *X* is the ID of the original Adaptation set, and then generates and includes an I-frame only rendition in the stream. If you use encryption, MediaPackage encrypts the I-frame only

rendition with the same content key as the original video rendition. This rendition enables player functionality like fast forward and rewind.

13For **Period triggers**, choose how MediaPackage creates media presentation description (MPD) periods in the DASH output manifest. Choose from the following:

- **None** – MediaPackage doesn't create additional periods. It formats the manifest as a single period and doesn't include SCTE-35 markers in the segments.
- **Trigger new periods on ads** – MediaPackage creates and inserts in the manifest multiple periods based on SCTE-35 ad markers from the input content. These periods separate portions of the content, such as setting boundaries between the main content and ad content. For more information about how MediaPackage configures periods in the manifest, see DASH manifest options in AWS Elemental MediaPackage.

> **Important**
>
> Multiple periods are required if you use AWS Elemental MediaTailor for personalized ad insertion in DASH content. For more information about this service, see the AWS Elemental MediaTailor User Guide.

SCTE-35 Options

The following fields dictate how MediaPackage processes SCTE-35 messages from the input stream. For more information, see SCTE-35 message options in AWS Elemental MediaPackage.

1. (Optional) For **Ad triggers**, choose the SCTE-35 message types that you want to be treated as ad markers in the output. If you don't make a selection here, MediaPackage inserts ad markers in the output manifest based on these message types:
   - Splice insert
   - Provider advertisement
   - Distributor advertisement
   - Provider placement opportunity
   - Distributor placement opportunity
2. (Optional) For **Ads on delivery restrictions**, choose what ad insertion action MediaPackage takes based on delivery restriction flags in the segmentation descriptors of SCTE-35 messages.
   - **None** – MediaPackage doesn't insert any ad markers in the output manifest.

- **Restricted** – MediaPackage inserts ad markers when there *are* delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.
- **Unrestricted** – MediaPackage inserts ad markers when there *aren't* delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.
- **Both** – MediaPackage inserts ad markers whether or not there are delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.

If you choose not to insert ad markers, MediaPackage also won't create periods. The output manifest is contained in a single period.

Package encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM solution provider and be set up to use encryption. For information, see the section called “Content encryption and DRM”.

Define the encryption values.

1. To serve content without copyright protection, keep **No encryption** selected.
2. To serve content with copyright protection, choose **Encrypt content** and complete the additional fields as follows:
   a. For **Resource ID**, enter an identifier for the content. The service sends this to the key server to identify the current endpoint. How unique you make this depends on how fine-grained you want access controls to be. The service does not allow you to use the same ID for two simultaneous encryption processes. The resource ID is also known as the content ID.

      The following example shows a resource ID.

```
MovieNight20171126093045
```

b. For **System IDs**, enter unique identifiers for your streaming protocol and DRM system. Provide up to two IDs. If you provide more than one system ID, enter one per line and choose **Add**. For a list of common system IDs, see DASH-IF System IDs. If you do not know your IDs, ask your DRM solution provider.

c. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

   The following example shows a URL.

```
https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
```

d. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

   The following example shows a role ARN.

```
arn:aws:iam::444455556666:role/SpekeAccess
```

e. (Optional) For **SPEKE version**, select the SPEKE version that you'd like to use for encryption. SPEKE Version 1.0 is the legacy version that uses CPIX Version 2.0, and supports single key encryption. SPEKE Version 2.0 uses CPIX Version 2.3, and supports multiple key encryption. For more information about using SPEKE with MediaPackage, see Content encryption and DRM in AWS Elemental MediaPackage.

   If you select **SPEKE Version 2.0**, then also choose a **Video encryption preset** and an **Audio encryption preset**. The video and audio presets determine which content keys MediaPackage uses to encrypt the audio and video tracks in your stream. For more information about these presets, see SPEKE Version 2.0 presets.

   When using SPEKE Version 2.0, MediaPackage disables key rotation.

f. **Certificate ARN** – (Optional) Enter a 2048 RSA certificate ARN to use for content key encryption. Use this option only if your DRM key provider supports content key encryption. If you use this and your key provider doesn't support it, the event fails.

   To enter a certificate ARN here, you must have already imported the corresponding certificate into AWS Certificate Manager. Then enter the certificate ARN from ACM here.

For information about content key encryption, see Preparing and managing certificates for use with content keys.

g. (Optional) For **Key rotation interval**, enter the frequency, in seconds, of key changes for live workflows, in which content is streamed real time. The service retrieves content keys before the live content begins streaming, and then retrieves them as needed over the lifetime of the workflow. By default, key rotation is set to 60 seconds, which is equivalent to setting it to `60`. To disable key rotation, set this interval to `0` (zero).

The following example setting causes the service to rotate keys every thirty minutes.

```
1800
```

For information about key rotation, see Understanding key rotation behavior.

Access control settings fields

Define the access control values.

1. Select **Allow origination** to enable this endpoint to serve content to requesting devices. It is uncommon to disallow origination on an endpoint.

   Typically, the only reason that you won't allow an endpoint to serve content is if it's only being used to harvest VOD content from the live stream. For more information, see Creating live-to-VOD assets with AWS Elemental MediaPackage.
2. Choose **Allow all incoming clients** to serve content to all requesting IP addresses and ranges or choose **Restrict by IP address** to limit the IP addresses that this endpoint serves. If you restrict by IP address, for **IP allowlist**, enter the IP addresses and ranges that this endpoint serves content to. One CIDR block per line.
3. Select **Use CDN authorization** to require that content requests to this endpoint include a valid authorization code.
4. (Optional) For **Secrets role ARN**, enter the ARN for the IAM role that grants MediaPackage access to AWS Secrets Manager. The Secrets role ARN must be in this format: `arn:aws:iam::`*`accountID`*`:role/`*`name`*.
5. (Optional) For **CDN identifier secret ARN**, enter the ARN for the authorization code secret in Secrets Manager that your CDN uses for authorization

to access your endpoint. The CDN identifier must be in this format: `arn:aws:secretsmanager:region:accountID:secret:guid`.

For information about how this authorization works, see CDN authorization in AWS Elemental MediaPackage.

Stream selection fields

Define the streams to include.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

1. (Optional) For **Stream order**, choose the order that video bitrates are presented to the player:
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Video bitrate ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Video bitrate descending** to sort the output streams starting with the highest bitrate and ending with the lowest.
2. (Optional) For **Min video bitrate**, enter the minimum bitrate (in bits per second) that video tracks must be at or above to be available for playback from this endpoint.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate (in bits per second) that video tracks must be at or below to be available for playback from this endpoint.

Creating a Microsoft Smooth Streaming endpoint

Create an endpoint that formats content for devices that support Microsoft Smooth Streaming.

To create a Microsoft Smooth Streaming endpoint (console)

1. Access the channel that the endpoint will be associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, choose **Manage endpoints**.
3. Complete the fields as described in the following topics:

- New endpoint fields
- Packager settings fields
- Package encryption fields
- Access control settings fields
- Stream selection fields

4. Choose **Save**.

   If you enabled Amazon CloudFront distribution creation from the AWS Elemental MediaPackage console and this is your first endpoint on the channel, MediaPackage adds an origin to the distribution. You can view the CloudFront CDN URL and endpoint information in the endpoints section of the channel's details page.

   The endpoint is active and can deliver content as soon as requests are sent to its URL endpoints. MediaPackage scales resources up and down to allow the right amount of capacity for your traffic.

   When you're creating an endpoint, you will receive an error if you exceed the quotas on the account. An error similar to Too many requests, please try again. Resource limit exceeded means that either you've exceeded the API request quotas, or you've already reached the maximum number of endpoints allowed on this channel. If you think you received this error wrongfully, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

New endpoint fields

When you're creating an endpoint, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with AWS Elemental MediaPackage using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

1. For **ID**, enter a name that describes the endpoint. The ID is the primary identifier for the endpoint and must be unique for your account in the AWS Region.
2. (Optional) For **Description**, enter any descriptive text that helps you to identify the endpoint.
3. For **Manifest name**, enter a short string that will be appended to the end of the endpoint URL. The manifest name helps to create a unique path to this endpoint.

4. (Optional) For **Startover window**, enter the size of the window (in seconds) to create a window of the live stream that's available for on-demand viewing. Viewers can start-over or catch-up on content that falls within the window. For more information about implementing start-over and catch-up TV, see Time-shifted viewing reference in AWS Elemental MediaPackage.
5. (Optional) For **Time delay**, enter the duration (in seconds) to delay when content is available to players. The minimum time is 5 seconds. The maximum time is 86,400 seconds (24 hours).

   Use time delay to redefine the live point and make content available at a time that equals "now" minus the delay specified. With a 60-second time delay, content that MediaPackage receives at 12:20 isn't available until 12:21. Requests for playback at 12:20 will be served with content from 12:19. Likewise, if you're serving content across time zones, you can set a time delay equal to the time zone difference to make content available at, for example, 8:00 local time.

   When you use time delay in conjunction with a startover window, the time delay duration must be less than the startover window duration.

   > **Tip**
   >
   > Use a time delay to help reduce buffering during input switching when you're using input redundancy with short output segments. Note that the delay can increase latency in content playback.

Packager settings fields

The Packager settings fields hold general information about the endpoint.

1. For **Packaging type**, choose **Microsoft Smooth**.
2. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.
3. (Optional) For **Manifest window duration**, enter the total duration (in seconds) of the manifest.

Package encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder

Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM solution provider and be set up to use encryption. For information, see the section called "Content encryption and DRM".

Define the encryption values.

1. To serve content without copyright protection, keep **No encryption** selected.
2. To serve content with copyright protection, choose **Encrypt content** and complete the additional fields as follows:
   a. For **Resource ID**, enter an identifier for the content. The service sends this to the key server to identify the current endpoint. How unique you make this depends on how fine-grained you want access controls to be. The service does not allow you to use the same ID for two simultaneous encryption processes. The resource ID is also known as the content ID.

      The following example shows a resource ID.

      ```
      MovieNight20171126093045
      ```

   b. For **System ID**, enter unique identifiers for your streaming protocol and DRM system. Provide up to one system ID. If you do not know your ID, ask your DRM solution provider.
   c. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

      The following example shows a URL.

      ```
      https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
      ```

   d. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

      The following example shows a role ARN.

```
arn:aws:iam::444455556666:role/SpekeAccess
```

e. **Certificate ARN** – (Optional) Enter a 2048 RSA certificate ARN to use for content key encryption. Use this option only if your DRM key provider supports content key encryption. If you use this and your key provider doesn't support it, the event fails.

To enter a certificate ARN here, you must have already imported the corresponding certificate into AWS Certificate Manager. Then enter the certificate ARN from ACM here.

For information about key encryption, see Preparing and managing certificates for use with content keys.

Access control settings fields

Define the access control values.

1. Select **Allow origination** to enable this endpoint to serve content to requesting devices. It is uncommon to disallow origination on an endpoint.

   Typically, the only reason that you won't allow an endpoint to serve content is if it's only being used to harvest VOD content from the live stream. For more information, see Creating live-to-VOD assets with AWS Elemental MediaPackage.
2. Choose **Allow all incoming clients** to serve content to all requesting IP addresses and ranges or choose **Restrict by IP address** to limit the IP addresses that this endpoint serves. If you restrict by IP address, for **IP allowlist**, enter the IP addresses and ranges that this endpoint serves content to. One CIDR block per line.
3. Select **Use CDN authorization** to require that content requests to this endpoint include a valid authorization code.
4. (Optional) For **Secrets role ARN**, enter the ARN for the IAM role that grants MediaPackage access to AWS Secrets Manager. The secrets role ARN must be in this format: `arn:aws:iam::accountID:role/name`.
5. (Optional) For **CDN identifier secret ARN**, enter the ARN for the authorization code secret in Secrets Manager that your CDN uses for authorization to access your endpoint. The CDN identifier must be in this format: `arn:aws:secretsmanager:region:accountID:secretguid`.

For information about how this authorization works, see CDN authorization in AWS Elemental MediaPackage.

Stream selection fields

Define the streams to include.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

1. (Optional) For **Stream order**, choose the order that video bitrates are presented to the player.
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Video bitrate ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Video bitrate descending** to sort the output streams starting with the highest bitrate and ending with the lowest.
2. (Optional) For **Min video bitrate**, enter the minimum bitrate (in bits per second) that video tracks must be at or above to be available for playback from this endpoint.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate (in bits per second) that video tracks must be at or below to be available for playback from this endpoint.

Creating a CMAF endpoint

Create an endpoint that formats content for devices that support Apple HLS fragmented MP4 (fMP4).

To create a CMAF endpoint (console)

1. Access the channel that the endpoint will be associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, choose **Manage endpoints**.
3. Complete the fields as described in the following topics:
   - New endpoint fields
   - Packager settings fields

- Package encryption fields
- Access control settings fields
- Stream selection fields

4. Choose **Save**.

   If you enabled Amazon CloudFront distribution creation from the AWS Elemental MediaPackage console and this is your first endpoint on the channel, MediaPackage adds an origin to the distribution. You can view the CloudFront CDN URL and endpoint information in the endpoints section of the channel's details page.

   The endpoint is active and can deliver content as soon as requests are sent to its URL endpoints. MediaPackage scales resources up and down to allow the right amount of capacity for your traffic.

   When you're creating an endpoint, you will receive an error if you exceed the quotas on the account. An error similar to Too many requests, please try again. Resource limit exceeded means that either you've exceeded the API request quotas, or you've already reached the maximum number of endpoints allowed on this channel. If you think you received this error wrongfully, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

New endpoint fields

When you're creating an endpoint, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with AWS Elemental MediaPackage using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

1. For **ID**, enter a name that describes the endpoint. The ID is the primary identifier for the endpoint and must be unique for your account in the AWS Region.
2. (Optional) For **Description**, enter any descriptive text that helps you to identify the endpoint.
3. For **Manifest name**, enter a short string that will be appended to the end of the endpoint URL. The manifest name helps to create a unique path to this endpoint.
4. (Optional) For **Startover window**, enter the size of the window (in seconds) to create a window of the live stream that's available for on-demand viewing. Viewers can start-over or catch-up on

content that falls within the window. For more information about implementing start-over and catch-up TV, see Time-shifted viewing reference in AWS Elemental MediaPackage.

5. (Optional) For **Time delay**, enter the duration (in seconds) to delay when content is available to players. The minimum time is 5 seconds. The maximum time is 86,400 seconds (24 hours).

   Use time delay to redefine the live point and make content available at a time that equals "now" minus the delay specified. With a 60-second time delay, content that MediaPackage receives at 12:20 isn't available until 12:21. Requests for playback at 12:20 will be served with content from 12:19. Likewise, if you're serving content across time zones, you can set a time delay equal to the time zone difference to make content available at, for example, 8:00 local time.

   When you use time delay in conjunction with a startover window, the time delay duration must be less than the startover window duration.

   > **Tip**
   >
   > Use a time delay to help reduce buffering during input switching when you're using input redundancy with short output segments. Note that the delay can increase latency in content playback.

Packager settings fields

The Packager settings fields hold general information about the endpoint.

1. For **Packaging type**, choose **Common Media Application Format (CMAF)**.
2. For **HLS Manifest ID**, enter an ID that will be the primary identifier for the manifest. The ID must be unique for this endpoint. You cannot change this ID after it's created.
3. (Optional) For **Segment prefix**, enter a custom name for the segments in the HLS child manifest. The segment prefix is prepended to the segment name to create a unique identifier for each segment.

   Example

   If the segment prefix is `movie`, a segment from the child manifest is `movie_1_2.ts`.
4. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different

from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.

5. (Optional) For **Live playlist window duration**, enter the total duration (in seconds) of the parent manifest.
6. For **Manifest name**, enter a string that will be appended to the end of the endpoint URL. The manifest name helps to create a unique path to this manifest on this endpoint. The HLS manifest name overrides the manifest name that you provided in the New Endpoint **Manifest name** field (if applicable).
7. (Optional) Select **Include IFrame only stream** to include an additional I-frame only stream along with the other tracks in the manifest. MediaPackage generates an I-frame only stream from the first rendition in the manifest. The service inserts `EXT-I-FRAMES-ONLY` tags in the output manifest, and then compiles and includes an I-frames only playlist in the stream. This playlist enables player functionality like fast forward and rewind.
8. (Optional) For **Program date/time interval**, enter the interval (in seconds) at which MediaPackage should insert the `EXT-X-PROGRAM-DATE-TIME` tags in the manifest.

   The `EXT-X-PROGRAM-DATE-TIME` tag holds the time of the segment. When program date time (PDT) information is available in the source content, MediaPackage uses this same information on the output content. Otherwise, MediaPackage uses Coordinated Universal Time (UTC) for the PDT.

   The PDT information helps downstream players to synchronize the stream to the wall clock, enabling functionality like viewer seek in the playback timeline and time display on the player.
9. (Optional) For **Playlist type**, choose **None**, **Event**, or **VOD**. When specified as either event or VOD, a corresponding `EXT-X-PLAYLIST-TYPE` entry is included in the media playlist. Indicates if the playlist is live to VOD content.
10. (Optional) Use the following fields to dictate how MediaPackage processes SCTE-35 messages from the input stream. For more information, see SCTE-35 message options in AWS Elemental MediaPackage.

    a. (Optional) For **Ad markers**, choose how ad markers are included in the packaged content.

       Choose from the following:

       - **None** – Omit all SCTE-35 ad markers from the output.
       - **Passthrough** – Copy the SCTE-35 ad markers directly from the input HLS input stream to the output.

- **SCTE-35 enhanced** – Generate ad markers and blackout tags in the output based on the SCTE-35 input messages from the input stream.
- **Daterange** – Emit `EXT-X-DATERANGE` tags in HLS and CMAF manifests to signal ads and program transitions.

b. (Optional) For **Ad triggers**, choose the SCTE-35 message types that you want to be treated as ad markers in the output. If you don't make a selection here, MediaPackage inserts ad markers in the output manifest based on these message types:

- Splice insert
- Provider advertisement
- Distributor advertisement
- Provider placement opportunity
- Distributor placement opportunity

c. (Optional) For **Ads on delivery restrictions**, choose what ad insertion action MediaPackage takes based on delivery restriction flags in the segmentation descriptors of SCTE-35 messages.

- **None** – MediaPackage doesn't insert any ad markers in the output manifest.
- **Restricted** – MediaPackage inserts ad markers when there *are* delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.
- **Unrestricted** – MediaPackage inserts ad markers when there *aren't* delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.
- **Both** – MediaPackage inserts ad markers whether or not there are delivery restrictions in the SCTE-35 message types that you indicated in **Customize ad triggers**.

Package encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

Important

To encrypt content, you must have a DRM provider and use a version of AWS SPEKE. For more information about how to use encryption for MediaPackage, see Content encryption and DRM in AWS Elemental MediaPackage.

Define the encryption values.

1. To serve content without copyright protection, keep **No encryption** selected.
2. To serve content with copyright protection, choose **Encrypt content** and complete the additional fields as follows:
   a. For **Resource ID**, enter an identifier for the content. The service sends this to the key server to identify the current endpoint. How unique you make this depends on how fine-grained you want access controls to be. The service does not allow you to use the same ID for two simultaneous encryption processes. The resource ID is also known as the content ID.

      The following example shows a resource ID.

      ```
      MovieNight20171126093045
      ```

   b. For **System IDs**, enter a unique identifier for your streaming protocol and DRM system. Provide up to three IDs. If you provide more than one system ID, enter one per line and choose **Add**. If you do not know your IDs, ask your system provider.
   c. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

      The following example shows a URL.

      ```
      https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
      ```

   d. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

      The following example shows a role ARN.

      ```
      arn:aws:iam::444455556666:role/SpekeAccess
      ```

e. (Optional) For **SPEKE version**, select the SPEKE version that you'd like to use for encryption. SPEKE Version 1.0 is the legacy version that uses CPIX Version 2.0, and supports single key encryption. SPEKE Version 2.0 uses CPIX Version 2.3, and supports multiple key encryption. For more information about using SPEKE with MediaPackage, see Content encryption and DRM in MediaPackage.

   If you select **SPEKE Version 2.0**, then also choose a **Video encryption preset** and an **Audio encryption preset**. The video and audio presets determine which content keys MediaPackage uses to encrypt the audio and video tracks in your stream. For more information about these presets, see SPEKE Version 2.0 presets.

   When using SPEKE Version 2.0, MediaPackage disables key rotation.

f. **Certificate ARN** – (Optional) Enter a 2048 RSA certificate ARN to use for content key encryption. Use this option only if your DRM key provider supports content key encryption. If you use this and your key provider doesn't support it, the event fails.

   To enter a certificate ARN here, you must have already imported the corresponding certificate into AWS Certificate Manager. Then enter the certificate ARN from ACM here.

   For information about content key encryption, see Preparing and managing certificates for use with content keys.

g. For **Encryption Method**, choose **Sample-AES** for CMAF Apple HLS FairPlay or choose **AES-CTR** for Microsoft PlayReady and Google Widevine.

h. (Optional) For **Constant initialization vector** enter a 128-bit, 16-byte hex value represented by a 32-character string, to be used with the key for encrypting content.

i. (Optional) For **Key rotation interval**, enter the frequency, in seconds, of key changes for live workflows, in which content is streamed real time. The service retrieves content keys before the live content begins streaming, and then retrieves them as needed over the lifetime of the workflow. By default, key rotation is set to 60 seconds, which is equivalent to setting it to `60`. To disable key rotation, set this interval to `0` (zero).

   The following example setting causes the service to rotate keys every thirty minutes.

   ```
   1800
   ```

   For information about key rotation, see Understanding key rotation behavior.

Access control settings fields

Define the access control values.

1. Select **Allow origination** to enable this endpoint to serve content to requesting devices. It's uncommon to disallow origination on an endpoint.

   Typically, the only reason that you won't allow an endpoint to serve content is if it's only being used to harvest VOD content from the live stream. For more information, see Creating live-to-VOD assets with AWS Elemental MediaPackage.
2. Choose **Allow all incoming clients** to serve content to all requesting IP addresses and ranges or choose **Restrict by IP address** to limit the IP addresses that this endpoint serves. If you restrict by IP address, for **IP allowlist**, enter the IP addresses and ranges that this endpoint serves content to. One CIDR block per line.
3. Select **Use CDN authorization** to require that content requests to this endpoint include a valid authorization code.
4. (Optional) For **Secrets role ARN**, enter the ARN for the IAM role that grants MediaPackage access to AWS Secrets Manager. The secrets role ARN must be in this format: `arn:aws:iam::accountID:role/name`
5. (Optional) For **CDN identifier secret ARN**, enter the ARN for the authorization code secret in Secrets Manager that your CDN uses for authorization to access your endpoint. The CDN identifier must be in this format: `arn:aws:secretsmanager:region:accountID:secret:guid`

For information about how this authorization works, see CDN authorization in AWS Elemental MediaPackage.

Stream selection fields

Define the streams to include.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

1. (Optional) For **Stream order**, choose the order that video bitrates are presented to the player.
   - **Original** to sort the output streams in the same order that the incoming source uses.

- **Video bitrate ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
- **Video bitrate descending** to sort the output streams starting with the highest bitrate and ending with the lowest.

2. (Optional) For **Min video bitrate**, enter the minimum bitrate (in bits per second) that video tracks must be at or above to be available for playback from this endpoint.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate (in bits per second) that video tracks must be at or below to be available for playback from this endpoint.

Viewing all endpoints associated with a channel

View all endpoints that are associated with a specific channel to ensure that the content is available in all necessary stream formats.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to view the endpoints that are associated with a channel. For information about viewing endpoints through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To view a channel's endpoints (console)

1. Access the channel that the endpoint is associated to, as described in Viewing channel details.

   MediaPackage displays all existing endpoints as a table or as individual cards.
2. (Optional) Choose **Preferences** to adjust your viewing preferences (such as page size and properties that are displayed).

Viewing a single endpoint

View the details about a specific endpoint to obtain its playback URL and to view the packaging settings that it's currently using.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to view the details of an endpoint. For information about viewing endpoint details through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To view a single endpoint's details (console)

1. Access the channel that the endpoint is associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, choose the endpoint ID to view details such as package information and playback preview. For downstream device requests, you must provide the endpoint URL from the **Endpoint URL** field or the CloudFront CDN URL.

Editing an endpoint

Edit the packaging preferences on an endpoint to optimize the viewing experience. You can't change the packager type after you save an endpoint. To serve content with a different packager, create a different endpoint.

If you edited the channel to enable Amazon CloudFront distribution creation from the AWS Elemental MediaPackage console, you can also edit the endpoint to add an origin to the distribution (if you didn't already add one through alternate means). When you save the edited endpoint, MediaPackage automatically works with CloudFront to create the origin.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to change an endpoint's settings. For information about editing an endpoint through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

When you're editing an endpoint, don't put sensitive identifying information like customer account numbers into free-form fields such as the **Name** field. This includes when you work with MediaPackage using the console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

To edit an endpoint (console)

1. Access the channel that the endpoint is associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, choose the endpoint ID and choose **Manage endpoints**.
3. Edit the endpoint options that you want to change.

   For information about endpoint attributes, see Creating an endpoint.

4. Choose **Save**.

Deleting an endpoint

Endpoints can serve content until they're deleted. Delete the endpoint if it should no longer respond to playback requests. You must delete all endpoints from a channel before you can delete the channel.

> **Warning**
>
> If you delete an endpoint, the playback URL stops working.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to delete an endpoint. For information about deleting an endpoint through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To delete an endpoint (console)

1. Access the channel that the endpoint is associated with, as described in Viewing channel details.
2. On the details page for the channel, under **Origin endpoints**, select the origin endpoint that you want to delete.
3. Select **Delete**.
4. In the **Delete endpoints** confirmation dialog box, choose **Delete**.

Previewing an endpoint

Preview an endpoint's playback to ensure that AWS Elemental MediaPackage is receiving the content stream and can package it. The preview is helpful for avoiding playback failures after the endpoint is published and for troubleshooting later if there are any playback issues.

You can use the MediaPackage console to preview playback from the endpoint.

To preview an endpoint's playback (console)

1. Access the channel that the endpoint is associated with, as described in Viewing channel details.

2. On the details page for the channel, under **Origin endpoints**, select the origin endpoint that you want to preview.
3. To preview playback, do one of the following:

   - Choose **Preview** to play content with the embedded player.
   - Choose **QR code** to view and scan the QR code for playback on a compatible device.

Delivering VOD content from AWS Elemental MediaPackage

AWS Elemental MediaPackage uses the following resources for video on demand (VOD) content:

- *Packaging groups* hold one or more packaging configurations. The group enables you to apply multiple output configurations to an asset at the same time. You can associate a group to multiple assets so that they all have the same configurations for their outputs.
- *Packaging configurations* tell MediaPackage how to package the output from an asset. In the configuration, you define encryption, bitrate, and packaging settings.
- *Assets* ingest your source content and dynamically apply packaging configurations in response to playback requests.

  For supported VOD inputs and codecs, see VOD supported codecs and input types.

The following sections describe how to use these resources to manage VOD content in MediaPackage.

Topics

Working with packaging groups in AWS Elemental MediaPackage

A packaging group holds one or more packaging configurations. When a packaging group is associated with an asset, the packaging configurations define the outputs that are available from the asset. You can associate multiple assets with one packaging group. This enables you to apply the same configurations to multiple assets.

Topics

- Editing a packaging group
- Deleting a packaging group
- Adding a packaging configuration to a packaging group

Creating a packaging group

Create a packaging group to hold all of the packaging configurations for an asset. The packaging group, for example, tells AWS Elemental MediaPackage that an asset is available for output to devices that support Apple HLS and DASH-ISO.

When you create a packaging group, you have the option to enable CDN authorization. For more information about CDN authorization, see CDN authorization in AWS Elemental MediaPackage.

To create a packaging group, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. information about creating a packaging group with the AWS CLI or MediaPackage API, see Packaging_groups in the *AWS Elemental MediaPackage VOD API Reference*.

When you're creating a packaging group, don't put sensitive identifying information like customer account numbers into free-form fields, such as the **ID** field. This applies when you're using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

To create a packaging group (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose **Create group**.
4. In the **Creating packaging group** dialog box, do the following:

    1. For **ID**, enter a name that describes the packaging group. The ID is the primary identifier for the group, and must be unique for your account in this AWS Region.
    2. Choose **Create**.

MediaPackage displays the new packaging group's details page.

If you exceed the quotas for your account when you're creating a packaging group, you get an error. If you get an error similar to Too many requests, please try again. Resource limit exceeded, either you have exceeded the API request quotas, or you have already reached the maximum

number of packaging groups allowed on your account. If this is your first group, or if you think you mistakenly received this error, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

Viewing packaging group details

You can view all packaging groups that are configured in AWS Elemental MediaPackage or the details of a specific packaging group, including the packaging configurations that are associated with it.

To view packaging group details, you can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API. information about viewing a packaging group with the AWS CLI or MediaPackage API, see Packaging_groups id in the *AWS Elemental MediaPackage VOD API Reference*.

To view packaging groups (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.

   All groups are displayed on the console.
3. To view more information about a specific packaging group, choose the name of the group.

   MediaPackage displays summary information, such as the assets associated with this packaging group.

Editing a packaging group

Edit the packaging group to configure access control settings.

> **Note**
>
> You can't edit the packager group ID after the packaging group is created. If you want to change the packaging group ID, you must create a new packaging group.

You can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API to edit a packaging group's access control settings. For information about editing a packaging group using the AWS CLI or MediaPackage API, see the MediaPackage VOD API reference.

Deleting a packaging group

To stop AWS Elemental MediaPackage from delivering more content from an asset, delete the packaging group. Before you can delete the packaging group, you must delete the group's packaging configurations and any assets that use the group.

- To delete a packaging configuration, see Deleting a packaging configuration.
- To delete an asset, see Deleting an asset.

To delete a packaging group, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. For information about deleting a packaging group with the AWS CLI or MediaPackage API, see Packaging_groups id in the *AWS Elemental MediaPackage VOD API Reference*.

To delete a packaging group (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the packaging group that you want to delete.
4. Choose **Delete**.
5. In the **Packaging group delete** dialog box, choose **Delete** to finish deleting the packaging group.

Adding a packaging configuration to a packaging group

To define how AWS Elemental MediaPackage formats outputs from an asset, add a packaging configuration to a packaging group.

To add a packaging configuration to a packaging group, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. For more information about adding a packaging configuration with the AWS CLI or MediaPackage API, see Packaging_configurations in the *AWS Elemental MediaPackage VOD API Reference*.

For instructions on adding packaging configurations to a packaging group from the MediaPackage console, see Creating a packaging configuration.

Working with packaging configurations in AWS Elemental MediaPackage

A packaging configuration defines a single delivery point for an asset. The configuration holds all of the information that's needed for AWS Elemental MediaPackage to integrate with a player or content delivery network (CDN), such as Amazon CloudFront. Configure the configuration to output content in one of the available stream formats:

- Apple HLS – Packages content to Apple HTTP Live Streaming (HLS)
- Microsoft Smooth – Packages content for Microsoft Smooth Streaming players
- Common Media Application Format (CMAF) – Packages content to devices that support Apple HLS fragmented MP4 (fMP4)
- DASH-ISO – Packages content for the DASH-ISO ABR streaming protocol

The packaging configuration also holds information about digital rights management (DRM) and encryption integration, bitrate presentation order, and more.

Topics

Creating a packaging configuration

Create a packaging configuration to define how AWS Elemental MediaPackage prepares content for delivery from an asset.

To create a packaging configuration, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. For information about creating a packaging configuration with the AWS CLI or MediaPackage API, see Packaging_configurations in the *AWS Elemental MediaPackage VOD API Reference*.

When you're creating a packaging configuration, don't put sensitive identifying information like customer account numbers into free-form fields, such as the **ID** field. This applies when

you're using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

Topics

Creating an HLS packaging configuration

Create a packaging configuration that formats content for devices that support Apple HLS.

To create an Apple HLS packaging configuration (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the group that will contain the configuration that you're creating.
4. On the details page for the packaging group, under **Packaging configurations**, choose **Manage configurations**.
5. On the **Manage packaging configurations** page, under **Packaging configurations**, choose **Add** and select **New config**.
6. Complete the fields as described in the following topics:
   - General settings fields
   - Manifest settings fields
   - Stream selection fields
   - Encryption fields
7. Choose **Save**.

If you exceed the quotas for your account when you're creating a packaging configuration, you get an error. If you get an error similar to Too many requests, please try again. Resource limit exceeded,

either you have exceeded the API request quotas, or you have already reached the maximum number of packaging groups allowed on your account. If this is your first group, or if you think you mistakenly received this error, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

General settings fields

Provide general settings that apply to the entire packaging configuration.

1. For **ID**, enter a name that describes the configuration. The ID is the primary identifier for the configuration, and must be unique for your account in the AWS Region.
2. For **Package type**, choose **Apple HLS**.
3. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.

Manifest settings fields

Specify the format of the manifest that AWS Elemental MediaPackage delivers from an asset that uses this packaging configuration.

1. (Optional) For **Manifest name**, enter a short string that will be appended to the endpoint URL. The manifest name creates a unique path to this endpoint. If you don't enter a value, MediaPackage uses the default manifest name, *index*.
2. (Optional) Select **Include IFrame-only streams** to include an additional I-frame only stream along with the other tracks in the manifest. MediaPackage generates an I-frame only stream from the first rendition in the manifest. The service inserts `EXT-I-FRAMES-ONLY` tags in the output manifest, and then generates and includes an I-frames only playlist in the stream. This playlist enables player functionality like fast forward and rewind.
3. (Optional) Select **Use audio rendition groups** to group all audio tracks into a single HLS rendition group. For more information about rendition groups, see Rendition groups reference in AWS Elemental MediaPackage.
4. (Optional) Select **Repeat EXT-X-KEY** if you want the service to repeat the key before every segment of the manifest. By default, the key is written just once, after the header and before the segments. If you select **Repeat EXT-X-KEY**, the manifest is written as header, key, segment, key, segment, key, and so on, with every segment preceded by the key. Set this according to the

needs of the player. Selecting this option might result in an increase in client requests to the DRM server.

5. (Optional) Select **Include DVB subtitles** to passthrough digital video broadcasting (DVB) subtitles into the output.
6. (Optional) For **Program date/time interval** enter the interval (in seconds) at which MediaPackage should insert the `EXT-X-PROGRAM-DATE-TIME` tags in the manifest.

   The `EXT-X-PROGRAM-DATE-TIME` tag synchronizes the stream to the wall clock, enabling functionality like viewer seek in the playback timeline and time display on the player.
7. (Optional) For **Ad markers**, choose how ad markers are included in the packaged content.

   Choose from the following:

   - **None** – Omit all SCTE-35 ad markers from the output.
   - **Passthrough** – Copy the SCTE-35 ad markers directly from the input HLS input stream to the output.
   - **SCTE-35 Enhanced** – Generate ad markers and blackout tags based on the SCTE-35 input messages from the input stream.

Stream selection fields

Limit what incoming bitrates are available for playback and sort the streams in the output of an asset that uses this packaging configuration.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

To set minimum and maximum bitrates and sort the output, select **Enable stream selection** and complete the additional fields as follows:

1. (Optional) For **Stream order**, choose from the following:
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Descending** to sort the output streams starting with the highest bitrate and ending with the lowest.

2. (Optional) For **Min video bitrate**, enter the minimum bitrate threshold (in bits per second) that video tracks must must be at or above to be available for playback from this endpoint. This ensures that tracks are *at least* a certain bitrate.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate threshold (in bits per second) that video tracks must be at or below to be available for playback from this endpoint. This ensures that tracks are *no more than* a certain bitrate.

Encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM solution provider, and be set up to use encryption. For information, see the section called “Content encryption and DRM”.

To serve content with copyright protection, select **Enable encryption** and complete the additional fields as follows:

1. For **Encryption method**, choose **Sample-AES** for Apple HLS FairPlay or choose **AES-128** for Apple HLS AES-128.
2. (Optional) For **Constant initialization vector** enter a 128-bit, 16-byte hex value represented by a 32-character string, to be used with the key for encrypting content.
3. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

   The following example shows a URL.

```
https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
```

4. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

   The following example shows a role ARN.

   ```
   arn:aws:iam::444455556666:role/SpekeAccess
   ```

5. For **System IDs**, enter unique identifiers for your streaming protocol and DRM system. Provide up to three IDs for CMAF, two IDs for DASH, and exactly one for the other streaming protocols. If you provide more than one system ID, enter one per line and choose **Add**. For a list of common system IDs, see DASH-IF System IDs. If you don't know your IDs, ask your DRM solution provider.

Creating a DASH packaging configuration

Create a packaging configuration that formats content for devices that support DASH-ISO.

To create a DASH-ISO packaging configuration (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the group that will contain the configuration that you're creating.
4. On the details page for the packaging group, under **Packaging configurations**, choose **Manage configurations**.
5. On the **Manage packaging configurations** page, under **Packaging configurations**, choose **Add** and **New config**.
6. Complete the fields as described in the following topics:

   - General settings fields
   - Manifest settings fields
   - Stream selection fields
   - Encryption fields
7. Choose **Save**.

If you exceed the quotas for your account when you're creating a packaging configuration, you get an error. If you get an error similar to Too many requests, please try again. Resource limit exceeded,

either you have exceeded the API request quotas, or you have already reached the maximum number of packaging groups allowed on your account. If this is your first group, or if you think you mistakenly received this error, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

General settings fields

Provide general settings that apply to the entire packaging configuration.

1. For **ID**, enter a name that describes the configuration. The ID is the primary identifier for the configuration, and must be unique for your account in the Region.
2. For **Package type**, choose **DASH-ISO**.
3. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.

Manifest settings fields

Specify the format of the manifest that AWS Elemental MediaPackage delivers from an asset that uses this packaging configuration.

1. (Optional) For **Manifest name**, enter a short string that will be appended to the endpoint URL. The manifest name helps to create a unique path to this endpoint. If you don't enter a value, the default manifest name is *index*.
2. (Optional) For **Min buffer time**, enter the minimum amount of time (in seconds) that a player must keep in the buffer. If network conditions interrupt playback, the player will have additional buffered content before playback fails, allowing for recovery time before the viewer's experience is affected.
3. (Optional) For **Profile**, specify a DASH profile.

   Choose from the following:

   - **None** – the output doesn't use a DASH profile
   - **Hbbtv 1.5** – the output is compliant with HbbTV v1.5. For information about HbbTV v1.5, see the HbbTV specification website.
4. (Optional) For **Manifest layout**, choose if you want MediaPackage to serve a full, compact, or DRM top level compact manifest in response to playback requests.

- If you choose **Full**, MediaPackage presents the `SegmentTemplate` and `SegmentTimeline` tags for every `Representation` in the manifest.
- If you choose **Compact**, MediaPackage combines duplicate `SegmentTemplate` tags and presents them at the start of the manifest. This shortens the manifest and makes it easier for some devices to process it.
- If you choose **DRM top level compact**, MediaPackage places content protection elements at the media presentation description (MPD) level and are referenced at the `AdaptationSet` level. You can choose this option only if you're using the SPEKE Version 1.0 or 2.0 SHARED preset.

For more information about the manifest layout options, see Compacted DASH manifests.

5. (Optional) For **Segment template format**, choose how MediaPackage and playback requests refer to each segment.
   - If you choose **Number with timeline**, MediaPackage uses the `$Number$` variable to refer to the segment in the `media` attribute of the `SegmentTemplate` tag. The value of the variable is the sequential number of the segment. `SegmentTimeline` is included in each segment template.
   - If you choose **Time with timeline**, MediaPackage uses the `$Time$` variable to refer to the segment. The value of the variable is the timestamp of when on the manifest timeline the segment starts. `SegmentTimeline` is included in each segment template.
   - If you choose **Number with duration**, MediaPackage uses the `$Number$` variable and replaces the `SegmentTimeline` objects with a `duration` attribute in the segment template.

   > **Note**
   >
   > This option isn't supported in combination with multi-period DASH.

   For more information about the formatting options of the `SegmentTemplate` tag, see DASH manifest segment template format.
6. (Optional) Select **Include IFrame-only streams** to include an additional I-frame only stream along with the other tracks in the manifest. MediaPackage generates an I-frame only stream from the first rendition in the manifest. This playlist enables player functionality like fast forward and rewind.
7. For **Period triggers**, select how MediaPackage creates MPD periods in the DASH output manifest.

- **None** – MediaPackage doesn't create additional periods. It formats the manifest as a single period and doesn't include SCTE-35 markers in the segments.
- **Trigger new periods on ads** – MediaPackage creates and inserts in the manifest multiple periods based on SCTE-35 ad markers from the input content. These periods separate portions of the content, such as setting boundaries between the main content and ad content. For more information about how MediaPackage configures periods in the manifest, see DASH manifest options in AWS Elemental MediaPackage.

> **Important**
>
> Multiple periods are required if you use AWS Elemental MediaTailor for personalized ad insertion in DASH content. For more information about this service, see the AWS Elemental MediaTailor User Guide.

8. For **SCTE markers source**, specify the source of SCTE-35 markers to use from your input HLS content.
   - Select **Segments** to use SCTE-35 markers from input HLS media segments.
   - Select **Manifest** to use SCTE-35 markers, formatted using SCTE-35 Enhanced syntax (`#EXT-OATCLS-SCTE35` tags), from input HLS child manifests. SCTE-35 Elemental and SCTE-35 Daterange syntaxes are not supported.
9. (Optional) Select **Include encoder configuration in segments** for MediaPackage to place your encoder's Sequence Parameter Set (SPS), Picture Parameter Set (PPS), and Video Parameter Set (VPS) metadata in every video segment instead of in the init fragment. This lets you use different SPS/PPS/VPS settings for your assets during content playback.

Stream selection fields

Limit which incoming bitrates are available for playback and sort the streams in the output of an asset that uses this packaging configuration.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

To set minimum and maximum bitrates and sort the output, select **Enable stream selection** and complete the additional fields as follows:

1. (Optional) For **Stream order**, choose from the following:
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Descending** to sort the output streams starting with the highest bitrate and ending with the lowest.
2. (Optional) For **Min video bitrate**, enter the minimum bitrate threshold (in bits per second) that video tracks must be at or above to be available for playback from this endpoint. This ensures that tracks are *at least* a certain bitrate.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate threshold (in bits per second) that video tracks must be at or below to be available for playback from this endpoint. This ensures that tracks are *no more than* a certain bitrate.

Encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM solution provider, and be set up to use encryption. For information, see the section called “Content encryption and DRM”.

To serve content with copyright protection, select **Enable encryption** and complete the additional fields as follows:

1. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

   The following example shows a URL.

```
https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
```

2. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

   The following example shows a role ARN.

   ```
   arn:aws:iam::444455556666:role/SpekeAccess
   ```

3. For **System IDs**, enter unique identifiers for your streaming protocol and DRM system. Provide up to three IDs for CMAF, two IDs for DASH, and exactly one for the other streaming protocols. If you provide more than one system ID, enter one per line and choose **Add**. For a list of common system IDs, see DASH-IF System IDs. If you don't know your IDs, ask your DRM solution provider.
4. (Optional) For **SPEKE version**, choose the SPEKE version that you'd like to use for encryption. SPEKE Version 1.0 is the legacy version that uses CPIX Version 2.0, and supports single key encryption. SPEKE Version 2.0 uses CPIX Version 2.3, and supports multiple key encryption. For more information about using SPEKE with MediaPackage, see Content encryption and DRM in MediaPackage.

   If you select **SPEKE Version 2.0**, then also choose a **Video encryption preset** and an **Audio encryption preset**. The video and audio presets determine which content keys MediaPackage uses to encrypt the audio and video tracks in your stream. For more information about these presets, see SPEKE Version 2.0 presets.

   When using SPEKE Version 2.0, MediaPackage disables key rotation.

Creating a Microsoft Smooth packaging configuration

Create a packaging configuration that formats content for devices that support Microsoft Smooth.

To create a Microsoft Smooth packaging configuration (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the group that will contain the configuration that you're creating.

4. On the details page for the packaging group, under **Packaging configurations**, choose **Manage configurations**.
5. On the **Manage packaging configurations** page, under **Packaging configurations**, choose **Add** and select **New config**.
6. Complete the fields as described in the following topics:
   - General settings fields
   - Manifest settings fields
   - Stream selection fields
   - Encryption fields
7. Choose **Save**.

If you exceed the quotas for your account when you're creating a packaging configuration, you get an error. If you get an error similar to Too many requests, please try again. Resource limit exceeded, either you have exceeded the API request quota, or you have already reached the maximum number of packaging groups allowed on your account. If this is your first group, or if you think you mistakenly received this error, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

General settings fields

Provide general settings that apply to the entire packaging configuration.

1. For **ID**, enter a name that describes the configuration. The ID is the primary identifier for the configuration, and must be unique for your account in the Region.
2. For **Package type**, choose **Microsoft Smooth**.
3. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.

Manifest settings fields

Specify the format of the manifest that AWS Elemental MediaPackage delivers from an asset that uses this packaging configuration.

- (Optional) For **Manifest name**, enter a short string that will be appended to the endpoint URL. The manifest name helps to create a unique path to this endpoint. If you don't enter a value, the default manifest name is *index*.

Stream selection fields

Limit which incoming bitrates are available for playback and sort the streams in the output of an asset that uses this packaging configuration.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

To set minimum and maximum bitrates and sort the output, select **Enable stream selection** and complete the additional fields as follows:

1. (Optional) For **Stream order**, choose from the following:
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Descending** to sort the output streams starting with the highest bitrate and ending with the lowest.
2. (Optional) For **Min video bitrate**, enter the minimum bitrate threshold (in bits per second) that video tracks must be at or above to be available for playback from this endpoint. This ensures that tracks are *at least* a certain bitrate.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate threshold (in bits per second) that video tracks must be at or below to be available for playback from this endpoint. This ensure that tracks are *no more than* a certain bitrate.

Encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more

information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM solution provider, and be set up to use encryption. For information, see the section called "Content encryption and DRM".

To serve content with copyright protection, select **Enable encryption** and complete the additional fields as follows:

1. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

   The following example shows a URL.

   ```
   https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
   ```

2. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

   The following example shows a role ARN.

   ```
   arn:aws:iam::444455556666:role/SpekeAccess
   ```

3. For **System IDs**, enter unique identifiers for your streaming protocol and DRM system. Provide up to three IDs for CMAF, two IDs for DASH, and exactly one for the other streaming protocols. If you provide more than one system ID, enter one per line and choose **Add**. For a list of common system IDs, see DASH-IF System IDs. If you don't know your IDs, ask your DRM solution provider.

Creating a CMAF packaging configuration

Create a packaging configuration that formats content for devices that support Apple HLS fragmented MP4 (fMP4).

To create a CMAF packaging configuration (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.

2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the group that will contain the configuration that you're creating.
4. On the details page for the packaging group, under **Packaging configurations**, choose **Manage configurations**.
5. On the **Manage packaging configurations** page, under **Packaging configurations**, choose **Add** and select **New config**.
6. Complete the fields as described in the following topics:

   - General settings fields
   - Manifest settings fields
   - Stream selection fields
   - Encryption fields
7. Choose **Save**.

If you exceed the quotas for your account when you're creating a packaging configuration, you get an error. If you get an error similar to Too many requests, please try again. Resource limit exceeded, either you have exceeded the API request quotas, or you have already reached the maximum number of packaging groups allowed on your account. If this is your first group, or if you think you mistakenly received this error, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

General settings fields

Provide general settings that apply to the entire packaging configuration.

1. For **ID**, enter a name that describes the configuration. The ID is the primary identifier for the configuration, and must be unique for your account in the Region.
2. For **Package type**, choose **Common Media Application Format (CMAF)**.
3. (Optional) For **Segment duration**, enter the duration (in seconds) of each segment. Enter a value equal to, or a multiple of, the input segment duration. If the value that you enter is different from the input segment duration, AWS Elemental MediaPackage rounds segments to the nearest multiple of the input segment duration.

Manifest settings fields

Specify the format of the manifest that AWS Elemental MediaPackage delivers from an asset that uses this packaging configuration.

1. (Optional) For **Manifest name**, enter a short string that will be appended to the endpoint URL. The manifest name creates a unique path to this endpoint. If you don't enter a value, MediaPackage uses the default manifest name, *index*.
2. (Optional) In stream sets with a single video track, select **Include IFrame-only streams** to include an additional I-frame only stream along with the other tracks in the manifest. MediaPackage inserts `EXT-I-FRAMES-ONLY` tags in the manifest, and then compiles and includes an I-frames only playlist in the stream. This playlist enables player functionality like fast forward and rewind.
3. (Optional) Select **Repeat EXT-X-KEY** if you want the service to repeat the key before every segment of the manifest. By default, the key is written just once, after the header and before the segments. If you select **Repeat EXT-X-KEY**, the manifest is written as header, key, segment, key, segment, key, and so on, with every segment preceded by the key. Set this according to the needs of the player. Selecting this option might result in an increase in client requests to the DRM server.
4. (Optional) For **Program date/time interval**, enter the interval at which MediaPackage should insert the `EXT-X-PROGRAM-DATE-TIME` tags in the manifest.

   The `EXT-X-PROGRAM-DATE-TIME` tag synchronizes the stream to the wall clock, enabling functionality like viewer seek in the playback timeline and time display on the player.
5. (Optional) For **Ad markers**, choose how ad markers are included in the packaged content.

   Choose from the following:

   - **None** – Omit all SCTE-35 ad markers from the output.
   - **Passthrough** – Copy the SCTE-35 ad markers directly from the input HLS input stream to the output.
   - **SCTE-35 Enhanced** – Generate ad markers and blackout tags based on the SCTE-35 input messages from the input stream.
6. (Optional) Select **Include encoder configuration in segments**, for MediaPackage to place your encoder's Sequence Parameter Set (SPS), Picture Parameter Set (PPS), and Video Parameter Set (VPS) metadata in every video segment instead of in the init fragment. This lets you use different SPS/PPS/VPS settings for your assets during content playback.

Stream selection fields

Limit which incoming bitrates are available for playback and sort the streams in the output of an asset that uses this packaging configuration.

The minimum and maximum values take into account only the video bitrates. If the video bitrate is *below the minimum* specified rate, it's *not* included in the output, regardless of the sum of the bitrates for other tracks. Likewise, if the video bitrate is *below the maximum* specified rate, it *is* included in the output, regardless of the sum of the bitrates for other tracks.

To set minimum and maximum bitrates and sort the output, select **Enable stream selection** and complete the additional fields as follows:

1. (Optional) For **Stream order**, choose from the following:
   - **Original** to sort the output streams in the same order that the incoming source uses.
   - **Ascending** to sort the output streams starting with the lowest bitrate and ending with the highest.
   - **Descending** to sort the output streams starting with the highest bitrate and ending with the lowest.
2. (Optional) For **Min video bitrate**, enter the minimum bitrate threshold (in bits per second) that video tracks must be at or above to be available for playback from this endpoint. This ensures that tracks are *at least* a certain bitrate.
3. (Optional) For **Max video bitrate**, enter the maximum bitrate threshold (in bits per second) that video tracks must be at or below to be available for playback from this endpoint. This ensures that tracks are *no more than* a certain bitrate.

Encryption fields

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

> **Note**
>
> To encrypt content, you must have a DRM solution provider, and be set up to use encryption. For information, see the section called “Content encryption and DRM”.

To serve content with copyright protection, Select **Enable encryption** and complete the additional fields as follows:

1. For **System IDs**, enter unique identifiers for your streaming protocol and DRM system. Provide up to two system IDs. If you provide more than one system ID, enter one per line. If you do not know your IDs, ask your DRM solution provider.
2. For **URL**, enter the URL of the API Gateway proxy that you set up to talk to your key server. The API Gateway proxy must reside in the same AWS Region as MediaPackage.

   The following example shows a URL.

   ```
   https://1wm2dx1f33.execute-api.us-west-2.amazonaws.com/SpekeSample/copyProtection
   ```

3. (Optional) For **SPEKE version**, choose the SPEKE version that you'd like to use for encryption. SPEKE Version 1.0 is the legacy version that uses CPIX Version 2.0, and supports single key encryption. SPEKE Version 2.0 uses CPIX Version 2.3, and supports multiple key encryption. For more information about using SPEKE with MediaPackage, see Content encryption and DRM in MediaPackage.

   If you select **SPEKE Version 2.0**, then also choose a **Video encryption preset** and an **Audio encryption preset**. The video and audio presets determine which content keys MediaPackage uses to encrypt the audio and video tracks in your stream. For more information about these presets, see SPEKE Version 2.0 presets.

   When using SPEKE Version 2.0, MediaPackage disables key rotation.
4. (Optional) For **Constant initialization vector** enter a 128-bit, 16-byte hex value represented by a 32-character string, to be used with the key for encrypting content.
5. For **Role ARN**, enter the Amazon Resource Name (ARN) of the IAM role that provides you access to send your requests through API Gateway. Get this from your DRM solution provider.

   The following example shows a role ARN.

```
arn:aws:iam::444455556666:role/SpekeAccess
```

Viewing packaging configuration details

To ensure that the content is available in all necessary stream formats, view all packaging configurations that are associated with a specific packaging group or with an asset.

To view packaging configurations, you can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API. For information about viewing a packaging configuration with the AWS CLI or MediaPackage API, see Packaging_configurations id in the *AWS Elemental MediaPackage VOD API Reference*.

To view packaging configurations in a packaging group (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the group that contains the configurations that you want to view.

   The **Packaging configurations** section displays all of the configurations that are in this group.
4. To view the details of a specific packaging configuration, choose the **Id** of that configuration.

MediaPackage displays summary information, such as the assets associated with this packaging configuration.

To view all packaging configurations associated with an asset (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Assets**.
3. On the **Assets** page, choose the asset that you want to audit.

   The **Playback details** section displays all of the configurations that are associated with this asset. On this page, you can view the playback status of the asset in the **Status** column. The available statuses are as follows:

- **Not processed** - The asset hasn't been processed yet.

- **Processing** - MediaPackage is processing the asset. The asset isn't available for playback yet.
- **Processed** - The asset has been processed, and is available for playback.
- **Failed** - Processing failed.

> **Note**
>
> Status information isn't available for most assets ingested before September 30th, 2021.

Editing a packaging configuration

You can't edit a packaging configuration. If you need to make changes, create a new configuration and delete the original.

- To create a configuration, see Creating a packaging configuration.
- To delete a configuration, see Deleting a packaging configuration.

Deleting a packaging configuration

To remove a playback endpoint from an asset, delete the packaging configuration.

To delete a packaging configuration, you can use the AWS Elemental MediaPackage console, the AWS CLI, or the MediaPackage API. For information about deleting a packaging configuration with the AWS CLI or MediaPackage API, see Packaging_configurations id in the *AWS Elemental MediaPackage VOD API Reference*.

To delete a packaging configuration (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Packaging groups**.
3. On the **Packaging groups** page, choose the group that contains the configuration that you're deleting.
4. On the details page for the packaging group, under **Packaging configurations**, choose either the packaging configuration ID of the configuration that you're deleting and choose **Delete** or choose **Manage configurations**, **Actions**, **Delete**.
5. On the **Delete packaging configurations** page, choose **Delete**.

Working with assets in AWS Elemental MediaPackage

An asset holds all of the information that MediaPackage requires to ingest file-based video content from a source such as Amazon S3. Through the asset, MediaPackage ingests and dynamically packages content in response to playback requests. The configurations associated with the asset determine how it can be packaged for output.

After you ingest an asset, AWS Elemental MediaPackage provides a URL for each playback configuration associated with the asset. This URL is fixed for the lifetime of the asset, regardless of any failures that might happen over time. Downstream devices use the URL to send playback requests.

For supported VOD inputs and codecs, see VOD supported codecs and input types.

Topics

Ingesting an asset

To ingest source content, create an asset in AWS Elemental MediaPackage. When MediaPackage ingests content, it creates a unique playback URL for every packaging configuration that's associated with the asset.

> **Important**
>
> To ingest an asset, MediaPackage must have permissions to access the Amazon S3 bucket where the source content is stored. To create a role that gives MediaPackage the right permissions, see Allowing AWS Elemental MediaPackage to access other AWS services.

To create an asset, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. For information about creating a packaging configuration with the AWS CLI or MediaPackage API, see Assets in the *AWS Elemental MediaPackage VOD API Reference*.

When you're creating an asset, don't put sensitive identifying information like customer account numbers into free-form fields, such as the **ID** field. This applies when you're using the MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

To ingest an asset (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Assets**.
3. On the **Assets** page, choose **Ingest assets**.
4. On the **Ingest assets** page, complete the fields as described in the following topics:
   - Asset access fields
   - Asset details fields
   - Packaging settings field
5. Choose **Ingest assets**.

Ingesting a VOD asset is an asynchronous action. The time it takes before an asset becomes available for playback can vary based on several factors, such as asset duration and asset complexity. You can track when a VOD asset is ready for playback by monitoring the CloudWatch `VodAssetPlayable` events that MediaPackage sends when the asset is ready for playback. For more information, see VOD Playback Events.

If you exceed the quotas for your account when you're creating a packaging configuration, you get an error. If you get an error similar to Too many requests, please try again. Resource limit exceeded, either you have exceeded the API request quotas, or you have already reached the maximum number of packaging groups allowed on your account. If this is your first group, or if you think you mistakenly received this error, use the Service Quotas console to request quota increases. For more information about quotas in MediaPackage, see Quotas in AWS Elemental MediaPackage.

Asset access fields

The following fields describe how AWS Elemental MediaPackage accesses the source content in your Amazon S3 bucket. MediaPackage must have permissions to access the bucket. To create an IAM role with the right permissions, see Allowing AWS Elemental MediaPackage to access other AWS services.

1. For **Amazon S3 bucket name**, choose from a list of buckets that MediaPackage has detected in your account or enter the name of the bucket. The Amazon S3 bucket holds the source content that MediaPackage ingests and packages for playback. If you entered the name of the bucket, MediaPackage doesn't have visibility into this bucket, so it can't tell if the bucket is compatible or not.

   > **Note**
   >
   > If you don't have permissions to view Amazon S3 buckets, MediaPackage doesn't display any options. Contact your AWS administrator or enter the bucket name manually in the **Specify bucket name** field.

2. For **IAM role**, choose the IAM role with the MediaPackage permissions to read from the Amazon S3 bucket.

   - To choose from a list of roles that MediaPackage has detected on your account, choose **Use existing role** and choose the role.

     > **Note**
     >
     > If you don't have permissions to view IAM roles, MediaPackage doesn't display any options. Contact your AWS administrator or enter the role ARN manually in the **Specify custom role name** field.

   - To use a role that MediaPackage hasn't detected, choose **Specify custom role name** and enter the custom ARN of the role. Because MediaPackage doesn't have visibility into this role, it can't tell if the role provides the correct permissions or not.

Asset details fields

The following fields describe the source content that this asset uses.

If you have multiple sources for this asset, choose **Add asset** and complete the fields. Do this for all source contents.

> **Important**
>
> Source content must be in a .smil (MP4) or .m3u8 (HLS/TS) file format.

1. For **Filename**, Enter the full path to either the .smil manifest (MP4) or the .m3u8 parent playlist (HLS) within your Amazon S3 bucket, including the name of the source content. You don't need to enter the bucket name because you chose it in **S3 bucket name** field. For example, if your content is called `lion_movie.m3u8` and is in a subdirectory called `thursday_night` in a bucket called `movies`, you would enter the following in the **Filename** field:

   ```
   thursday_night/lion_movie.m3u8
   ```

   For more information about using .smil manifests with MediaPackage, see Requirements for .smil manifests.
2. For **ID**, enter a name that describes the asset. The ID is the primary identifier for the asset, and must be unique for your account in this Region. Supported characters are letters, numbers, underscores (_), and dashes (-).
3. (Optional) For **Resource ID**, enter an identifier for the content. When you're using SPEKE, the resource ID is the identifier that your key server uses to reference the content. MediaPackage sends the ID to the key server to identify the current asset. How unique you make the ID depends on the level of access controls you need. The service doesn't allow you to use the same ID for two simultaneous encryption processes. The resource ID is also known as the content ID.

   Example

   ```
   MovieNight20171126093045
   ```

Packaging settings field

The following field determines how AWS Elemental MediaPackage packages outputs from this asset.

- For **Packaging group**, choose the group that holds the configurations that you want to use for this asset. The packaging group determines which packaging configurations MediaPackage uses when it packages content to fulfill playback requests.

Viewing asset details

You can view all assets that are configured in AWS Elemental MediaPackage or the details of a specific asset, including the packaging configurations that are associated with it.

To view asset details, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. For information about creating a packaging configuration with the AWS CLI or MediaPackage API, see Assets id in the *AWS Elemental MediaPackage VOD API Reference*.

To view assets (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Assets**.

   All assets are displayed on the console.
3. To view more information about a specific asset, choose the name of the asset.

   MediaPackage displays summary information, such as the packaging configurations associated with this packaging asset and their playback URLs.

Editing an asset

You can't edit an asset. To make changes, ingest the asset again and delete the original.

- To ingest an asset, see Creating a packaging configuration.
- To delete an asset, see Deleting a packaging configuration.

Deleting an asset

To remove the packaging group URLs and to stop AWS Elemental MediaPackage from delivering further content, delete an asset.

To delete an asset, you can use the MediaPackage console, the AWS CLI, or the MediaPackage API. For information about creating a packaging configuration with the AWS CLI or MediaPackage API, see Assets id in the *AWS Elemental MediaPackage VOD API Reference*.

To delete an asset (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Video on demand**, choose **Assets**.
3. On the **Assets** page, select the asset that you want to delete.
4. Choose **Delete**.

Creating live-to-VOD assets with AWS Elemental MediaPackage

A live-to-VOD (video on demand) asset is a portion of a live stream that's been extracted and saved for playback later. For example, you might save clips from a game for a highlight reel, or a clip of a broadcast show to use later in advertisements for the show.

To create a live-to-VOD asset in MediaPackage, create a harvest job resource. The harvest job is a request that you create for MediaPackage to extract a portion of a live stream and save the clip as a live-to-VOD asset in an Amazon S3 bucket. The job runs once, then MediaPackage keeps a record of it on your account for 90 days. This record is for reference purposes only. You can't delete or modify it.

> **Important**
>
> To create live-to-VOD assets, you must allow MediaPackage to access and save to an Amazon S3 bucket. For instructions, see Allowing AWS Elemental MediaPackage to access other AWS services.

The following topics provide more information about live-to-VOD assets in MediaPackage.

Topics

Live-to-VOD requirements

Keep in mind these requirements when you're creating live-to-VOD assets in AWS Elemental MediaPackage.

Channel requirements

Create a new MediaPackage channel to harvest content from when there is a change to the stream in the upstream encoder (such as changes to the stream name, type, or codec). If you don't use a

new channel and the start and end times of the harvest job span the change, the harvest could behave in unexpected ways.

Endpoint requirements

The endpoint that you're harvesting the live-to-VOD asset from must meet these requirements:

- Startover must be enabled and have a **startover window** of 14 days or less. To check or change the size of the window, see the section called “Viewing a single endpoint”.
- Your endpoint must serve either clear (unencrypted) or encrypted DASH or HLS content.
- For DASH endpoints - Your DASH endpoint must use either the **Number with timeline** or **Time with timeline** segment template format. For information about creating DASH endpoints, see Creating a DASH endpoint.
- MediaPackage VOD doesn't currently support the ingestion of encrypted assets. If you're using your harvested assets in an MediaPackage video on demand workflow and your endpoint is encrypted, create an unencrypted shadow endpoint on the same channel. To do this, deselect **allow origination** so that the new endpoint can't be used for playback. MediaPackage creates the URL for endpoints that don't have origination enabled, but MediaPackage responds with an error to playback requests sent to this endpoint.

Live-to-VOD asset requirements

The live-to-VOD asset must meet these requirements:

- Its start time must fall on or after the encoder's start time.
- Its start and end times must be within the startover window on the endpoint.
- Its duration must not exceed the maximum live-to-VOD manifest length, which is 24 hours.

How live-to-VOD works

In the processing flow for live-to-VOD (video on demand) content, AWS Elemental MediaPackage extracts a clip of video from a live content stream. MediaPackage saves this clip as a live-to-VOD asset in Amazon S3. You can use the VOD content processing functionality in MediaPackage to deliver the asset to playback devices, or you can use a VOD encoding service that supports HLS or DASH inputs.

Here's an overview of the main steps:

1. You create a channel and endpoint to ingest a live stream and package it for HLS or DASH output. The endpoint must meet the requirements outlined in Live-to-VOD requirements.
2. You create a harvest job, which defines the live-to-VOD asset that you're extracting from the live stream. The asset must also meet the requirements outlined in *Live-to-VOD Requirements*.
3. MediaPackage harvests the timeframe that you indicated in the harvest job. The asset is segment-accurate. This means that if you have a 6-second segment, and the harvest job has a start time of three seconds into the segment, the asset will start three seconds earlier, at the start of the segment.

   After MediaPackage harvests the asset, it saves the asset in the Amazon S3 bucket that you indicated in the harvest job. MediaPackage creates a directory within that bucket and names the parent manifest based on the information that you provided in the **Manifest key** on the harvest job. For example, if the manifest key is **`thursdaynight/highlights/index.m3u8`**, MediaPackage creates a `thursdaynight/highlights` directory in your Amazon S3 bucket and names the parent manifest `index.m3u8`.

   MediaPackage creates a CloudWatch event when the harvest job completes or fails. For information about events for harvest jobs, see Harvest job notification events.

   MediaPackage keeps a read-only reference of the job on your account for 90 days. After 90 days, MediaPackage deletes the record of the job from your account. At this time, if your workflow requires it, you can reuse the identifier from harvest job.
4. At this point, the live-to-VOD functionality in MediaPackage is complete. The live-to-VOD asset is in your Amazon S3 bucket, and you can do with it what your workflow requires. For example, you can use the VOD functionality in MediaPackage or an encoding service to make the asset available for playback.

> **Important**
>
> Create a new MediaPackage channel to harvest content from when there is a change to the stream in the upstream encoder (such as changes to the stream name, type, or codec). If you don't use a new channel and the start and end times of the harvest job span the change, the harvest could behave in unexpected ways.

Working with harvest jobs

A harvest job represents a request to extract a live-to-VOD (video on demand) asset from an endpoint for a specific timeframe in the past. AWS Elemental MediaPackage uses information from the harvest job to determine the start and end times of the asset, and where to store it after the harvest job is complete.

A harvest job runs only once after it's been created. MediaPackage keeps a record of the job on your account for reference only. You can't modify or delete a record once you've created the harvest job.

Topics

Creating a harvest job

Create a harvest job to extract a live-to-VOD asset from an encrypted or clear (unencrypted) live HLS or DASH stream.

> **Important**
>
> To run a harvest job and save the live-to-VOD asset, MediaPackage must have permissions to access and write to the Amazon S3 bucket where the asset will be stored. To create a role that gives MediaPackage the right permissions, see Allowing AWS Elemental MediaPackage to access other AWS services.

You can use the MediaPackage console, the AWS CLI, or the MediaPackage API to create a harvest job. For information about creating a job through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

When you're creating a harvest job, don't put sensitive identifying information like customer account numbers into free-form fields, such as the **ID** field. This applies when you're using the

MediaPackage console, MediaPackage API, AWS CLI, or AWS SDKs. Any data that you enter into MediaPackage might get picked up for inclusion in diagnostic logs or Amazon CloudWatch Events.

To create a harvest job (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. In the navigation pane, under **Live**, choose **Harvest jobs**.
3. On the **Harvest jobs** page, choose **Create harvest job**.
4. On the **Create harvest job** page, complete the fields as described in the following topics:
   - Basic details fields
   - Start and end date and time fields
   - Destination fields
5. Choose **Create**.

Basic details fields

The basic details of a harvest job define its identifier and the source for the live-to-VOD asset.

1. For **ID**, enter a name that describes the harvest job. The ID is the primary identifier for the harvest job. You can reuse the ID when the harvest job expires from your account. Supported characters are letters, numbers, underscores (_), and dashes (-).
2. For **Origin endpoint**, select the endpoint that serves the live stream that you're harvesting the live-to-VOD asset from.

   Note the following considerations.

   - Your harvest job must fall within your MediaPackage endpoint's **startover window**. The startover window determines the time frame that assets can be harvested from your endpoint. For example, if your endpoint has a startover window of three days, you can harvest your asset anytime within that time frame.

     A MediaPackage endpoint can have a startover window between zero and 14 days. To adjust your endpoint's startover window, see Viewing a single endpoint.
   - Your harvested live-to-VOD asset can have a maximum duration of 24 hours. To set the live-to-VOD asset duration, see Start and end date and time fields in this chapter.
   - Your endpoint must serve either clear (unencrypted) or encrypted DASH or HLS content.

- MediaPackage VOD doesn't currently support the ingestion of encrypted assets. If you're using your harvested assets in an MediaPackage video on demand workflow and your endpoint is encrypted, create an unencrypted shadow endpoint on the same channel. To do this, deselect **allow origination** so that the new endpoint can't be used for playback. MediaPackage creates the URL for endpoints that don't have origination enabled, but MediaPackage responds with an error to playback requests sent to this endpoint. For more information, see Creating live-to-VOD assets with AWS Elemental MediaPackage.

Start and end date and time fields

The start and end date and time information defines the time range for the harvest job. The maximum duration of the harvest job is 24 hours. Times are based on the program date time (PDT) from the encoder. To ensure synchronization between the encoder and the playback device, be sure to include `EXT-X-PROGRAM-DATE-TIME` tags in the endpoint that you're harvesting from. For instructions, see Packager settings fields.

> **Note**
>
> The live-to-VOD asset timing is accurate up to the segment. This means that if you indicate a start or end time that falls within a segment, MediaPackage includes the entire segment in the asset. If you have a 3-second segment and that start time falls on the third second in the segment, the asset will begin two seconds earlier, at the start of the segment.

1. For **Date and time format**, choose the format that you're using to indicate the start and end times of the live-to-VOD asset.
   - **Local time** - the date and time is formatted according to the settings of your current browser session. Local time uses a 24-hour clock.
   - **Epoch seconds** - the date and time is formatted in seconds since the epoch.
   - **ISO-8601** - the date and time is formatted according to the ISO-8601 standard.
2. For **When the live-to-VOD asset begins**, enter when the live-to-VOD asset begins. The asset's begin time must be at the same time or after the live event started. The start time must also be within the startover window on the endpoint. If the endpoint has a window of 5 hours and the start time is 6 hours ago, the harvest job fails.
3. For **When the live-to-VOD asset ends**, enter when the live-to-VOD asset ends. The length of the asset can't exceed the startover window on the endpoint. If the endpoint has a window of

5 hours and your start time is 2019/07/29 07:15:00, the end time can't be after 2019/07/29 12:15:00. The end time must also be in the past.

Destination fields

The destination information defines how MediaPackage saves the live-to-VOD asset after it has been harvested from the live stream.

1. For **IAM role ARN**, enter the ARN for the IAM role that provides MediaPackage access to read and write from your Amazon S3 bucket where the live-to-VOD asset will be stored. This is the role that you created in Allowing AWS Elemental MediaPackage to access other AWS services.
2. For **Amazon S3 bucket name**, enter the bucket where you want MediaPackage to store the live-to-VOD asset. The Amazon S3 bucket name must be in the same region MediaPackage is harvesting from.
3. For **Manifest key**, enter the path within the bucket to the live-to-VOD asset, including the file name for the parent manifest of the asset. If the directory structure doesn't already exist in the bucket, MediaPackage creates it.

> **Important**
>
> The manifest key must be unique. When you use the same manifest key for multiple harvest jobs, the newest playlist for the asset overwrites existing playlists. The only time you should reuse a manifest key is when you are harvesting the same content, such as if there was a problem with a previous harvest of the content.

Viewing harvest job details

View all harvest jobs that you created within the last 90 days. After 90 days, a harvest job expires from your account.

You can use the MediaPackage console, the AWS CLI, or the MediaPackage API to view a harvest job. For information about viewing a job through the AWS CLI or MediaPackage API, see the AWS Elemental MediaPackage API Reference.

To view harvest job details (console)

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.

2. In the navigation pane, under **Live**, choose **Harvest jobs**.
3. On the **Harvest jobs** page, choose the harvest job to view its details.

Editing a harvest job

You can't edit a harvest job. To create a harvest job with different settings, see Creating a harvest job.

Deleting a harvest job

You can't delete a harvest job.

- To create a harvest job with different settings, see Creating a harvest job.
- To delete a VOD asset that MediaPackage created with a harvest job, see Delete an Object and a Bucket in the *Amazon Simple Storage Service Getting Started Guide.*

AWS Elemental MediaPackage features

The following sections describe the features that are available in AWS Elemental MediaPackage and how they work.

Topics

CDN authorization in AWS Elemental MediaPackage

Content Delivery Network (CDN) authorization helps you to protect your content from unauthorized use. When you configure CDN authorization, MediaPackage only fulfills playback requests that are authorized between MediaPackage and your CDN. This prevents users from bypassing the CDN in order to directly access your content on the origin.

How it works

You configure your CDN, such as Amazon CloudFront, to include a *custom HTTP header* in content requests to MediaPackage.

Custom HTTP header and example value.

```
X-MediaPackage-CDNIdentifier: 9ceebbe7-9607-4552-8764-876e47032660
```

You store the header value as a *secret* in AWS Secrets Manager. When your CDN sends a playback request, MediaPackage verifies that the secret's value matches the custom HTTP header value.

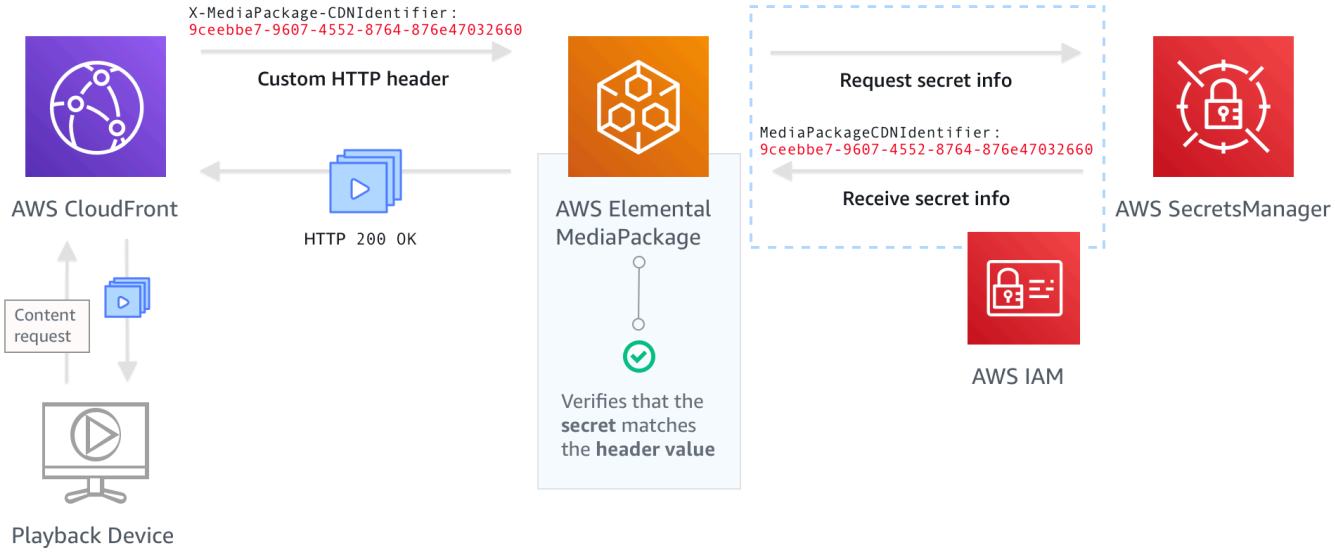
MediaPackage is given permission to read the secret with an AWS Identity and Access Management permissions policy and role.

Secret key and example value.

```
{“MediaPackageCDNIdentifier”: "9ceebbe7-9607-4552-8764-876e47032660"}
```

If the values match, MediaPackage serves the content along with an HTTP `200 OK` status code. If it's not a match, or if the authorization request fails, then MediaPackage doesn't serve the content, and sends an HTTP `403 Unauthorized` status code.

The following image shows successful CDN authorization using Amazon CloudFront.



For step-by-step instructions on how to set up CDN authorization, see Setting up CDN authorization.

Setting up CDN authorization

Complete the following steps to set up CDN authorization.

Topics

- Step 1: Configure a CDN custom origin HTTP header

- Step 2: Store the value as a secret in AWS Secrets Manager
- Step 3: Create an IAM policy and role for MediaPackage access to Secrets Manager
- Step 4: Enable CDN authorization in MediaPackage

Step 1: Configure a CDN custom origin HTTP header

In your CDN, configure a custom origin HTTP header that contains the header **`X-MediaPackage-CDNIdentifier`** and a value. For the value, we recommend that you use the UUID version 4 format, which produces a 36-character string. If you aren't using the UUID version 4 format, the value must be 8-128 characters long.

If your CDN has authorization headers configured, MediaPackage returns an error 404 until CDN authorization is enabled on the endpoint.

> **Important**
>
> The value you choose should be a static value. There isn't native integration between your CDN and AWS Secrets Manager, so the value should be static both in your CDN and in AWS Secrets Manager. If you change this value after you configure your CDN and your secret, you have to manually rotate the value. For more information, see Rotating the CDN header value.

Example header and value

```
X-MediaPackage-CDNIdentifier: 9ceebbe7-9607-4552-8764-876e47032660
```

To create a custom header in Amazon CloudFront

1. Sign in to the AWS Management Console and open the CloudFront console at https://console.aws.amazon.com/cloudfront/v4/home.
2. Create or edit a distribution.
3. In **Origin Settings**, complete the fields. You will use this same value for your secret in Secrets Manager.
   - For **Header Name**, enter **`X-MediaPackage-CDNIdentifier`**.

- For **Value**, enter a value. We recommend that you use UUID version 4 format, which produces a 36-character string. If you aren't using the UUID version 4 format, the value must be 8-128 characters long.

4. Complete the rest of the fields and save the distribution.

For more information about custom headers in CloudFront, see Forwarding customer headers to your origin in the *Amazon CloudFront Developer Guide*.

Step 2: Store the value as a secret in AWS Secrets Manager

Store the same value that you use in your custom origin HTTP header as a *secret* in AWS Secrets Manager. The secret must use the same AWS account and Region settings as your AWS Elemental MediaPackage resources. MediaPackage doesn't support sharing secrets across accounts or Regions. However, you can use the same secret across multiple endpoints in the same Region and on the same account.

To store a secret in Secrets Manager

1. Sign in to the AWS Secrets Manager console at https://console.aws.amazon.com/secretsmanager/.
2. Choose **Store a new secret**. For **Secret type**, choose **Other type of secrets**.
3. For **Key/value pairs**, enter the key and value information.
   - In the box on the left, enter **`MediaPackageCDNIdentifier`**.
   - In the box on the right, enter the value that you configured for your custom origin HTTP header. For example, `9ceebbe7-9607-4552-8764-876e47032660`.
4. For **Encryption key**, you can keep the default value as **DefaultEncryptionKey**.
5. Choose **Next**.
6. For **Secret name**, we recommend that you prefix it with **`MediaPackage/`** so that you know it's a secret used for MediaPackage. For example, **`MediaPackage/cdn_auth_us-west-2`**.
7. Choose **Next**.
8. For **Configure automatic rotation**, keep the default **Disable automatic rotation** setting.

   If you need to rotate the authorization code later, see Rotating the CDN header value.
9. Choose **Next**, and then choose **Store**.

This takes you to the list of your secrets.

10. Select your secret name to view the **Secret ARN**. The ARN has a value similar to `arn:aws:secretsmanager:us-west-2:123456789012:secret:MediaPackage/cdn_auth_test-xxxxxx`. You use the Secret ARN when you configure CDN authorization for MediaPackage in Step 4: Enable CDN Authorization in MediaPackage.

Step 3: Create an IAM policy and role for MediaPackage access to Secrets Manager

Create an IAM policy and role to give MediaPackage read access to Secrets Manager. When MediaPackage receives a playback request from the CDN, it verifies that the stored secret value matches the value in the custom HTTP header. Follow the steps in the section called "Allowing AWS Elemental MediaPackage to access other AWS services" to set up the policy and role.

Step 4: Enable CDN authorization in MediaPackage

You can enable CDN authorization for your endpoints or video on demand (VOD) packaging groups with the MediaPackage console, AWS CLI, or MediaPackage API. You use the ARN for the IAM policy and role that you created in Step 3: Create an IAM policy and role for MediaPackage access to Secrets Manager.

> **Tip**
>
> Use the same secret across multiple endpoints in the same Region and on the same account. Reduce costs by creating a new secret only when necessary for your workflow.

If your CDN has authorization headers configured, MediaPackage returns an error 404 until CDN authorization is enabled on the endpoint.

To enable CDN authorization for live content with the console

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. If you don't already have a channel, create one. For help, see Creating a channel.
3. Create or edit an endpoint.
4. In **Access control settings**, select **Use CDN authorization**. Complete the fields:

- In **Secrets role ARN**, enter the ARN for the IAM role that you created in Step 3: Create an IAM policy and role for MediaPackage access to Secrets Manager.
- In **CDN identifier secret ARN**, enter the ARN for the secret in Secrets Manager that your CDN uses for authorization to access your endpoint.

5. Complete the remaining fields as needed and save the endpoint.

To enable CDN authorization for VOD content with the console

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. If you don't already have a VOD packaging group, create one. For help, see Creating a packaging group.
3. Create or edit a packaging group.
4. In **Configure access control**, select **Enable authorization**. Complete the fields:
   - In **Secrets role ARN**, enter the ARN for the IAM role that you created in Step 3: Create an IAM policy and role for MediaPackage access to Secrets Manager.
   - In **CDN identifier secret ARN**, enter the ARN for the secret in Secrets Manager that your CDN uses for authorization to access your endpoint.
5. Complete the remaining fields as needed and save the packaging group.

You have now completed the setup for CDN authorization. Requests to this endpoint must contain the same authorization code that you saved in Secrets Manager.

To enable CDN authorization with the MediaPackage API

For information about enabling CDN authorization with the MediaPackage API, see the following API references:

- MediaPackage live API reference
- MediaPackage VOD API reference

Rotating the CDN header value

If you change the CDN custom origin HTTP header value, you need to rotate the stored secret value in Secrets Manager. The following procedure describes how to rotate your value in Secrets Manager

to make sure that your CDN's HTTP header value and the Secrets Manager stored secret value are in sync.

To rotate the value

1. Update the stored secret value in Secrets Manager as described in Modifying a secret in the *AWS Secrets Manager User Guide*.

   To ensure continued playback for active streams, MediaPackage authorizes requests that use either the current value in Secrets Manager or one version back.
2. Wait 10 minutes for MediaPackage to recognize that the value has changed in Secrets Manager.
3. In your CDN, update the value in `X-MediaPackage-CDNIdentifier` to the new authorization code.
4. Wait for your CDN to update fully with the new value before you send any requests through it to MediaPackage.

   To disable the previous secret value, save the new secret value two times. This way, both the current and previous secret versions have the same value.

Content encryption and DRM in AWS Elemental MediaPackage

Protect your content from unauthorized use through content encryption and digital rights management (DRM). AWS Elemental MediaPackage uses the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate content encryption and decryption by a DRM provider. Using SPEKE, the DRM provider supplies encryption keys to MediaPackage through the SPEKE API. The DRM provider also supplies licenses to supported media players for decryption. For more information about how SPEKE is used with services and features running in the cloud, see AWS cloud-based architecture in the *Secure Packager and Encoder Key Exchange API Specification guide*.

Limitations and requirements

When implementing content encryption for AWS Elemental MediaPackage, refer to the following limitations and requirements:

- Use the AWS Secure Packager and Encoder Key Exchange (SPEKE) API to facilitate integration with a digital rights management (DRM) provider. For information about SPEKE, see What is Secure Packager and Encoder Key Exchange?

- Your DRM provider must support SPEKE. For a list of DRM providers that support SPEKE, see the Get on board with a DRM platform provider topic in the *MediaPackage User Guide*. Your DRM solution provider can help you set up DRM encryption use in MediaPackage.
- Use MediaPackage to encrypt live and video on demand (VOD) content. Assets that must be delivered through the MediaPackage VOD service must be harvested from an unencrypted HLS live endpoint. You can harvest live-to-VOD assets from HLS and DASH endpoints that are protected by DRM or encryption. However, the MediaPackage VOD service can't ingest these assets because they're encrypted (not clear) content. For more information about this kind of workflow, see Creating live-to-VOD assets with AWS Elemental MediaPackage.

The following sections provide guidance on how to choose and implement content encryption using SPEKE for MediaPackage.

Topics

Choosing the right SPEKE Version

SPEKE Version 1 supports the use of a single encryption key for all audio and video tracks, and uses CPIX Version 2.0. For audio and video tracks, SPEKE Version 2.0 supports the use of multiple, distinct encryption keys and uses CPIX Version 2.3. For more information about SPEKE Version 2.0 encryption configurations, see SPEKE Version 2.0 presets.

If multiple key encryption, or Content Protection Information Exchange (CPIX) Version 2.3, are mandatory requirements for your content delivery, then SPEKE Version 2.0 is a good choice. However, SPEKE Version 2.0 support is progressive across endpoint types in MediaPackage. This means that some live options, like key rotation, aren't available yet. Take these constraints in consideration when crafting your SPEKE integration strategy. To learn more about the SPEKE Version 2.0 roadmap for MediaPackage, contact your AWS account team.

Supported protocols and DRM platforms

The following tables list the different protocols and digital rights management (DRM) platforms that SPEKE Version 1.0 and SPEKE Version 2.0 support.

> **Note**
>
> Irdeto Content Protection is not supported in combination with SPEKE Version 1.0.

SPEKE Version 1.0 – Support matrix for protocol and DRM system	**Microsoft PlayReady**	**Google Widevine**	**Apple FairPlay**	**AES-128**
Live				
Apple HLS	Not supported	Not supported	√ Has key rotation	√ Has key rotation
CMAF Apple HLS	Not supported	√ Has key rotation Supports only cbcs encryption	√ Has key rotation Supports only cbcs encryption	Not supported
DASH	√ Has key rotation	√ Has key rotation	Not supported	Not supported
Microsoft Smooth	√	Not supported	Not supported	Not supported
VOD				
Apple HLS	Not supported	Not supported	√	√
CMAF Apple HLS	Not supported	√	√	Not supported

		Supports only cbcs encryption	Supports only cbcs encryption	
DASH	√	√	Not supported	Not supported
Microsoft Smooth	√	Not supported	Not supported	Not supported

SPEKE Version 2.0 – Support matrix for protocol and DRM system	**Microsoft PlayReady**	**Google Widevine**	**Apple FairPlay**	**Irdeto Content Protection**
Live				
CMAF Apple HLS	√ Supports cbcs and cenc encryption	√ Supports cbcs and cenc encryption	√ Supports cbcs encryption	Not supported
DASH	√	√	Not supported	√
VOD				
CMAF Apple HLS	√ Supports only cbcs encryption	√ Supports only cbcs encryption	√ Supports only cbcs encryption	Not supported
DASH	√	√	Not supported	√

Deploying SPEKE

Your digital rights management (DRM) solution provider can help you get set up to use DRM encryption in MediaPackage. Generally, the provider gives you a SPEKE gateway to deploy in your AWS account in the same AWS Region where MediaPackage is running. Along with configuring your

origin endpoints with the right encryption settings, you must configure event notifications for the key provider events that MediaPackage is generating as CloudWatch Events. For information about configuring encryption settings for your endpoint, see the applicable section for your protocol: HLS encryption fields, MSS encryption fields, CMAF encryption fields, and DASH encryption fields.

If you must build your own API Gateway to connect MediaPackage to your key service, you can use the SPEKE Reference Server available on GitHub as a starting point.

Preparing and managing certificates for use with content keys

AWS Elemental MediaPackage uses a Content Protection Information Exchange (CPIX) document to communicate with SPEKE about content keys that are used to encrypt your content. For the most secure digital rights management (DRM) encryption solution, use encrypted content keys in the CPIX document.

To use encrypted content keys, the following requirements must be met:

- The encrypted content must be live. Video on demand (VOD) and live-to-VOD workflows don't support encrypted content keys in the CPIX document.
- Your DRM key provider must support encrypted content keys. If you enable this feature for a key provider that doesn't handle content key encryption, playback fails.
- You must import a suitable certificate into AWS Certificate Manager (ACM) in the same Region that you run MediaPackage. For information about ACM, see the AWS Certificate Manager User Guide.

  The following procedures describe how to prepare and manage the certificate.

To prepare a certificate for DRM content key encryption

1. Obtain a 2048 RSA, SHA-512-signed certificate.
2. Open the ACM console at https://console.aws.amazon.com/acm/.
3. Import the certificate into ACM according to the instructions at Importing certificates into AWS certificate manager. Note the resulting certificate ARN because you will need it later.

   For use in DRM encryption, your certificate must have a status of **Issued** in ACM.

To use a certificate in AWS Elemental MediaPackage

When you use DRM encryption in your endpoint configuration, provide your certificate ARN in the encryption parameters. This enables content key encryption. You can use the same certificate ARN for multiple events. For information, see the encryption settings information in the section called "Working with endpoints".

To renew a certificate

To renew a certificate that you are using in AWS Elemental MediaPackage, reimport it in ACM. The certificate renews without any disruption of its use in MediaPackage.

To delete a certificate

To delete a certificate from ACM, it must not be associated with any other service. Delete the certificate ARN from endpoint configurations where you have used it, then delete it from ACM.

> **Note**
>
> If you delete a certificate ARN from an active endpoint, the endpoint keeps running, but stops using content key encryption.

Understanding key rotation behavior

When you enable key rotation on live content from HLS, CMAF, and DASH endpoints, AWS Elemental MediaPackage retrieves content keys before the live content begins. As the content progresses, MediaPackage retrieves new keys at the interval that you set on the endpoint, as described in Package encryption fields.

If MediaPackage is unable to retrieve the content key, it takes the following actions:

- If MediaPackage successfully retrieved a content key for this endpoint before, it uses the last key that it fetched. This ensures that endpoints that worked previously continue to work.
- If MediaPackage has *not* successfully retrieved a content key for this endpoint before, MediaPackage responds to the playback request with error 404.

In all cases, when MediaPackage can't fetch a content key, it generates a CloudWatch event, as described in Key provider notification events.

SPEKE Version 2.0 presets

SPEKE Version 2.0 supports the use of multiple, distinct encryption keys for audio and video tracks. MediaPackage uses **presets** to configure the encryption. The MediaPackage API defines these presets, and they appear in the MediaPackage console in the **Video encryption preset** and **Audio encryption preset** menus of the **Package Encryption endpoints configuration** section. The presets map encryption keys to specific audio or video tracks, based on the number of channels for audio tracks, and based on the video resolution for video tracks. MediaPackage uses specific combinations of audio and video encryption presets to support three different encryption scenarios:

- Scenario 1: Unencrypted tracks and encrypted tracks
- Scenario 2: Single encryption key for all audio and video tracks
- Scenario 3: Multiple encryption keys for audio and video tracks

Scenario 1: Unencrypted tracks and encrypted tracks

You can choose *not* to encrypt the audio or the video tracks by selecting the **UNENCRYPTED** preset in the **Video encryption preset** or the **Audio encryption preset** menus. You can't select **UNENCRYPTED** for both audio and video presets, because doing so would mean that you don't intend to encrypt any of the tracks at all. Also, you can't combine **UNENCRYPTED** and **SHARED** presets for audio and video, because **SHARED** is a special preset. For more information, see Scenario 2: Single encryption key for all audio and video tracks.

The following list describes valid combinations of **UNENCRYPTED** presets:

- **UNENCRYPTED** for audio tracks, and any video preset with a name that starts with `PRESET-VIDEO-`
- **UNENCRYPTED** for video tracks, and any audio preset with a name that starts with `PRESET-AUDIO-`

Scenario 2: Single encryption key for all audio and video tracks

The SPEKE Version 2.0 **SHARED** preset uses a single encryption key for all audio and video tracks, as in SPEKE Version 1.0. When you select the **SHARED** preset, select it for both audio and video encryption.

Scenario 3: Multiple encryption keys for audio and video tracks

When you use a preset with a name that starts with `PRESET-VIDEO-` or `PRESET-AUDIO-`, MediaPackage encrypts the audio tracks and video tracks with the number of encryption keys that the specific preset defines. The following tables show how many keys MediaPackage requests from the key server and how those keys map to tracks. If no track matches the criteria for a particular key, MediaPackage does not use that key to encrypt any track.

MediaPackage encrypts I-frame only trickplay tracks with the key corresponding to their resolution.

In the following table, the **Key name** value is the value of the `ContentKeyUsageRule@IntendedTrackType` attribute that MediaPackage uses in the CPIX document. This is sent to the SPEKE server for a specific content key.

Video encryption presets

Preset name	Number of keys	Key name	Minimum resolution	Maximum resolution
PRESET-VIDEO-1	1	VIDEO	No minimum or maximum resolution. MediaPackage encrypts all tracks with the same key.	
PRESET-VIDEO-2	2	SD	No minimum	<= 1024x576
		HD	> 1024x576	No maximum
PRESET-VIDEO-3	3	SD	No minimum	<= 1024x576
		HD	> 1024x576	<= 1920x1080
		UHD	> 1920x1080	No maximum
PRESET-VIDEO-4	4	SD	No minimum	<= 1024x576
		HD	> 1024x576	<= 1920x1080
		UHD1	> 1920x1080	<= 4096x2160
		UHD2	> 4096x2160	No maximum

Preset name	Number of keys	Key name	Minimum resolution	Maximum resolution
PRESET-VIDEO-5	5	SD	No minimum	<= 1024x576
		HD1	> 1024x576	<= 1280x720
		HD2	> 1280x720	<= 1920x1080
		UHD1	> 1920x1080	<= 4096x2160
		UHD2	> 4096x2160	No maximum
PRESET-VIDEO-6	4	SD	No minimum	<= 1024x576
		HD1	> 1024x576	<= 1280x720
		HD2	> 1280x720	<= 1920x1080
		UHD	> 1920x1080	No maximum
PRESET-VIDEO-7	3	SD+HD1	No minimum	<= 1280x720
		HD2	> 1280x720	<= 1920x1080
		UHD	> 1920x1080	No maximum
PRESET-VIDEO-8	4	SD+HD1	No minimum	<= 1280x720
		HD2	> 1280x720	<= 1920x1080
		UHD1	> 1920x1080	<= 4096x2160
		UHD2	> 4096x2160	No maximum
SHARED	1	ALL	No minimum or maximum resolution. MediaPackage encrypts all video and audio tracks with the same key.	
UNENCRYPTED	0	*N/A*	MediaPackage does not encrypt any video track.	

In the following table, the **Key name** value is the value of the `ContentKeyUsageRule@IntendedTrackType` attribute that MediaPackage uses in the CPIX document. This is sent to the SPEKE server for a specific content key.

Audio encryption presets

Preset name	Number of keys	Key name	Minimum number of channels	Maximum number of channels
PRESET-AUDIO-1	1	AUDIO	No minimum or maximum number of channels. MediaPackage encrypts all audio and video tracks with the same key.	
PRESET-AUDIO-2	2	STEREO_AUDIO	No minimum	2
		MULTICHANNEL_AUDIO	> 2	No maximum
PRESET-AUDIO-3	3	STEREO_AUDIO	No minimum	2
		MULTICHANNEL_AUDIO_3_6	> 2	<= 6
		MULTICHANNEL_AUDIO_7	> 6	No maximum
SHARED	1	ALL	No minimum or maximum number of channels. MediaPackage encrypts all audio and video tracks with the same key.	
UNENCRYPTED	0	*N/A*	MediaPackage does not encrypt any audio track.	

Now you know how MediaPackage supports SPEKE Version 2.0 presets for unencrypted tracks and encrypted tracks. With these presets, you can use a single encryption key for all audio and video tracks, and multiple encryption keys for audio and video tracks.

Removing tags from the parent manifest from AWS Elemental MediaPackage

MediaPackage signals in the parent manifest the `#EXT-X-SESSION-KEY` tag for every track type on an HLS or CMAF endpoint. This tag enables playback devices to pre-fetch keys when a key is shared across multiple streams. There are times when you might not want this optional tag, such as when you're using only a subset of the tracks and don't want all of the keys referenced in the parent manifest. With SPEKE v2, you can append a query parameter to your manifest requests that will remove all `#EXT-X-SESSION-KEY` tags from the parent manifest. Because each child manifest has its own `#EXT-X-KEY` tag for obtaining a decryption key, the `#EXT-X-SESSION-KEY` is often superfluous.

To remove the `#EXT-X-SESSION-KEY` tag from MediaPackage manifest responses, use the following query parameter: `aws.drmsettings=excludesessionkeys`

The following section provides more information about using query parameters.

Query syntax

The base query parameter for removing `#EXT-X-SESSION-KEY` tags is `aws.drmsettings`, which is followed by optional parameter name and value pairs. To construct the query, append `?aws.drmsettings=` to the end of the MediaPackage endpoint URL, followed by the parameter name and value.

An Apple HLS filter query might look like this:

`https://example-mediapackage-endpoint.mediapackage.us-west-2.amazonaws.com/out/v1/examplemediapackage/index.m3u8?aws.drmsettings=excludesessionkeys`

The query syntax is listed in the following table.

> **Note**
>
> If you use Amazon CloudFront as your CDN, you might need to set additional configurations. For more information, see Configure cache behavior for all endpoints.

Query string component	Description
`?`	A restricted character that marks the beginning of a query.
`aws.drmsettings=`	The base query, which is followed by parameters constructed of name and value pairs.
`:`	Used to associate the parameter name with a value. For example, *`parameter_name :value`*.
`;`	Separates parameters in a query that contains multiple parameter s. For example, *`parameter1_name:value ;parameter 2_name:minValue-maxValue`* . When used in a list of parameters for the same query, implies an `AND` operation.

Error conditions

Some playback devices will return errors if the manifest or segments include invalid or unknown query parameters. The following are query parameters that MediaPackage can process:

- `m`
- `start`
- `end`
- `aws.manifestfilter`
- `aws.drmsettings`

If you have query parameters other than those listed, use a CDN such as Amazon CloudFront to remove the unnecessary parameters. For more information, see Cache content based on query string parameters in the *Amazon CloudFront Developer Guide*.

The following table contains additional common error conditions.

Error condition	Example	HTTP status code
A list parameter is not found and is not part of a constrained list	`?aws.manifestfilter=audio_language:dahlia`	200
Only subtitle streams are present in the stream	`?aws.manifestfilter=audio_sample_rate:0-1;video_bitrate=0-1`	200
Duplicate filter parameter	`?aws.manifestfilter=audio_sample_rate:0-48000;aws.manifestfilter=audio_sample_rate:0-48000`	400
Invalid parameter	`?aws.manifestfilter=donut_type:rhododendron`	400
Invalid range parameter	`?aws.manifestfilter=audio_sample_rate:300-0`	400
Invalid range value (more than `INT_MAX`)	`?aws.manifestfilter=audio_sample_rate:0-2147483648`	400
Malformed query string	`?aws.manifestfilter=audio_sample_rate:is:0-44100`	400
Parameter string is greater than 1024 characters	`?aws.manifestfilter=audio_language:abcdef....`	400

Error condition	Example	HTTP status code
Query parameters on an HLS or CMAF bitrate manifest	`index_1.m3u8?aws.manifestfilter=video_codec:h264`	400
Query parameters on a segment request	`..._1.[ts|mp4|vtt.]?aws.manifestfilter=video_codec:h264`	400
Repeated query parameter	`?aws.manifestfilter=audio_sample_rate:0-48000;aws.manifestfilter=video_bitrate:0-1`	400
Application of the filter results in an empty manifest (content has no streams that meet the conditions defined in the query string)	`?aws.manifestfilter=audio_sample_rate:0-1;video_bitrate=0-1`	400

DASH manifest options in AWS Elemental MediaPackage

This section describes the options that AWS Elemental MediaPackage offers for modifying live output DASH manifests.

These options don't apply to video on demand (VOD) outputs or harvested live-to-VOD assets.

Default DASH manifest

The following is a truncated example of a DASH manifest with no treatments:

```
<MPD>
  <Period>
    <AdaptationSet>
      <Representation>
        <SegmentTemplate>
```

```
          <SegmentTimeline>
            <S />
          </SegmentTimeline>
        </SegmentTemplate>
      </Representation>
    </AdaptationSet>
    .
    .
  </Period>
</MPD>
```

The elements of the DASH manifest are nested within the `MPD` (media presentation description) object. These are the elements of the manifest:

- `Period` - The entire manifest is nested in one period.
- `AdaptationSet` - Groups together representations of the same type (video, audio, or captions). There are one or more `AdaptationSets` in the `Period`.
- `Representation` - Describes an audio, video, or captions track. There are one or more `Representations` in each `AdaptationSet`. Each representation is a track.
- `SegmentTemplate` - Defines properties of the representation, such as the timescale and access URLs for media and initialization segments. There is one `SegmentTemplate` for each `Representation`.
- `SegmentTimeline` - Describes when each segment is available for playback. There is one `SegmentTimeline` for each `SegmentTemplate`.
- `S` - Describes when the segment is available (`t` value), the duration of the segment (`d` value), and a count of how many additional consecutive segments have this same duration (`r` value). There are one or more segments in the `SegmentTimeline`.

MediaPackage can modify how some of these elements are presented in the output manifest. You can use the following treatment options on the output live manifest:

- Separate the manifest into multiple periods, to allow ad breaks. See DASH manifest options in AWS Elemental MediaPackage.
- Reduce the length of the manifest to make processing and playback more efficient. See Compacted DASH manifests.
- Control what segment information is used in the media URL in the `SegmentTemplate` properties. See DASH manifest segment template format.

Multi-period DASH in AWS Elemental MediaPackage

The ability to insert multiple periods in DASH manifests for both VOD and live is available in AWS Elemental MediaPackage.

A period is a chunk of content in the DASH manifest, defined by a start time and duration. By default, the entire manifest is contained in one period but MediaPackage can partition the DASH manifest into multiple periods to indicate boundaries between ads and the main content. For example, if you're using MediaPackage with a downstream ad service such as AWS Elemental MediaTailor, choose **Trigger new period on ads** on the MPEG-DASH endpoint in MediaPackage. This option tells MediaPackage that the DASH manifest is to be formatted with multiple periods.

- For information about AWS Elemental MediaTailor, see the *AWS Elemental MediaTailor User Guide*.
- For information about DASH-ISO endpoints in MediaPackage, see Creating a DASH endpoint.
- For more information about how multi-period DASH works in MediaPackage, see the following *How it Works* section.

How multi-period DASH works

To use the multi-period DASH feature, the input to MediaPackage must have SCTE-35 ad marker messages. These messages inform MediaPackage of where to create period boundaries. This is how MediaPackage processes those messages:

1. MediaPackage detects the SCTE-35 messages from the input source.
2. Using the attributes of the SCTE-35 messages, MediaPackage calculates where the boundaries are between the end of the main content and the ads. This calculation is (scte35 `ptsAdjustment` + scte35 `ptsTime`) / (EventStream `timescale`).

   Example

   In the following example, the period starts at 44.075 seconds because (183003 + 3783780) / 90000 = 44.075:

```
<Period start="PT44.075S" id="21">
    <EventStream timescale="90000" schemeIdUri="urn:scte:scte35:2013:xml">
      <Event>
        <scte35:SpliceInfoSection protocolVersion="0" ptsAdjustment="183003"
 tier="4095">
```

```
          <scte35:SpliceInsert spliceEventId="1000"
 spliceEventCancelIndicator="false" outOfNetworkIndicator="true"
 spliceImmediateFlag="false" uniqueProgramId="7" availNum="1" availsExpected="4">
            <scte35:Program><scte35:SpliceTime ptsTime="3783780"/></scte35:Program>
          </scte35:SpliceInsert>
        </scte35:SpliceInfoSection>
      </Event>
    </EventStream>
     .
     .
</Period>
```

3. MediaPackage inserts the `EventStream`, `Event`, and `scte35` tags with additional information into the manifest and surrounds the ad period with a `Period` tag, as shown in the preceding example. MediaPackage groups all adaptation sets before the first ad period into a period, and any subsequent adaptation sets after the ad are grouped into a period, until the next SCTE-35 marker. Here is a complete manifest example with multiple periods. It uses `SpliceInsert` SCTE-35 ad markers:

   Example

```
<?xml version="1.0" encoding="utf-8"?>
<MPD>
  <Period start="PT0.000S" id="0" duration="PT44.075S">
    <AdaptationSet mimeType="video/mp4" segmentAlignment="true"
 subsegmentAlignment="true" startWithSAP="1" subsegmentStartsWithSAP="1"
 bitstreamSwitching="true">
      <Representation id="1" width="960" height="540" frameRate="30000/1001"
 bandwidth="1000000" codecs="avc1.4D401F">
        <SegmentTemplate timescale="30000" media="index_video_1_0_$Number$.mp4?
m=1528413503" initialization="index_video_1_0_init.mp4?m=1528413503" startNumber="6"
 presentationTimeOffset="0">
          <SegmentTimeline>
            <S t="361301" d="60060" r="15"/>
          </SegmentTimeline>
        </SegmentTemplate>
      </Representation>
    </AdaptationSet>
    <AdaptationSet mimeType="audio/mp4" segmentAlignment="0" lang="eng">
      <Representation id="2" bandwidth="96964" audioSamplingRate="48000"
 codecs="mp4a.40.2">
```

```
        <SegmentTemplate timescale="48000" media="index_audio_2_0_$Number$.mp4?
m=1528413503" initialization="index_audio_2_0_init.mp4?m=1528413503" startNumber="6"
 presentationTimeOffset="0">
          <SegmentTimeline>
            <S t="578305" d="96256" r="3"/>
            <S t="963329" d="95232"/>
            <S t="1058561" d="96256" r="5"/>
            <S t="1636097" d="95232"/>
            <S t="1731329" d="96256" r="3"/>
          </SegmentTimeline>
        </SegmentTemplate>
      </Representation>
    </AdaptationSet>
  </Period>
  <Period start="PT44.075S" id="21">
    <EventStream timescale="90000" schemeIdUri="urn:scte:scte35:2013:xml">
      <Event>
        <scte35:SpliceInfoSection protocolVersion="0" ptsAdjustment="183003"
 tier="4095">
          <scte35:SpliceInsert spliceEventId="1000"
 spliceEventCancelIndicator="false" outOfNetworkIndicator="true"
 spliceImmediateFlag="false" uniqueProgramId="7" availNum="1" availsExpected="4">
            <scte35:Program><scte35:SpliceTime ptsTime="3783780"/></scte35:Program>
          </scte35:SpliceInsert>
        </scte35:SpliceInfoSection>
      </Event>
    </EventStream>
    <AdaptationSet mimeType="video/mp4" segmentAlignment="true"
 subsegmentAlignment="true" startWithSAP="1" subsegmentStartsWithSAP="1"
 bitstreamSwitching="true">
      <Representation id="1" width="960" height="540" frameRate="30000/1001"
 bandwidth="1000000" codecs="avc1.4D401F">
        <SegmentTemplate timescale="30000" media="index_video_1_0_$Number$.mp4?
m=1528413503" initialization="index_video_1_0_init.mp4?m=1528413503" startNumber="22"
 presentationTimeOffset="1322261">
          <SegmentTimeline>
            <S t="1322261" d="60060" r="13"/>
          </SegmentTimeline>
        </SegmentTemplate>
      </Representation>
    </AdaptationSet>
    <AdaptationSet mimeType="audio/mp4" segmentAlignment="0" lang="eng">
      <Representation id="2" bandwidth="96964" audioSamplingRate="48000"
 codecs="mp4a.40.2">
```

```
        <SegmentTemplate timescale="48000" media="index_audio_2_0_$Number$.mp4?
m=1528413503" initialization="index_audio_2_0_init.mp4?m=1528413503" startNumber="22"
 presentationTimeOffset="2115617">
          <SegmentTimeline>
            <S t="2116353" d="96256"/>
            <S t="2212609" d="95232"/>
            <S t="2307841" d="96256" r="5"/>
            <S t="2885377" d="95232"/>
            <S t="2980609" d="96256" r="4"/>
          </SegmentTimeline>
        </SegmentTemplate>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

If your input has `TimeSignal` SCTE-35 ad markers instead of `SpliceInsert`, the EventStream within the ad period looks like this:

```
<EventStream timescale="90000" schemeIdUri="urn:scte:scte35:2013:xml">
  <Event>
    <scte35:SpliceInfoSection protocolVersion="0" ptsAdjustment="183265" tier="4095">
      <scte35:TimeSignal>
        <scte35:SpliceTime ptsTime="1350000"/>
      </scte35:TimeSignal>
      <scte35:SegmentationDescriptor segmentationEventId="1073741825"
 segmentationEventCancelIndicator="false" segmentationDuration="450000">
        <scte35:DeliveryRestrictions webDeliveryAllowedFlag="false"
 noRegionalBlackoutFlag="true" archiveAllowedFlag="true" deviceRestrictions="3"/>
        <scte35:SegmentationUpid segmentationUpidType="1" segmentationUpidLength="3"
 segmentationTypeId="48" segmentNum="0" segmentsExpected="0">012345</
scte35:SegmentationUpid>
      </scte35:SegmentationDescriptor>
    </scte35:SpliceInfoSection>
  </Event>
</EventStream>
```

MediaPackage also embeds `scte35:SpliceInsert` messages as metadata in the individual video segments.

If you're using a downstream ad service, that service looks for the SCTE-35 markers in the manifest that MediaPackage provides and inserts ads based on those markers.

Compacted DASH manifests

The ability to compact DASH manifests to improve performance and processing on low-power devices for both VOD and live is available in AWS Elemental MediaPackage.

The default DASH manifest from MediaPackage includes duplicate data about each representation (track). For some players, processing a manifest with all this data is difficult and slow. To reduce some of the burden, MediaPackage can compact the manifest by moving some attributes from the `Representation` object to the `AdaptationSet` object. This way, rather than having the attributes defined for each representation in the manifest, they're defined once at a higher level. The representations then inherit these attributes from the adaptation set.

Example Default DASH manifest

In the following example, the `SegmentTemplate` object and all of its elements are listed in every `Representation.` Each adaptation set in the manifest has this same layout:

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true" subsegmentAlignment="true"
 startWithSAP="1" subsegmentStartsWithSAP="1" bitstreamSwitching="true">
   <Representation id="1" width="640" height="360" frameRate="30/1" bandwidth="749952"
 codecs="avc1.640029">
      <SegmentTemplate timescale="30000" media="index_video_1_0_$Number$.mp4?
m=1543947824" initialization="index_video_1_0_init.mp4?m=1543947824" startNumber="1">
         <SegmentTimeline>
           <S t="62000" d="60000" r="9"/>
         </SegmentTimeline>
      </SegmentTemplate>
   </Representation>
   <Representation id="2" width="854" height="480" frameRate="30/1" bandwidth="1000000"
 codecs="avc1.640029">
      <SegmentTemplate timescale="30000" media="index_video_3_0_$Number$.mp4?
m=1543947824" initialization="index_video_3_0_init.mp4?m=1543947824" startNumber="1">
         <SegmentTimeline>
           <S t="62000" d="60000" r="9"/>
         </SegmentTimeline>
      </SegmentTemplate>
   </Representation>
   <Representation id="3" width="1280" height="720" frameRate="30/1"
 bandwidth="2499968" codecs="avc1.640029">
```

```
      <SegmentTemplate timescale="30000" media="index_video_5_0_$Number$.mp4?
m=1543947824" initialization="index_video_5_0_init.mp4?m=1543947824" startNumber="1">
         <SegmentTimeline>
           <S t="62000" d="60000" r="9"/>
         </SegmentTimeline>
      </SegmentTemplate>
   </Representation>
</AdaptationSet>
```

Example Compacted DASH manifest

In this example, the `SegmentTemplate` objects and all of their elements are collapsed into one and moved to the `AdaptationSet`. The playback device understands that each representation in this adaptation set uses this same template:

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true" subsegmentAlignment="true"
 startWithSAP="1" subsegmentStartsWithSAP="1" bitstreamSwitching="true">
   <SegmentTemplate timescale="30000" media="index_video_$RepresentationID$_0_
$Number$.mp4?m=1543947824" initialization="index_video_$RepresentationID$_0_init.mp4?
m=1543947824" startNumber="1">
     <SegmentTimeline>
       <S t="62000" d="60000" r="9"/>
     </SegmentTimeline>
   </SegmentTemplate>
   <Representation id="1" width="640" height="360" frameRate="30/1" bandwidth="749952"
 codecs="avc1.640029"/>
   <Representation id="2" width="854" height="480" frameRate="30/1" bandwidth="1000000"
 codecs="avc1.640029"/>
   <Representation id="3" width="1280" height="720" frameRate="30/1"
 bandwidth="2499968" codecs="avc1.640029"/>
</AdaptationSet>
```

For information about compacting a DASH manifest, see [How AWS Elemental MediaPackage Compacts Manifests].

How AWS Elemental MediaPackage compacts manifests

To compact the DASH manifest from the AWS Elemental MediaPackage console, choose **Compact** for **Manifest layout** on the DASH endpoint. To ensure that tracks are available at the right time, MediaPackage checks the frame rate and audio sampling rate in the source content to determine if the manifest can be compacted.

> **Note**
>
> Captions tracks always use the same rate, so MediaPackage always compacts adaptation sets with captions.

MediaPackage takes the following actions:

- If the rates are the same across all representations in an adaptation set, MediaPackage collapses all of the `SegmentTemplate` objects into one and moves it to the `AdaptationSet` level. This way, the information in the template isn't repeated throughout the manifest. To allow the playback device to use the same template information across representations, MediaPackage adds a `$RepresentationID$` variable to the `media` and `initialization` request URLs. The playback device replaces this variable with the ID of the representation that it's currently requesting. MediaPackage also moves the `ContentProtection` element, when it's present, to the adaptation set as well.
- If the rates are different across representations, MediaPackage compacts and moves the `SegmentTemplate` with the most frequent rate to the `AdaptationSet`. Representations with a different rate keep their segment template. The rate for the representation overrides the one at the adaptation set.
- If there are exactly two frame rates in use in a video adaptation set, MediaPackage compacts as follows:
  - When 24 and 48 are used, the compacted template uses 48 for the frame rate and 48000 for the timebase.
  - When 25 and 50 are used, the compacted template uses 50 for the frame rate and 50000 for the timebase.
  - When 29.97 and 59.94 are used, the compacted template uses 59.95 for the frame rate and 60000 for the timebase.
  - When 30 and 60 are used, the compacted template uses 60 for the frame rate and 60000 for the timebase.

  If there are two video frame rates in use but they aren't in one of the doubled patterns above, then that set can't be compacted.
- If there are no duplicate rates across representations in an adaptation set, then that set can't be compacted.

DASH manifest segment template format

The ability to select the format of the DASH segment template is available with only live workflows in AWS Elemental MediaPackage.

The following sections describe how you can modify the `SegmentTemplate` object in DASH manifests to better fit your playback device's requirements.

Topics

`media` Attribute in `SegmentTemplate`

The `media` attribute in the `SegmentTemplate` properties defines the URL where playback devices send segment requests. By default, this URL uses a `$Number$` variable to identify the specific segment that's requested. When a playback device requests the segment, it replaces the variable with the number identifier of the segment. For the first segment in the representation, replace this identifier with the value of the `startNumber` from the `SegmentTemplate` properties. Each additional segment increments by one.

Some players navigate the segments better when the segments are identified instead by the timestamp for when playback is available. To support this use case, MediaPackage uses the `$Time$` variable instead of `$Number$` in the URL of the `media` attribute. When a playback device requests the segment, it replaces the variable with the availability start time of the segment. This start time is identified in the `t` value of the segment (S) properties in the `SegmentTimeline` object. For an example, see How It Works.

How the `$Time$` variable works

Enable the `$Time$` variable through the **Segment template format** setting on the DASH endpoint, as described in Creating a DASH endpoint. AWS Elemental MediaPackage takes the following actions:

1. When MediaPackage generates the DASH manifest, it uses the `$Time$` variable in the `media` value of the `SegmentTemplate` object, as shown in the following example:

Example

```
<SegmentTemplate timescale="30" media="index_video_1_0_$Time$.mp4?m=1122792372"
 initialization="index_video_1_0_init.mp4?m=1122792372" startNumber="2937928">
```

2. When a playback device requests segments, it uses the URL defined in the `media` attribute and replaces the variable with the availability start time of the segment that's requested.

> **Important**
>
> The value that replaces the variable must be an exact `t` value of a segment. If the request uses an arbitrary timestamp, MediaPackage doesn't seek the closest segment.

Example

The following is an example of a segment template from a representation. It uses the `$Time$` variable:

```
<SegmentTemplate timescale="30000" media="155_video_1_2_$Time$.mp4?m=1545421124"
 initialization="155_video_1_2_init.mp4?m=1545421124" startNumber="710">
  <SegmentTimeline>
       <S t="255197799" d="360360" r="8"/>
       <S t="258441039" d="334334"/>
  </SegmentTimeline>
</SegmentTemplate>
```

The request URL for the first segment is **155_video_1_2_*255197799*.mp4**. With a 360360 duration, the next segment request is **155_video_1_2_*255558159*.mp4**, and so on through the ninth segment.

The final segment request is **155_video_1_2_*258441039*.mp4**.

`duration` Attribute in the `SegmentTemplate`

In a default DASH manifest, `SegmentTemplate` holds a `SegmentTimeline`. The timeline describes all the segments in `Representation`, including their duration and their start time. With live events, AWS Elemental MediaPackage adds segments to the timeline as it receives them from

your encoder. To be aware of newly available segments, the playback device must regularly request an updated manifest from MediaPackage.

If all the segments in a representation have the same duration, you can help to reduce latency and shorten the manifest by enabling MediaPackage to remove the `SegmentTimeline` objects. In their place, MediaPackage adds a `duration` attribute to the `SegmentTemplate` properties. The playback device calculates when segments are available by using `duration` and `startNumber`. Because the playback device doesn't have to rely on an updated manifest to know about segments, it doesn't have to constantly request updates to maintain playback. For information about how the `duration` attribute works, see the following sections.

Topics

How the `duration` attribute works

Enable the `$duration$` attribute through the **Segment template format** setting on the DASH endpoint, as described in Creating a DASH endpoint. This is what happens with the manifest:

1. When AWS Elemental MediaPackage generates the DASH manifest, it adds the `duration` attribute to the `SegmentTemplate` object, as shown in the following example:

   Example

   ```
   <SegmentTemplate timescale="30000" media="index_video_1_0_$Number$.mp4?
   m=1535562908" initialization="index_video_1_0_init.mp4?m=1535562908"
    startNumber="175032" duration="90000" presentationTimeOffset="62061"/>
   ```

   A segment timeline and individual segment descriptions are not included in the segment template.

   > **Important**
   >
   > Except for the final segment, segments must be no more than 50% deviation from the value of the duration. With a 90000 duration, segments must be between 45000 and 135000 (1.5 to 4.5 seconds with a 30000 timescale).

Example

The following is an example of an adaptation set that uses the `duration` in the segment template:

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true"
 subsegmentAlignment="true" startWithSAP="1" subsegmentStartsWithSAP="1"
 bitstreamSwitching="true">
  <Representation id="1" width="852" height="480" frameRate="30/1"
 bandwidth="1200000" codecs="avc1.4D401F">
    <SegmentTemplate timescale="30000" media="index_video_1_0_$Number
$.mp4?m=1535562908" initialization="index_video_1_0_init.mp4?m=1535562908"
 startNumber="175032" duration="90000" presentationTimeOffset="62061"/>
  </Representation>
  <Representation id="2" width="640" height="360" frameRate="30/1" bandwidth="800000"
 codecs="avc1.4D401E">
    <SegmentTemplate timescale="30000" media="index_video_3_0_$Number
$.mp4?m=1535562908" initialization="index_video_3_0_init.mp4?m=1535562908"
 startNumber="175032" duration="90000" presentationTimeOffset="62061"/>
  </Representation>
  <Representation id="3" width="320" height="240" frameRate="30/1" bandwidth="499968"
 codecs="avc1.4D400D">
    <SegmentTemplate timescale="30000" media="index_video_5_0_$Number
$.mp4?m=1535562908" initialization="index_video_5_0_init.mp4?m=1535562908"
 startNumber="175032" duration="90000" presentationTimeOffset="62061"/>
  </Representation>
</AdaptationSet>
```

2. The playback device requests segments using the URL that's defined in the `media` attribute. In the URL, it replaces the `$Number$` variable with the number of the segment, starting with the value of the `startNumber` in the `SegmentTemplate` for the first segment.
3. If your playback device needs to determine the most recent segment, it uses this formula:

   ((wall clock time - `availabilityStartTime`) / (`duration` / `timescale`)) + `startNumber`

   Example

   A playback device is calculating the most recent segment with the following values:

   - Wall clock time from the playback device: 2018-11-16T19:18:30Z

- availabilityStartTime - Attribute from the MPD object of the manifest: 2018-11-16T19:08:30Z
- duration - Attribute from the SegmentTemplate object of the manifest: 90000
- timescale - Attribute from the SegmentTemplate: 30000
- startNumber - Attribute from the SegmentTemplate: 175032

The calculation it uses is ((2018-11-16T19:18:30Z - 2018-11-16T19:08:30Z) / (90000/30000)) + 175032

This calculation then becomes (600 seconds elapsed time) / (3 second segment durations) = 200 elapsed segments. Adding those segments to the 175032 start segment makes the most recent segment 175232.

duration Attribute limitations

To ensure proper playback and help prevent issues with conflicting segment durations, AWS Elemental MediaPackage enforces the following limitations for the duration attribute:

- You can enable the feature only when you create the endpoint.

  You can't modify the endpoint to later add the duration attribute to your DASH manifests. This includes changing from one segment template format to one that uses duration. For example, you can't create an endpoint that uses the $Time$ variable with SegmentTimeline, and then edit the endpoint to use the $Number$ variable with duration.
- You must keep the **segment duration** value that you set when you create the endpoint.

  You can't edit the endpoint to modify the segment duration.
- You must produce single period DASH manifests from endpoints that use duration.

  You can't use multi-period DASH with the duration attribute.
- Your ingest stream must use a regular segmentation cadence.
- You can't use variable segment length in the ingest stream. For example, resulting of a SCTE-35-related segmentation.

`duration` Attribute with compacted DASH manifests

Combining compacted manifests with the `duration` attribute will further reduce the size of the manifest, but not by much. Compacted manifests have one `SegmentTemplate` and `SegmentTimeline` per adaptation set. When you use the `duration` attribute, AWS Elemental MediaPackage removes the segment timeline. With both treatments, the manifest has one `SegmentTemplate` per adaptation set, and no `SegmentTimeline`. See the following examples.

For more information about compacted manifests, see Compacted DASH manifests.

> **Important**
>
> If the segments in a representation intentionally have varying sizes of segments, don't use the `duration` attribute. This treatment works only when the segments are a consistent size.

Example

The following is an example of a compacted manifest:

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true"
 subsegmentAlignment="true" startWithSAP="1" subsegmentStartsWithSAP="1"
 bitstreamSwitching="true">
   <SegmentTemplate timescale="30000" media="index_video_$RepresentationID
$_0_$Number$.mp4?m=1543947824" initialization="index_video_$RepresentationID
$_0_init.mp4?m=1543947824" startNumber="1">
     <SegmentTimeline>
       <S t="62000" d="60000" r="9"/>
     </SegmentTimeline>
   </SegmentTemplate>
   <Representation id="1" width="640" height="360" frameRate="30/1"
 bandwidth="749952" codecs="avc1.640029"/>
   <Representation id="2" width="854" height="480" frameRate="30/1"
 bandwidth="1000000" codecs="avc1.640029"/>
   <Representation id="3" width="1280" height="720" frameRate="30/1"
 bandwidth="2499968" codecs="avc1.640029"/>
</AdaptationSet>
```

The following is an example of a compacted manifest with the `duration` attribute:

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true"
 subsegmentAlignment="true" startWithSAP="1" subsegmentStartsWithSAP="1"
 bitstreamSwitching="true">
   <SegmentTemplate timescale="30000" media="index_video_$RepresentationID
$_0_$Number$.mp4?m=1543947824" initialization="index_video_$RepresentationID
$_0_init.mp4?m=1543947824" startNumber="1" duration="60000"/>
   <Representation id="1" width="640" height="360" frameRate="30/1"
 bandwidth="749952" codecs="avc1.640029"/>
   <Representation id="2" width="854" height="480" frameRate="30/1"
 bandwidth="1000000" codecs="avc1.640029"/>
   <Representation id="3" width="1280" height="720" frameRate="30/1"
 bandwidth="2499968" codecs="avc1.640029"/>
</AdaptationSet>
```

Manifest filtering

With manifest filtering, AWS Elemental MediaPackage dynamically produces client manifests based on parameters that you specify in a query appended to your playback request. This enables you to do things such as restrict viewer access to premium 4K HEVC content, or target specific device types and audio sample rate ranges, all from a single endpoint. Previously, you would have to configure multiple endpoints to accomplish this behavior. MediaPackage now provides a cost-effective way to dynamically produce different client manifests on the same endpoint.

Working with manifest filters

When you use a manifest filter, the resulting manifest includes only the audio and video streams that match the characteristics that you specify in your query. If no manifest filter is used, then all of the ingested streams are present in the endpoint output stream. The exception to this is if you have set stream filters for the endpoint, such as minimum video bitrate. In that case, the manifest filter is applied after the stream filter, which could skew your output, and is not recommended.

Manifest filtering can be used on all endpoint types supported by MediaPackage:

- Apple HLS
- DASH-ISO
- Microsoft Smooth Streaming
- CMAF

To use manifest filtering, append `aws.manifestfilter` query parameters to your playback request to MediaPackage. MediaPackage evaluates the query, and serves a client manifest based on those query parameters. Manifest queries are *not* case-sensitive and can be up to 1024 characters long. If the query is malformed, or if it there aren't streams that match the query parameters, MediaPackage returns an incomplete or empty manifest. For query syntax, see the following section.

> **Note**
>
> If you are using Apple HLS or CMAF endpoints, special conditions apply. For information about these conditions, see Special conditions for HLS and CMAF manifests.

Query syntax

The base query parameter is `aws.manifestfilter`, which is followed by optional parameter name and value pairs. To construct the query, append `?aws.manifestfilter=` to the end of the MediaPackage endpoint URL, followed by parameter names and values. For a list of all of the available parameters, see Manifest filter query parameters.

An Apple HLS filter query might look like this:

```
https://example-mediapackage-endpoint.mediapackage.us-
west-2.amazonaws.com/out/v1/examplemediapackage/index.m3u8?
aws.manifestfilter=audio_sample_rate:0-44100;video_bitrate:0-2147483647;video_co
US,de
```

The query syntax is listed in the following table.

Query string component	Description
`?`	A restricted character that marks the beginning of a query.
`aws.manifestfilter=`	The base query, which is followed by parameters constructed of name and value pairs. For a list of all of the available parameters, see Manifest filter query parameters.
`:`	Used to associate the parameter name with a value. For example, *`parameter_name :value`*.

Query string component	Description
;	Separates parameters in a query that contains multiple parameter s. For example, *parameter1_name:value ;parameter 2_name:minValue-maxValue* .
,	Separates a list of values. For example, parameter _name: *value1*,*value2*,*value3*. Comma-separated values in a list imply an OR relationship.
-	Used to define a parameter's minimum - maximum value range. For example, audio_sample_rate:0-44100 . When a numerical value is used in a range, it is included in the range definition. This means that streams must be greater than or equal to the minimum value, and less than or equal to the maximum value. With ranges, the minimum and maximum values are mandatory. The supported range values are 0 - 2147483647 .

Note

If you use Amazon CloudFront as your CDN, you might need to set additional configurations. For more information, see Configure cache behavior for all endpoints..

Manifest filter query parameters

MediaPackage supports the following query parameters.

Category	Name	Description	Example
Audio	audio_bitrate	• The audio bitrate in bits per second. • **Accepted values**: Two integers aggregated with a dash that define an inclusive range. The supported range values are 0 - 2147483647 .	stream.mpd?aws.manifestfilter=audio_bitrate:0-2147483647

Category	Name	Description	Example
Audio	`audio_channels`	• The number of audio channels. • **Accepted values**: Two integers aggregated with a dash that define an inclusive range. The supported range values are `1` - `32767`.	`stream.mpd?aws.manifestfilter=audio_channels:1-8`
Audio	`audio_codec`	• The audio codec type. • **Accepted values**: `AACL`, `AACH`, `AC-3`, `EC-3`. You must include the `-` for `AC-3` and `EC-3`. The values are *not* case-sensitive.	`stream.mpd?aws.manifestfilter=audio_codec:AACL,AC-3`
Audio	`audio_language`	• Audio languages or functional codes derived from encoder passthrough. • **Accepted values**: Arbitrary strings, such as two or four character ISO-639-1 language codes. You must use the same language strings that are set for your encoder. The values are *not* case-sensitive.	`stream.mpd?aws.manifestfilter=audio_language:fr,en-US,de`
Audio	`audio_sample_rate`	• The audio sample rate in Hz. • **Accepted values**: Two integers aggregated with a dash that define an inclusive range. The supported range values are `0` - `2147483647` .	`stream.mpd?aws.manifestfilter=audio_sample_rate:0-44100`

Category	Name	Description	Example
Subtitle	`subtitle_language`	• The subtitle language or functional codes derived from encoder passthrough. • **Accepted values**: Arbitrary strings, such as two or four character ISO-639-1 language codes. You must use the same language strings that are set for your encoder. The values are *not* case-sensitive.	`stream.mpd?aws.manifestfilter=subtitle_language:en-US, hi`
Video	`trickplay_height`	• The height of the trick-play image in pixels. This applies to both I-frame only and image-based trick-play. **Note** If you're using this parameter with I-frame only trick-play, `trickplay_height` and `video_height` should have similar values. If the values are not the same, I-frame only tracks might be removed from a manifest. • The `trickplay_height` filter is applied before tiling for DASH on VOD. • **Accepted values**: Two integers aggregated with a dash that define an inclusive range. The supported range values are 1 - 2147483647 .	`stream.mpd?aws.manifestfilter=trickplay_height:200-1200`

Category	Name	Description	Example
Video	`trickplay_type`	• The trickplay track type. You can filter on iframe or image trickplay tracks or use the value none to filter out all trickplay tracks (both iframe and image). • **Accepted values**: `iframe`, `image`, none. The values are *not* case-sensitive.	`stream.mpd?aws.manifestfilter=trickplay_type:iframe`

Category	Name	Description	Example
Video	`video_bitrate`	• The video bitrate in bits per second. **Note** If you're using this parameter, we recommend that you use only the `video_bitrate` filter parameter to set the video bitrate. Don't also set the minimum and maximum video bitrate via the MediaPackage console or AWS CLI. The `video_bitrate` filter applies to the video bitrate settings created at the endpoint. If you use the parameter and set the bitrate in the console or AWS CLI, your output might be skewed. • **Accepted values**: Two integers aggregated with a dash that define an inclusive range. The supported range values are `0` - `2147483647` . • **Note** You can't use this parameter with trick-play streams.	`stream.mpd?aws.manifestfilter=video_bitrate:0-2147483647`
Video	`video_codec`	• The video codec type. • **Accepted values**: `H264`, `H265`. The values are *not* case-sensitive.	`stream.mpd?aws.manifestfilter=video_codec:h264`

Category	Name	Description	Example
Video	`video_dynamic_range`	• The video dynamic range. • **Accepted values**: `hdr10`, `hlg`, `sdr`. The values are *not* case-sensitive.	`stream.mpd?aws.manifestfilter=video_dynamic_range:hdr10`
Video	`video_framerate`	• The video frame rate range in the NTSC format. • **Accepted values**: Two floating-point numbers aggregated with a dash that define an inclusive range. Each number can have up to three optional fractional values. For example, `29.97` or `29.764`. The supported range values are `1` - `999.999`.	`stream.mpd?aws.manifestfilter=video_framerate:23.976-30`
Video	`video_height`	• The height of the video in pixels. **Note** If you're using this parameter with I-frame only trick-play, `trickplay_height` and `video_height` should have similar values. If the values are not the same, I-frame only tracks might be removed from a manifest. • **Accepted values**: Two integers aggregated with a dash that define an inclusive range. The supported range values are `1` - `32767`.	`stream.mpd?aws.manifestfilter=video_height:720-1080`

Manifest filtering examples

These are manifest filtering examples.

Example 1: Target a player that supports AVC and a 44.1k audio sample rate

The viewer is playing content on a device that can only support AVC and a 44.1k audio sample rate. You set the `video_codec` and `audio_sample_rate` to filter out streams that don't fit these requirements.

```
?aws.manifestfilter=audio_sample_rate:0-44100;video_codec:h264
```

Example 2: Restrict 4k HEVC content

Your 4K HEVC stream is 15 Mbps, and all your other streams are less than 9 Mbps. To exclude the 4K stream from the stream set, you set a threshold of 9,000,000 bits per second to filter out the higher bitrate.

```
?aws.manifestfilter=video_bitrate:0-9000000
```

Example 3: Include video between 23.976 and 30 frames per second

To only include video within a certain frame rate range, use `video_framerate`. This parameter accepts floating-point numbers with up to three optional decimal values.

```
?aws.manifestfilter=video_framerate:23.976-30
```

Special conditions for HLS and CMAF manifests

If you are using HLS or CMAF manifests, these special conditions apply.

- For HLS manifests, we strongly recommend that you use audio rendition groups to avoid removing the video streams that are multiplexed with the audio streams that are filtered out. For more information about rendition groups, see Rendition groups reference in AWS Elemental MediaPackage.
- In HLS and CMAF manifests, the audio sample rate is not signaled, so it's not easy to visually check the original or filtered manifests for this setting. To verify the audio sample rate, check the audio sample rate at the encoder level and output level.
- In HLS and CMAF manifests, the `BANDWIDTH` attribute for a variant associates the bandwidth of the audio track with the video track, whether it is multiplexed with the video track, or if it is an audio rendition track referenced by the video track. Therefore, you can't visually inspect the

original and filtered manifests to confirm the `video_bitrate` filter has worked. To verify the filter, check the video bitrate at the encoder level and output level.

- For HLS and CMAF manifests, request parameters appended to bitrate playlists or segments result in an HTTP 400 error.

Error conditions

Some playback devices will return errors if the manifest or segments include invalid or unknown query parameters. The following are query parameters that MediaPackage can process:

- `m`
- `start`
- `end`
- `aws.manifestfilter`
- `aws.drmsettings`

If you have query parameters other than those listed, use a CDN such as Amazon CloudFront to remove the unnecessary parameters. For more information, see Cache content based on query string parameters in the *Amazon CloudFront Developer Guide*.

The following table contains additional common error conditions.

Error condition	Example	HTTP status code
A list parameter is not found and is not part of a constrained list	`?aws.manifestfilter=audio_language:dahlia`	200
Only subtitle streams are present in the stream	`?aws.manifestfilter=audio_sample_rate:0-1;video_bitrate=0-1`	200
Duplicate filter parameter	`?aws.manifestfilter=audio_sample_rate:0-48000;aws.mani`	400

Error condition	Example	HTTP status code
	`festfilter=audio_sample_rate:0-48000`	
Invalid parameter	`?aws.manifestfilter=donut_type:rhododendron`	400
Invalid range parameter	`?aws.manifestfilter=audio_sample_rate:300-0`	400
Invalid range value (more than `INT_MAX`)	`?aws.manifestfilter=audio_sample_rate:0-2147483648`	400
Malformed query string	`?aws.manifestfilter=audio_sample_rate:is:0-44100`	400
Parameter string is greater than 1024 characters	`?aws.manifestfilter=audio_language:abcdef....`	400
Query parameters on an HLS or CMAF bitrate manifest	`index_1.m3u8?aws.manifestfilter=video_codec:h264`	400
Query parameters on a segment request	`..._1.[ts|mp4|vtt.]?aws.manifestfilter=video_codec:h264`	400
Repeated query parameter	`?aws.manifestfilter=audio_sample_rate:0-48000;aws.manifestfilter=video_bitrate:0-1`	400

Error condition	Example	HTTP status code
Application of the filter results in an empty manifest (content has no streams that meet the conditions defined in the query string)	`?aws.manifestfilter=audio_sample_rate:0-1;video_bitrate=0-1`	400

Metadata passthrough

AWS Elemental MediaPackage automatically passes through ID3 and key-length-value (KLV) metadata from a channel's input to the channel's output stream. You don't need to adjust your endpoint's configuration to enable metadata passthrough.

For more information about how MediaPackage handles metadata, see the following sections.

Topics

- ID3 metadata considerations
- KLV metadata considerations

ID3 metadata considerations

Timed ID3 metadata is a general-purpose mechanism that adds synchronized metadata to streams. The metadata is used for a variety of purposes, ranging from interactive applications to audience measurement.

Supported MediaPackage endpoint types

MediaPackage supports ID3 metadata passthrough for the following endpoint types:

- Live and VOD HLS, DASH, and CMAF endpoints

Metadata carriage

Here is how ID3 is carried as metadata in the following specifications:

- HLS - Metadata is carried in the elementary stream. For more information, see section 2.0 of the Apple *Timed Metadata for HTTP Live Streaming* reference.

- CMAF and DASH - Metadata is carried in the Event Message box version 1. For more information, see Carriage of ID3 Timed Metadata in CMAF. Event Message boxes include a `scheme_id_uri` field set to `https://aomedia.org/emsg/ID3` and a `value` field set to `0`.

Metadata signaling

DASH manifests include a `<InbandEventStream schemeIdUri="https://aomedia.org/emsg/ID3" value="0"/>` element in AdaptationSets that include tracks with ID3 metadata.

HLS manifests don't have specific metadata signaling.

MediaLive configuration

You can produce ID3 metadata in AWS Elemental MediaLive MediaPackage output groups either by passing through ID3 metadata, or inserting ID3 metadata using the schedule.

KLV metadata considerations

KLV is a data encoding standard for including synchronized metadata in streams. The binary nature of KLV makes it efficient when the volume of metadata is significant. KLV can be used for various use cases ranging from aerial surveillance to transmitting sensors data in industry use cases, or for real-time athlete and object tracking in live sports use cases.

Supported MediaPackage endpoint types

MediaPackage supports KLV metadata passthrough for the following endpoint types:

- Live DASH endpoints

Metadata carriage

Metadata is carried in the Event Message box version 1, as described in the *MISB ST 1910.1 specification*. For synchronous KLV tracks, Event Message boxes include a `scheme_id_uri` field set to `urn:misb:KLV:bin:1910.1` and a `value` field set to `KLVx:01FC`. For asynchronous KLV tracks, the value field is set to `KLVx:01BD`. In both cases, `x` is the index of the track in the stream.

Metadata signaling

DASH manifests include a `<InbandEventStream schemeIdUri="urn:misb:KLV:bin:1910.1" value="KLVx:01FC"/>` or

<InbandEventStream schemeIdUri="urn:misb:KLV:bin:1910.1" value="KLVx:01BD"/> element in AdaptationSets that include tracks with KLV metadata, depending on the synchronicity nature of the carried track.

MediaLive configuration

You can pass through KLV metadata from your MediaLive channel. For more information, see klv in the *AWS Elemental MediaLive User Guide*.

Rendition groups reference in AWS Elemental MediaPackage

Rendition groups are used in HLS and CMAF outputs. A rendition group collects all subtitle or audio tracks and makes them available for all video renditions in the stream. When you enable rendition groups, MediaPackage pulls together all audio variants (such as different languages or codecs) and groups them for use with any video rendition. MediaPackage automatically puts subtitles into a rendition group.

Audio and subtitles tracks are required to be in their own rendition groups for CMAF outputs.

The following sections further describe when you can use rendition groups.

> **Note**
>
> DASH and Microsoft Smooth Streaming do not use rendition groups. This is because all audio, video, and subtitle or caption tracks are presented to the player, and the player determines which are used during playback.

When to use rendition groups

Rendition groups are used only in HLS and CMAF outputs. Rendition groups are most beneficial when you have multiple languages or multiple audio codecs in your streams. Rendition groups should be used in the following use cases:

> **Note**
>
> If you harvest a live-to-VOD asset from a live HLS stream with rendition groups, the groups are passed through to the asset as well.

- With CMAF outputs, if there are any audio or subtitle tracks

  CMAF requires all audio tracks in one rendition group, and all subtitles in another. Audio or subtitles can't be muxed with video tracks.

- One or more video tracks with multiple audio languages or codecs

  When rendition groups are enabled, MediaPackage pulls all audio renditions together for shared use between the video tracks. In this way, you don't have to duplicate all the audio options across all the video tracks.

- Multiple audio-only tracks and multiple subtitle tracks

  When both the audio tracks and subtitle tracks are in rendition groups, all the audio options can be combined with any subtitle track.

- One audio-only track and multiple subtitle tracks

  MediaPackage automatically pulls subtitle tracks into a rendition group so that the audio track can be used with any subtitle. Because there is only one audio and the subtitles are already grouped, you don't need to tell MediaPackage to use rendition groups in this case.

When not to use rendition groups

Rendition groups can't or shouldn't be used in the following use cases:

- Multiple video tracks in the stream, but only one language or codec is used for the audio. If the same audio is used with multiple video tracks, and rendition groups are also used, then your rendition group will have duplicates of the same audio track (one for each video).

  Keep the audio and video muxed in the stream, and do not use a rendition group.

- DASH or Microsoft Smooth Streaming outputs. These protocols do not support rendition groups. Instead, the output stream includes all tracks, and the player determines which to play based on rules from the player side or from the manifest (such as language or bitrate selection).

  To limit the tracks available to a player, use the stream selection options from the MediaPackage console or the MediaPackage API.

SCTE-35 message options in AWS Elemental MediaPackage

This section describes the options that AWS Elemental MediaPackage (MediaPackage) offers for configuring how SCTE-35 messages are handled in live HLS, DASH, and CMAF outputs. For live-to-VOD assets, MediaPackage passes the SCTE-35 messages from the live stream through to the harvested asset. These options don't apply to Microsoft Smooth Streaming or video on demand (VOD) outputs.

SCTE-35 messages accompany video in your source content. These messages signal where MediaPackage should insert ad markers when it packages the content for output. By default, MediaPackage inserts markers for the following message types in the source content:

- `splice_insert`
- `time_signal` with the following segmentation types:
  - Provider advertisement
  - Distributor advertisement
  - Provider placement opportunity
  - Distributor placement opportunity

  The `time_signal` must also include delivery restriction flags in the `segmentation_descriptor`.

When these commands are present, MediaPackage inserts corresponding ad markers in the output manifests:

- For HLS and CMAF outputs, MediaPackage inserts `EXT-X-CUE-OUT` and `EXT-X-CUE-IN` tags.
- For DASH outputs, MediaPackage inserts `EventStream` tags to create multiple periods, when you have multi-period manifests enabled.

The following sections describe how you can modify MediaPackage SCTE-35 message handling behavior.

SCTE-35 settings in MediaPackage

You can modify how MediaPackage interacts with SCTE-35 messages from your source content. Configure the following settings on your endpoints. For more information, see the following:

- For the MediaPackage console, see the section called "Creating an endpoint".
- For the MediaPackage API, see Origin_endpoints in the *AWS Elemental MediaPackage Live API Reference*.

> **Important**
>
> To modify how MediaPackage handles SCTE-35 messages, you should be familiar with the SCTE-35 standard. You can view the most recent standards here: SCTE Standards Catalog. You should also be familiar with how SCTE-35 is implemented in your source content.

Ad markers

This setting is available on HLS and CMAF endpoints.

Ad markers allows you to specify what MediaPackage does when it detects SCTE-35 messages. These are the options:

- **None** – MediaPackage ignores the SCTE-35 messages and doesn't include ad markers in the output manifest.
- **SCTE-35 enhanced** – MediaPackage includes ad markers and blackout tags in the output manifest for SCTE-35 messages that meet the requirements in **Customize ad triggers** and **Ads on delivery restrictions**.
- **Passthrough** – MediaPackage copies all SCTE-35 messages from the source content and inserts them in the output manifest.

Customize ad triggers

This setting is available on HLS, DASH, and CMAF endpoints.

Customize ad triggers identifies which SCTE-35 message types MediaPackage treats as ads in the output manifest.

If you don't change this setting, MediaPackage treats these message types as ads:

- Splice insert
- Provider advertisement
- Distributor advertisement

- Provider placement opportunity
- Distributor placement opportunity

Ads on delivery restrictions

This setting is available on HLS, DASH, and CMAF endpoints.

Ads on delivery restrictions sets conditions for what SCTE-35 messages become ads, based on the delivery restriction flags in the `segmentation_descriptor` of the messages. MediaPackage inserts an ad marker that corresponds to the positioning of the messages of the right type that meet the delivery restriction conditions.

If you don't change this setting, MediaPackage converts messages that are classified as *restricted* (they have delivery restriction flags) to ad markers in the output manifest.

> **Note**
>
> Splice insert SCTE-35 messages don't have `segmentation_descriptor`. If you choose splice insert in **Customize ad triggers**, all splice inserts become ad markers in the output manifest.

How it works

The **Ad markers**, **Customize ad triggers**, and **Ads on delivery restrictions** settings work together to determine what MediaPackage does with SCTE-35 messages from the source content.

When there are SCTE-35 messages in the source content, MediaPackage takes the following action based on the value that you selected in **Ad markers**:

- For **None**, MediaPackage does nothing with the SCTE-35 messages. No ad markers are inserted in the output manifest.
- For **Passthrough**, MediaPackage copies all SCTE-35 messages from the source content and inserts them in the output manifest.
- For **SCTE-35 enhanced**, MediaPackage checks for messages that meet the requirements that you set. In the output manifest, MediaPackage inserts ad markers that correspond to the applicable messages. To check for your requirements, MediaPackage does the following:
  1. Checks if any SCTE-35 messages match the message types that you indicated in **Customize ad triggers**

2. For messages of the right types, checks if the delivery restriction flags in `segmentation_descriptor` meet the conditions that you set in **Ads on delivery restrictions**
3. For messages of the right type that meet the delivery restriction conditions, inserts in the output manifest an ad marker for each message that has a unique SCTE segmentation ID, as described earlier in this chapter

- For **Daterange**, MediaPackage inserts `EXT-X-DATERANGE` tags to signal ads and program transition events in HLS and CMAF output manifests.

Important note on SCTE-35 data tracks

MediaPackage also signals SCTE-35 markers present in the source that are not ad markers. MediaPackage selects the first available data track from the input content for SCTE-35 signal processing (typically identified as PID 500). For proper handling by MediaPackage, ensure that your SCTE-35 ad signals are included in this first data track.

EXT-X-DATERANGE ad markers

Daterange ad markers are used to signal ads and program transitions in live HLS and CMAF manifests. When you enable daterange ad markers on your endpoint, MediaPackage inserts `EXT-X-DATERANGE` tags into the manifest where there are SCTE-35 `time_signal` or `splice_insert` tags present. `EXT-X-DATERANGE` is used in concert with `EXT-X-PROGRAM-DATE-TIME` tags.

For information about the `EXT-X-DATERANGE` and `EXT-X-PROGRAM-DATE-TIME` tags for HLS, see the HTTP Live Streaming 2nd Edition Specification.

Enabling daterange via the console

To enable daterange ad markers when creating or editing an endpoint, in the MediaPackage console, under **Packager settings** > **Additional configuration** > **Ad marker**, choose **Daterange**.

If you choose Daterange, you *must* also enter a **Program date/time interval (sec)** value that's greater than **0**. The program date/time interval is set in the same **Additional configuration** pane as the ad marker settings.

Enabling daterange via the AWS CLI

To enable daterange ad markers for your endpoint, run the following command in the AWS CLI replacing *region* with your own information:

```
 aws --endpoint=https://mediapackage.region.amazonaws.com mediapackage --region region
create-origin-endpoint --channel-id test_channel --id hlsmuxed
 --hls-package "{\"ProgramDateTimeIntervalSeconds\":60,\"AdMarkers\":\"DATERANGE\"}"
```

Important

You must set a `ProgramDateTimeIntervalSeconds` value that's greater than **`0`**.

Enabling daterange via the MediaPackage API or AWS SDK

To learn how to enable daterange ad markers for HLS endpoints via the MediaPackage live API or AWS SDK, see the following:

- MediaPackage Live API reference
- AWS SDK

Example HLS manifest showing SCTE-35 EXT-X-DATERANGE signaling

This example HLS manifest generated by MediaPackage uses `EXT-X-DATERANGE` and `EXT-X-PROGRAM-DATE-TIME` tags to signal events in the live stream.

Note

The `DURATION`, `PLANNED-DURATION`, and `END-DATE` attributes of the `EXT-X-DATERANGE` tag are optional. If these attributes aren't present in the SCTE-35 input, or aren't set when you create your endpoint via the MediaPackage API, then they are omitted from the generated manifests.

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-TARGETDURATION:8
#EXT-X-MEDIA-SEQUENCE:11
#EXT-X-DATERANGE:ID="2415919105",START-DATE="2020-05-03T00:01:00.018Z",PLANNED-
DURATION=29.988,SCTE35-
OUT=0xFC303000000002CDE400FFF00506FE00526C14001A021843554549900000017FC00000292EA80A04ABCD00013
```

```
#EXT-X-DATERANGE:ID="2147483649",START-DATE="2020-05-03T00:00:30.030Z",PLANNED-
DURATION=90.006,SCTE35-
CMD=0xFC303000000002CDE400FFF00506FE00293D6C001A021843554549800000017FFF00007B9ABC0A04ABCD00011
#EXT-X-PROGRAM-DATE-TIME:2020-05-03T00:01:08.040Z
#EXTINF:7.560,
../../../../index_1_11.ts?m=1588607409
#EXTINF:7.560,
../../../../index_1_12.ts?m=1588607409
#EXTINF:6.846,
../../../../index_1_13.ts?m=1588607409
#EXT-X-DATERANGE:ID="2415919105",START-DATE="2020-05-03T00:01:00.018Z",END-
DATE="2020-05-03T00:01:30.006Z",DURATION=29.988
#EXTINF:0.714,
../../../../index_1_14.ts?m=1588607409
#EXTINF:7.560,
../../../../index_1_15.ts?m=1588607409
#EXTINF:7.560,
../../../../index_1_16.ts?m=1588607409
#EXTINF:7.560,
../../../../index_1_17.ts?m=1588607409
#EXTINF:6.636,
../../../../index_1_18.ts?m=1588607409
#EXT-X-DATERANGE:ID="2147483649",START-DATE="2020-05-03T00:00:30.030Z",END-
DATE="2020-05-03T00:02:00.036Z",DURATION=90.006,SCTE35-
CMD=0xFC304A00000002CDE400FFF00506FE00A4D828003402184355454980000017FC000000000000A04ABCD00011
#EXT-X-DATERANGE:ID="2147483650",START-DATE="2020-05-03T00:02:00.036Z",PLANNED-
DURATION=90.006,SCTE35-
CMD=0xFC304A00000002CDE400FFF00506FE00A4D828003402184355454980000017FC000000000000A04ABCD00011
#EXTINF:0.924,
../../../../index_1_19.ts?m=1588607409
#EXTINF:7.560,
../../../../index_1_20.ts?m=1588607409
#EXT-X-PROGRAM-DATE-TIME:2020-05-03T00:02:08.520Z
#EXTINF:7.560,
../../../../index_1_21.ts?m=1588607409
#EXT-X-ENDLIST
```

Time-shifted viewing reference in AWS Elemental MediaPackage

Time-shifted viewing is available with live workflows in AWS Elemental MediaPackage.

Time-shifted viewing means that viewers can start watching a live stream at a time earlier than "now," allowing them to join from the beginning a show that's already in progress or to watch a show that's already completed. MediaPackage supports time-shifted viewing for content that's up to 336 hours (14 days) old. You can enable time-shifted viewing for some or all of this content by defining the **startover window** on the endpoint. Content that falls within that window is available for playback when playback requests include valid start and end parameters. Requests for content outside the window configured on the endpoint result in an HTTP error 404.

Alternatively, you can harvest a clip of a live stream and make it available as a video on demand (VOD) asset. For information about harvesting VOD assets, see Creating live-to-VOD assets with AWS Elemental MediaPackage.

In the following steps, "now" is the current time according to the program date time (PDT), when it's present in the source content from the encoder. If the source content doesn't include PDT information, "now" refers to the MediaPackage ingest time of the most recent segment.

> **Important**
>
> Create a new MediaPackage channel to harvest content from when there is a change to the stream in the upstream encoder (such as changes to the stream name, type, or codec). If you don't use a new channel and the start and end times of the startover window span the change, the time-shifted manifest could behave in unexpected ways.

To enable time-shifted viewing

1. Enable time-shifted viewing by typing a value for **Startover window** on the MediaPackage endpoint object. You can do this through either the MediaPackage console or the MediaPackage API.

   When requests with start and end parameters that are within the startover window are sent to this endpoint, MediaPackage generates a manifest for the requested timeframe. If the start or end parameters are outside of the startover window, the playback request fails. If no start and end parameters are used, the service generates a standard manifest.

   > **Note**
   >
   > You might notice that the manifest lags behind real time when you initially create a startover window on an endpoint. This is because MediaPackage starts filling the

manifest from the start of the window, and works up to "now." So, if you have a 24-hour startover window, MediaPackage fills the manifest starting 24 hours ago and working up to "now."

2. Ensure that content requests contain start and end parameters as needed. MediaPackage accepts requests for up to 24 hours of content.

   For packager-specific rules about how you can notate the parameters, see Rules for start and end parameters.

   The start and end parameters determine the time boundaries of the manifest. These are the expected behaviors based on request start and end parameters:

   - If both start and end parameters are used in the URL, the resulting manifest has a fixed start and end time that correspond to the specified start and end parameters.

     If the end time is in the future, the tags in the manifest are consistent with a live manifest. Otherwise, if the end time is in the past, the tags in the manifest are consistent with a video on demand (VOD) manifest. For information about the manifest differences, see Live and VOD manifest reference.
   - If a start parameter is specified but not an end, the resulting manifest has a fixed start time that corresponds to the specified start parameter, and the end of the manifest grows as the live content progresses.

     Note

     For HLS output, many playback devices start playback at the current time ("now"). To view the content from the actual start time of the playback window, viewers can seek back on the playback progress bar.
   - If no parameters are specified, a standard manifest is generated starting "now" with no end time.
   - If an end parameter is specified but no start, the manifest is generated in the same way as when no parameters are specified. The manifest starts "now" and has no end time.

Important

When using time-shifted viewing, we recommend using consistent playback windows across player sessions, rather than generating a unique start or end time for each viewer. This yields better caching at the CDN, and will avoid running into potential throttling related to those requests, on the MediaPackage level.

Rules for start and end parameters

Start and end parameters denote the beginning and end of a time-shifted manifest. The playback device can append parameters to the end of a manifest request or include the parameters within the request.

In all cases, the date and time must be notated in one of the following formats:

- ISO 8601 dates, such as 2017-08-18T21:18:54+00:00. Where -08:00 is the timezone UTC -08:00.
- POSIX (or Epoch) time, such as 1503091134

The following topics describe the location rules by packager type.

DASH parameter rules

Start and end parameters in the URL request for DASH content can use standard parameter notation, or can be included as path elements in the URL.

- Query parameter notation – start and end parameters are included at the end of the request URL

  Example

```
https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
v1/997cbb27697d4863bb65488133bff26f/sports.mpd?start=1513717228&end=1513720828
```

- Path elements – start and end parameters are included in the path of the request URL

Example

```
https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
v1/997cbb27697d4863bb65488133bff26f/start/2017-12-19T13:00:28-08:00/end/
2017-12-19T14:00:28-08:00/sports.mpd
```

HLS and CMAF parameter rules

Start and end parameters in the URL request for HLS content can use standard parameter notation, or can be included as path elements in the URL. The rules for HLS and CMAF are the same, except that when you're inserting path elements in the CMAF endpoint, the elements have to be after the manifest ID in the URL.

- Query parameter notation – start and end parameters are included at the end of the request URL

  Example HLS

  ```
  https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
  v1/064134724fd74667ba294657a674ae72/
  comedy.m3u8?start=2017-12-19T13:00:28-08:00&end=2017-12-19T14:00:28-08:00
  ```

  Example CMAF

  ```
  https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
  v1/064134724fd74667ba294657a674ae72/manifest_id/
  news.m3u8?start=2018-04-04T01:14:00-08:00&end=2018-04-04T02:15:00-08:00
  ```

- Path elements – start and end parameters are included in the path of the request URL

  Example HLS

  ```
  https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
  v1/064134724fd74667ba294657a674ae72/start/1513717228/end/1513720828/comedy.m3u8
  ```

Example CMAF

```
https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
v1/064134724fd74667ba294657a674ae72/manifest_id/start/1522807213/end/1522800013/
news.m3u8
```

Microsoft Smooth Streaming parameter rules

Start and end parameters in the URL request for Microsoft Smooth Streaming content can be included as path elements in the URL.

- Path elements – start and end parameters are included in the path of the request URL

Example

```
https://cf98fa7b2ee4450e.mediapackage.us-east-1.amazonaws.com/out/
v1/1f76b3b4f94c44a485c0e4e560afe50e/start/1513717228/end/1513720828/drama.ism/
Manifest
```

Working with trick-play in AWS Elemental MediaPackage

Trick-play, sometimes called trick mode, provides a visual cue to viewers as they rewind, fast-forward, or seek through content in a digital video player. This helps the person using the video player to visualize where they are in the content timeline.

AWS Elemental MediaPackage supports I-frame and image-based trick-play for live and video on demand (VOD) workflows. For I-frame trick-play, MediaPackage generates an I-frame track from the first rendition in your HLS multivariant playlist. For image-based trick-play, MediaPackage passes through the image media playlist that you configure in your upstream encoder. To learn how to use I-frame and image-based trick-play for MediaPackage, see the sections in this topic.

MediaPackage supports the following trick-play types:

Supported trick-play types for live workflows

Streaming protocol	I-frame only	Image-based
Apple HLS	√	√
CMAF Apple HLS	√	√
DASH	√	√

Supported trick-play types for VOD workflows

Streaming protocol	I-frame only	Image-based
Apple HLS	√	√
CMAF Apple HLS	√	√
DASH	√	√

Topics

Using I-frame playlists to enable trick-play

MediaPackage supports live and on-demand trick-play by creating an I-frame playlist from an existing VOD asset or live stream. The I-frame playlist contains the I-frame only video segments that your player uses for the image thumbnails. For information about I-frame playlists, see the HTTP Live Streaming 2nd Edition specification: https://datatracker.ietf.org/doc/html/rfc8216#section-4.3.3.6.

To use an I-frame playlist to enable trick-play

- In the MediaPackage console, choose **Include I-frame only stream** when creating or editing an endpoint or packaging configuration. MediaPackage generates an I-frame only stream from the first rendition in the manifest. The service inserts `EXT-I-FRAMES-ONLY` tags in the output

manifest, and then generates and includes an I-frames only playlist in the stream. This playlist enables player functionality like fast forward and rewind.

Using image media playlists to enable trick-play

To use image-based trickplay, in your upstream encoder you create an HLS *image media playlist* that contains JPEG image segments. MediaPackage automatically passes through the image segments to the output. These segments are the thumbnail images and image metadata that the video player uses for visual cues. These segments must conform to the Image Media Playlist specification, version 0.4. The service supports the time-based implementation of the specification.

For information about how to configure your upstream encoder to generate an image media playlist, see Configuring your upstream encoder to generate image media playlists.

Input source requirements

Your HLS source content must meet the following requirements:

- The HLS parent playlist that references the image playlist must include the `EXT-X-IMAGE-STREAM-INF` tag.
- The image playlist must include the following tags:
  - An `EXT-X-IMAGES-ONLY` tag above the segment list.
  - If using tiled thumbnails, `EXT-X-TILES` tags above each image segment that specifies the tiling information. Tiled thumbnails are only available for VOD workflows.

    Note

    We recommend that you use decimal durations in the `EXT-INF` and `EXT-X-TILES` tags to help MediaPackage give players the most accurate image durations.
- You must use image segments that are valid JPEG image files less than 20 MB. For tiled thumbnails, the image segments can be tiled, with multiple thumbnails in a grid in the JPEG, or a single tile can occupy the entire JPEG.
  - For live, each JPEG must contain only one image segment. The encoder must produce image segments and video segments at the same cadence.

You can use AWS Media Services to generate an HLS source in your upstream encoder that complies with the Image Media Playlist specification, version 0.4. For more information, see the following section Configuring your upstream encoder to generate image media playlists.

Limitations

Keep in mind the following limitations when using image-based trick-play for MediaPackage:

- MediaPackage doesn't combine image segments for packaging configurations. For example, if the service ingests a VOD asset with an image asset with a 2 second segment duration, and you specify a segment output duration of 6 seconds, we combine the video and audio segments to be 6 seconds long, but image segments will remain 2 seconds.
- Depending on your HLS player requirements, the use of `EXT-X-PROGRAM-DATE-TIME` tags might be necessary to display the trick-play image. This applies to live and VOD workflows.

Considerations when using image-based trick-play for DASH

MediaPackage supports single or tiled thumbnails for VOD workflows, and single thumbnails for live workflows. Your HLS content must conform to the Image Media Playlist specification, version 0.4. See the following paragraph for specific requirements. When MediaPackage outputs content from a DASH packaging configuration or endpoint, the service outputs thumbnails based on the DASH-IF Interoperability Points specification, v4.3, section 6.2.6.

In addition to the general requirements listed before this section, keep in mind the following requirements and limitations when using trick-play for DASH.

- MediaPackage only supports DASH tiled thumbnails for VOD workflows.
- In general, the service doesn't support multi-period DASH for packaging configurations that use `NUMBER_WITH_DURATION` because it impacts segment alignment. This limitation also applies to trick-play.
- The service generates the image segment time format for live and VOD as follows:
  - For live, the image segment's time format is the same as your endpoint's time format for audio and video segments. This format is set by the **segment template format** on your endpoint. For example, if your endpoint has a segment template format of `NUMBER_WITH_TIMELINE`, the image segment uses `NUMBER_WITH_TIMELINE` for the time format.
  - For VOD, the image segment uses `NUMBER_WITH_DURATION` regardless of which time format you set for your packaging configuration. For example, if you choose the

`NUMBER_WITH_TIMELINE` segment template format for your packaging configuration, the service will use `NUMBER_WITH_TIMELINE` for video and audio Adaptation Sets, but will use `NUMBER_WITH_DURATION` for the image Adaptation Sets.

Configuring your upstream encoder to generate image media playlists

Your HLS source must conform to the Image Media Playlist specification, version 0.4. You can use the following AWS Media Services to create an HLS stream that complies with the specification. For more information, see the following documentation:

- Trick-play track via the Image Media Playlist specification in the *Elemental Live User Guide*.
- Trick-play track via the Image Media Playlist specification in the *AWS Elemental MediaLive User Guide*.
- HlsImageBasedTrickPlay in the *AWS Elemental MediaConvert API Reference*.

Security in AWS Elemental MediaPackage

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that's 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. To learn about the compliance programs that apply to AWS Elemental MediaPackage, see AWS Services in Scope by Compliance Program.
- **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 company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using MediaPackage. The following topics show you how to configure MediaPackage to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your MediaPackage resources.

Topics

Data protection in AWS Elemental MediaPackage

The AWS shared responsibility model applies to data protection in AWS Elemental MediaPackage. As described in this model, AWS is responsible for protecting the global infrastructure that runs

all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see Data Privacy FAQ. For information about data protection in Europe, see the General Data Protection Regulation (GDPR) Center.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with MediaPackage or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Topics

- Implementing DRM with AWS Elemental MediaPackage
- Implementing CDN authorization with AWS Elemental MediaPackage

Implementing DRM with AWS Elemental MediaPackage

Use encryption to protect your content from unauthorized access. MediaPackage supports digital rights management (DRM). With DRM, you can make sure that once you distribute your content, only authorized viewers can watch it.

For information about using DRM with MediaPackage, see Content encryption and DRM in AWS Elemental MediaPackage.

Implementing CDN authorization with AWS Elemental MediaPackage

Use content delivery network (CDN) authorization to ensure only authorized devices can access your content. With CDN authorization, playback requests must include the appropriate header and authorization code that you create. MediaPackage refuses playback requests that don't include the correct code.

For more information about CDN authorization, see CDN authorization in AWS Elemental MediaPackage.

Identity and Access Management for AWS Elemental MediaPackage

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use MediaPackage resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How AWS Elemental MediaPackage works with IAM
- Identity-based policy examples for MediaPackage
- IAM policy examples for secrets in AWS Secrets Manager

- Cross-service confused deputy prevention
- Troubleshooting MediaPackage identity and access
- Learn More
- Using Service-Linked Roles for MediaPackage

Audience

How you use AWS Identity and Access Management (IAM) differs based on your role:

- **Service user** - request permissions from your administrator if you cannot access features (see Troubleshooting MediaPackage identity and access)
- **Service administrator** - determine user access and submit permission requests (see How AWS Elemental MediaPackage works with IAM)
- **IAM administrator** - write policies to manage access (see Identity-based policy examples for MediaPackage)

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated as the AWS account root user, an IAM user, or by assuming an IAM role.

You can sign in as a federated identity using credentials from an identity source like AWS IAM Identity Center (IAM Identity Center), single sign-on authentication, or Google/Facebook credentials. For more information about signing in, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

For programmatic access, AWS provides an SDK and CLI to cryptographically sign requests. For more information, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity called the AWS account *root user* that has complete access to all AWS services and resources. We strongly recommend that you don't use the root user for everyday tasks. For tasks that require root user credentials, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users to use federation with an identity provider to access AWS services using temporary credentials.

A *federated identity* is a user from your enterprise directory, web identity provider, or Directory Service that accesses AWS services using credentials from an identity source. Federated identities assume roles that provide temporary credentials.

For centralized access management, we recommend AWS IAM Identity Center. For more information, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity with specific permissions for a single person or application. We recommend using temporary credentials instead of IAM users with long-term credentials. For more information, see Require human users to use federation with an identity provider to access AWS using temporary credentials in the *IAM User Guide*.

An *IAM group* specifies a collection of IAM users and makes permissions easier to manage for large sets of users. For more information, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity with specific permissions that provides temporary credentials. You can assume a role by switching from a user to an IAM role (console) or by calling an AWS CLI or AWS API operation. For more information, see Methods to assume a role in the *IAM User Guide*.

IAM roles are useful for federated user access, temporary IAM user permissions, cross-account access, cross-service access, and applications running on Amazon EC2. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy defines permissions when associated with an identity or resource. AWS evaluates these policies when a principal makes a request. Most policies are stored in AWS as JSON documents. For more information about JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Using policies, administrators specify who has access to what by defining which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. An IAM administrator creates IAM policies and adds them to roles, which users can then assume. IAM policies define permissions regardless of the method used to perform the operation.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you attach to an identity (user, group, or role). These policies control what actions identities can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be *inline policies* (embedded directly into a single identity) or *managed policies* (standalone policies attached to multiple identities). To learn how to choose between managed and inline policies, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples include IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. You must specify a principal in a resource-based policy.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Other policy types

AWS supports additional policy types that can set the maximum permissions granted by more common policy types:

- **Permissions boundaries** – Set the maximum permissions that an identity-based policy can grant to an IAM entity. For more information, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – Specify the maximum permissions for an organization or organizational unit in AWS Organizations. For more information, see Service control policies in the *AWS Organizations User Guide*.

- **Resource control policies (RCPs)** – Set the maximum available permissions for resources in your accounts. For more information, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Advanced policies passed as a parameter when creating a temporary session for a role or federated user. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS Elemental MediaPackage works with IAM

Before you use IAM to manage access to MediaPackage, learn what IAM features are available to use with MediaPackage.

IAM features you can use with MediaPackage

IAM feature	MediaPackage support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys (service-specific)	Yes
ACLs	No
ABAC (tags in policies)	Yes
Temporary credentials	Yes
Principal permissions	Yes

IAM feature	MediaPackage support
Service roles	Yes
Service-linked roles	Partial

To get a high-level view of how MediaPackage and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for MediaPackage

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for MediaPackage

To view examples of MediaPackage identity-based policies, see Identity-based policy examples for MediaPackage.

Resource-based policies within MediaPackage

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for MediaPackage

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Include actions in a policy to grant permissions to perform the associated operation.

To see a list of MediaPackage actions, see Actions defined by AWS Elemental MediaPackage in the *Service Authorization Reference*.

Policy actions in MediaPackage use the following prefix before the action:

```
mediapackage
```

To specify multiple actions in a single statement, separate them with commas.

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

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Describe`, include the following action:

```
"Action": "mediapackage:Describe*"
```

To view examples of MediaPackage identity-based policies, see Identity-based policy examples for MediaPackage.

Policy resources for MediaPackage

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. As a best practice, specify a resource using its Amazon Resource Name (ARN). For actions that don't support resource-level permissions, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

MediaPackage has the following resource ARNs:

```
arn:${Partition}:mediapackage:${Region}:${Account}:channels/${channelID}
arn:${Partition}:mediapackage:${Region}:${Account}:origin_endpoints/${endpointID}
```

For more information about the format of ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces.

For example, to specify the `9a6b3953e242400eb805f324d95788e3` channel in your statement, use the following ARN:

```
"Resource": "arn:aws:mediapackage:us-
east-1:111122223333:channels/9a6b3953e242400eb805f324d95788e3"
```

To specify all instances that belong to a specific account, use the wildcard (*):

```
"Resource": "arn:aws:mediapackage:us-east-1:111122223333:channels/*"
```

Some MediaPackage actions, such as those for creating resources, can't be performed on a specific resource. In those cases, you must use the wildcard (*).

```
"Resource": "*"
```

To see a list of MediaPackage resource types and their ARNs, see Resources defined by AWS Elemental MediaPackage in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions defined by AWS Elemental MediaPackage.

To view examples of MediaPackage identity-based policies, see Identity-based policy examples for MediaPackage.

Policy condition keys for MediaPackage

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element specifies when statements execute based on defined criteria. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of MediaPackage condition keys, see Condition keys for AWS Elemental MediaPackage in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions defined by AWS Elemental MediaPackage.

To view examples of MediaPackage identity-based policies, see Identity-based policy examples for MediaPackage.

ACLs in MediaPackage

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with MediaPackage

Supports ABAC (tags in policies): Yes

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes called tags. You can attach tags to IAM entities and AWS resources, then design ABAC policies to allow operations when the principal's tag matches the tag on the resource.

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.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with MediaPackage

Supports temporary credentials: Yes

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Cross-service principal permissions for MediaPackage

Supports forward access sessions (FAS): Yes

Forward access sessions (FAS) use the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. For policy details when making FAS requests, see Forward access sessions.

Service roles for MediaPackage

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break MediaPackage functionality. Edit service roles only when MediaPackage provides guidance to do so.

Choosing an IAM role in MediaPackage

When you create an asset resource in MediaPackage, you must choose a role to allow MediaPackage to access Amazon S3 on your behalf. If you previously created a service role or service-linked role, MediaPackage provides you with a list of roles to choose from. It's important to choose a role that allows access to read from the Amazon S3 bucket and retrieve content. For more information, see Allowing AWS Elemental MediaPackage to access other AWS services.

Service-linked roles for MediaPackage

Supports service-linked roles: Partial

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for MediaPackage

By default, users and roles don't have permission to create or modify MediaPackage resources. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by MediaPackage, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for AWS Elemental MediaPackage in the *Service Authorization Reference*.

Topics

- Policy best practices
- Using the MediaPackage console
- Allow users to view their own permissions

Policy best practices

Identity-based policies determine whether someone can create, access, or delete MediaPackage resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the MediaPackage console

To access the AWS Elemental MediaPackage console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the MediaPackage resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the MediaPackage console, also attach the MediaPackage *ReadOnly* AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

`AWSElementalMediaPackageReadOnly`

Allow users to view their own permissions

This example shows how you might create 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 console or programmatically using the AWS CLI or AWS API.

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

```
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM policy examples for secrets in AWS Secrets Manager

During setup, you create an IAM policy that you assign to AWS Elemental MediaPackage. This policy allows AWS Elemental MediaPackage to read secrets that you have stored in AWS Secrets Manager. The settings for this policy are entirely up to you. The policy can range from most restrictive (allowing access to only specific secrets) to least restrictive (allowing access to any secret that you create using this AWS account). We recommend using the most restrictive policy as a best practice. However, the examples in this section show you how to set up policies with different levels of restriction. Because AWS Elemental MediaPackage needs only read access to secrets, all the examples in this section show only the actions necessary to read the values that you store.

Topics

- Allow read access to specific secrets in AWS Secrets Manager
- Allow read access to all secrets created in a specific Region in AWS Secrets Manager
- Allow read access to all resources in AWS Secrets Manager

Allow read access to specific secrets in AWS Secrets Manager

The following IAM policy allows read access to specific resources (secrets) that you create in AWS Secrets Manager.

JSON

```
{
            "Version":"2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Action": [
                  "secretsmanager:GetResourcePolicy",
                  "secretsmanager:GetSecretValue",
                  "secretsmanager:DescribeSecret",
                  "secretsmanager:ListSecretVersionIds"
                ],
                "Resource": [
                  "arn:aws:secretsmanager:us-
west-2:111122223333:secret:aes128-1a2b3c",
                  "arn:aws:secretsmanager:us-
west-2:111122223333:secret:aes192-4D5e6F",
                  "arn:aws:secretsmanager:us-
west-2:111122223333:secret:aes256-7g8H9i"
                ]
              }
            ]
          }
```

Allow read access to all secrets created in a specific Region in AWS Secrets Manager

The following IAM policy allows read access to all secrets that you create in a specific AWS Region in AWS Secrets Manager. This policy applies to resources that you have created already and all resources that you create in the future in the specified Region.

JSON

```
{
            "Version":"2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Action": [
```

```
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": [
              "arn:aws:secretsmanager:us-west-2:111122223333:secret:*"
            ]
          }
        ]
      }
```

Allow read access to all resources in AWS Secrets Manager

The following IAM policy allows read access to all resources that you create in AWS Secrets Manager. This policy applies to resources that you have created already and all resources that you create in the future.

JSON

```
{
        "Version":"2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "secretsmanager:GetResourcePolicy",
              "secretsmanager:GetSecretValue",
              "secretsmanager:DescribeSecret",
              "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": ["*"]
          }
        ]
      }
```

Cross-service confused deputy prevention

The confused deputy problem is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action. In AWS, cross-service impersonation can result in the confused deputy problem. Cross-service impersonation can occur when one service (the *calling service*) calls another service (the *called service*). The calling service can be manipulated to use its permissions to act on another customer's resources in a way it should not otherwise have permission to access. To prevent this, AWS provides tools that help you protect your data for all services with service principals that have been given access to resources in your account.

We recommend using the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in resource policies to limit the permissions that AWS Elemental MediaPackage gives another service to the resource. Use `aws:SourceArn` if you want only one resource to be associated with the cross-service access. Use `aws:SourceAccount` if you want to allow any resource in that account to be associated with the cross-service use.

The most effective way to protect against the confused deputy problem is to use the `aws:SourceArn` global condition context key with the full ARN of the resource. If you don't know the full ARN of the resource or if you are specifying multiple resources, use the `aws:SourceArn` global context condition key with wildcard characters (*) for the unknown portions of the ARN. For example, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

If the `aws:SourceArn` value does not contain the account ID, such as an Amazon S3 bucket ARN, you must use both global condition context keys to limit permissions.

The following example shows how you can use the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in MediaPackage to prevent the confused deputy problem when working with harvest jobs.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "mediapackage.amazonaws.com"
```

```
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:mediapackage:*:123456789012:harvest_jobs/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

Troubleshooting MediaPackage identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with MediaPackage and IAM.

Topics

- I'm not authorized to perform an action in MediaPackage
- I'm not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my MediaPackage resources

I'm not authorized to perform an action in MediaPackage

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `mediapackage:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 mediapackage:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `mediapackage:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I'm not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to MediaPackage.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in MediaPackage. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

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

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my MediaPackage resources

You can create a role that users in other accounts or people outside of 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.

To learn more, consult the following:

- To learn whether MediaPackage supports these features, see How AWS Elemental MediaPackage works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.

- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Learn More

For more information about identity and access management for MediaPackage, continue to the following pages:

- How AWS Elemental MediaPackage works with IAM
- Identity-based policy examples for MediaPackage
- Troubleshooting MediaPackage identity and access

Using Service-Linked Roles for MediaPackage

AWS Elemental MediaPackage uses IAM service-linked roles. A service-linked role is a unique type of IAM role that's linked directly to MediaPackage. Service-linked roles are predefined by MediaPackage and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up MediaPackage easier because you don’t have to manually add the necessary permissions. MediaPackage defines the permissions of its service-linked roles, and unless defined otherwise, only MediaPackage 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 a service-linked role only after first deleting their related resources. This protects your MediaPackage resources because you can't inadvertently remove permission to access the resources.

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 MediaPackage

MediaPackage uses the service-linked role named **AWSServiceRoleForMediaPackage** – MediaPackage uses this service-linked role to invoke CloudWatch to create and manage log groups, log streams, and log events.

The AWSServiceRoleForMediaPackage service-linked role trusts the following services to assume the role:

- `mediapackage.amazonaws.com`

The role permissions policy allows MediaPackage to complete the following actions on the specified resources:

- Action: `logs:PutLogEvents` on `arn:aws:logs:*:*:log-group:/aws/MediaPackage/*:log-stream:*`
- Action: `logs:CreateLogStream, logs:CreateLogGroup, logs:DescribeLogGroups, logs:DescribeLogStreams` on `arn:aws:logs:*:*:log-group:/aws/MediaPackage/*`

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 MediaPackage

You don't need to manually create a service-linked role. When you enable access logging in the AWS Management Console, the AWS CLI, or the AWS API, MediaPackage creates the service-linked role for you.

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 enable access logging, MediaPackage creates the service-linked role for you again.

You can also use the IAM console to create a service-linked role with the **MediaPackage** use case. In the AWS CLI or the AWS API, create a service-linked role with the `mediapackage.amazonaws.com` service name. For more information, see Creating a Service-Linked Role in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a Service-Linked Role for MediaPackage

MediaPackage does not allow you to edit the AWSServiceRoleForMediaPackage 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 MediaPackage

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's 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 MediaPackage 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 MediaPackage resources used by the AWSServiceRoleForMediaPackage

- Disable access logging in the AWS Management Console, the AWS CLI, or the AWS API.

To manually delete the service-linked role using IAM

- Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForMediaPackage service-linked role. For more information, see Deleting a Service-Linked Role in the *IAM User Guide*.

Supported Regions for MediaPackage Service-Linked Roles

MediaPackage supports using service-linked roles in all of the regions where the service is available. For more information, see AWS Regions and Endpoints.

Logging and monitoring in AWS Elemental MediaPackage

This section provides an overview of the options for logging and monitoring in AWS Elemental MediaPackage for security purposes. For more information about logging and monitoring in MediaPackage see Logging and monitoring in AWS Elemental MediaPackage.

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Elemental MediaPackage and your AWS solutions. You should collect monitoring data from all parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. AWS provides several tools for monitoring your MediaPackage resources and responding to potential incidents.

Amazon CloudWatch alarms

Using CloudWatch alarms, you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, a notification is sent to an Amazon SNS topic or AWS Auto Scaling policy. CloudWatch alarms do not invoke actions because they are in a particular state. Rather, the state must have changed and been maintained for a specified number of periods. For more information, see Monitoring AWS Elemental MediaPackage with Amazon CloudWatch metrics.

AWS CloudTrail logs

CloudTrail provides a record of actions taken by a user, role, or an AWS service in AWS Elemental MediaPackage. Using the information collected by CloudTrail, you can determine the request that was made to MediaPackage, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging AWS Elemental MediaPackage API calls with AWS CloudTrail.

AWS Elemental MediaPackage access logs

Server access logs provide detailed records about requests that are made to a channel. Server access logs are useful for many applications. For example, access log information can be useful in security and access audits. For more information, see Access logging.

AWS Trusted Advisor

Trusted Advisor draws upon best practices learned from serving hundreds of thousands of AWS customers. Trusted Advisor inspects your AWS environment and then makes recommendations

when opportunities exist to save money, improve system availability and performance, or help close security gaps. All AWS customers have access to five Trusted Advisor checks. Customers with a Business or Enterprise support plan can view all Trusted Advisor checks.

For more information, see AWS Trusted Advisor.

Compliance validation for AWS Elemental MediaPackage

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. For more information about your compliance responsibility when using AWS services, see AWS Security Documentation.

Resilience in AWS Elemental MediaPackage

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS global infrastructure.

Infrastructure security in AWS Elemental MediaPackage

As a managed service, AWS Elemental MediaPackage is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access MediaPackage through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Logging and monitoring in AWS Elemental MediaPackage

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Elemental MediaPackage and your other AWS solutions. AWS provides the following monitoring tools to watch MediaPackage, report when something is wrong, and take automatic actions when appropriate:

- *Amazon CloudWatch* monitors your AWS resources and the applications that you run on AWS in real-time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track CPU usage or other metrics of your Amazon EC2 instances and automatically launch new instances when needed. For more information, see the Amazon CloudWatch User Guide.
- *Amazon CloudWatch Events* delivers a near real-time stream of system events that describe changes in AWS resources. CloudWatch Events enables automated event-driven computing, as you can write rules that watch for certain events and trigger automated actions in other AWS services when these events happen. For more information, see the Amazon CloudWatch Events User Guide.
- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.
- *AWS Elemental MediaPackage access logs* provide detailed records about requests that are made to a channel. Access logs are useful for many applications. For example, access log information can be useful in security and access audits. For more information, see Access logging.
- *MediaPackage manifest update headers* indicate when the service last updated the manifest and segment sequence in workflows that don't use dynamic ad insertion. MediaPackage includes these custom headers in playback responses. These headers are helpful when troubleshooting issues related to stale manifests. For more information, see Monitoring manifest update time.

Topics

- Logging AWS Elemental MediaPackage API calls with AWS CloudTrail
- Access logging
- Monitoring manifest update time
- Monitoring AWS media services with workflow monitor

Monitoring AWS Elemental MediaPackage with Amazon CloudWatch metrics

You can monitor AWS Elemental MediaPackage using CloudWatch, which collects raw data and processes it into readable, near real-time metrics. These statistics are kept for 15 months, so that you can access historical information and gain a better perspective on how your web application or service is performing. You can also set alarms that watch for certain thresholds, and send notifications or take actions when those thresholds are met. For more information, see the Amazon CloudWatch User Guide.

To view metrics using the MediaPackage console

MediaPackage displays metrics throughout the console.

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. Navigate to the appropriate page to view metrics:
   - For metrics on all channels and endpoints in the AWS Region, go to the **Channels** page.
   - For metrics on a specific channel and all of its endpoints, go to the channel's details page.
   - For metrics on a specific endpoint and its channel, go to the endpoint's details page.
3. (Optional) To refine the metrics view, choose **Open in CloudWatch**.

To view metrics using the CloudWatch console

Metrics are grouped first by the service namespace, and then by the various dimension combinations within each namespace.

1. Sign in to the AWS Management Console and open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. Under **All metrics**, choose the **AWS/MediaPackage** namespace.

4. Choose the metric dimension to view the metrics (for example, choose `channel` to view metrics per channel).

To view metrics using the AWS CLI

At a command prompt, use the following command:

```
aws cloudwatch list-metrics --namespace "AWS/MediaPackage"
```

Topics

AWS Elemental MediaPackage live content metrics

The `AWS/MediaPackage` namespace includes the following metrics for live content. AWS Elemental MediaPackage publishes metrics to CloudWatch every minute, if not sooner.

Metric	Description
`ActiveInput`	Indicates if an input has been used as the source for an endpoint in MediaPackage (it has been active). A value of 1 indicates that the input was active, and a 0 (zero) indicates that it wasn't. Units: None Valid dimension: • Combination of `IngestEndpoint` and `OriginEndpoint`
`EgressBytes`	Number of bytes that MediaPackage successfully sends for each request. If MediaPackage doesn't receive any requests for output in the specified interval, then no data is given. Units: Bytes

Metric	Description
	Valid statistics: • `Average` – Average bytes (`Sum`/`SampleCount`) that AWS Elemental MediaPackage outputs over the configured interval. • `Maximum` – Largest individual output request (in bytes) made to AWS Elemental MediaPackage. • `Minimum` – Smallest individual output request (in bytes) made to AWS Elemental MediaPackage. • `SampleCount` – Number of requests that's used in the statistical calculation. • `Sum` – Total number of bytes that AWS Elemental MediaPackage outputs over the configured interval. Valid dimensions: • `Channel` • Combination of `Channel` and `OriginEndpoint` • `PackagingConfiguration` • No dimension

Metric	Description
EgressRequestCount	Number of content requests that AWS Elemental MediaPackage receives. If MediaPackage doesn't receive any requests for output in the specified interval, then no data is given. Units: Count Valid statistics: • Sum – Total number of output requests that AWS Elemental MediaPackage receives. Valid dimensions: • Channel • Combination of Channel and OriginEndpoint • StatusCodeRange • Combination of Channel and StatusCodeRange • Combination of Channel, OriginEndpoint , and StatusCodeRange • PackagingConfiguration • Combination of PackagingConfiguration and StatusCodeRange • No dimension

Metric	Description
EgressResponseTime	The time that it takes MediaPackage to process each output request. If MediaPackage doesn't receive any requests for output in the specified interval, then no data is given. Units: Milliseconds Valid statistics: • Average – Average amount of time (Sum/SampleCount) that it takes AWS Elemental MediaPackage to process output requests over the configured interval. • Maximum – Longest amount of time (in milliseconds) that it takes AWS Elemental MediaPackage to process an output request and provide a response. • Minimum – Shortest amount of time (in milliseconds) that it takes AWS Elemental MediaPackage to process an output request and provide a response. • SampleCount – Number of requests that's used in the statistical calculation. • Sum – Total amount of time that it takes AWS Elemental MediaPackage to process output requests over the configured interval. Valid dimensions: • Channel • Combination of Channel and OriginEndpoint • PackagingConfiguration

Metric	Description
IngressBytes	Number of bytes of content that AWS Elemental MediaPackage receives for each input request. If MediaPackage doesn't receive any requests for input in the specified interval, then no data is given. Units: Bytes Valid statistics: • Average – Average bytes (Sum/SampleCount) that MediaPackage receives over the configured interval. • Maximum – Largest individual input request (in bytes) made to AWS Elemental MediaPackage. • Minimum – Smallest individual input request (in bytes) made to AWS Elemental MediaPackage. • SampleCount – Number of requests that's used in the statistical calculation. • Sum – Total number of bytes that AWS Elemental MediaPackage receives over the configured interval. Valid dimensions: • Channel • Combination of Channel and IngestEndpoint • No dimension

Metric	Description
IngressResponseTime	The time that it takes MediaPackage to process each input request. If MediaPackage doesn't receive any requests for input in the specified interval, then no data is given. Units: Milliseconds Valid statistics: • Average – Average amount of time (Sum/SampleCount) that it takes MediaPackage to process input requests over the configured interval. • Maximum – Longest amount of time (in milliseconds) that it takes AWS Elemental MediaPackage to process an input request and provide a response. • Minimum – Shortest amount of time (in milliseconds) that it takes AWS Elemental MediaPackage to process an input request and provide a response. • SampleCount – Number of requests that's used in the statistical calculation. • Sum – Total amount of time that it takes MediaPackage to process input requests over the configured interval. Valid dimensions: • Channel • Combination of Channel and IngestEndpoint • No dimension

AWS Elemental MediaPackage live dimensions

You can filter the `AWS/MediaPackage` data using the following dimensions.

Dimension	Description
No Dimension	Metrics are aggregated and shown for all channels, endpoints, or status codes.
`Channel`	Metrics are shown only for the specified channel. Value: The autogenerated GUID of the channel. Can be used alone or with other dimensions: • Alone to show metrics for only the specified channel. • With the `originEndpoint` dimension to show metrics for the specified endpoint that's associated with the specified channel.
`IngestEndpoint`	Metrics are shown only for the specified ingest endpoint on a channel. Value: The autogenerated GUID of the ingest endpoint. Can be used with the following dimensions: • With the `channel` dimension to show metrics for the specified ingest endpoint that's associated with the specified channel. • With the `originEndpoint` dimension to show metrics for the specified ingest endpoint that's associated with the specified endpoint.

Dimension	Description
`OriginEndpoint`	Metrics are shown for the specified channel and endpoint combination. Value: The autogenerated GUID of the endpoint. Must be used with the `channel` dimension.
`StatusCodeRange`	Metrics are shown for the specified status code range. Value: `2xx`, `3xx`, `4xx`, or `5xx`. Can be used alone or with other dimensions: • Alone to show all output requests for the specified status range. • With the `channel` dimension to show output requests for all endpoints that are associated with the specified channel, with the specified status code range. • With the `channel` and `originEndpoint` dimensions to show output requests with a specific status code range on the specified endpoint that's associated with the specified channel.

AWS Elemental MediaPackage VOD content metrics

The `AWS/MediaPackage` namespace includes the following metrics for video on demand (VOD) content. AWS Elemental MediaPackage publishes metrics to CloudWatch every minute, if not sooner.

Metric	Description
EgressBytes	Number of bytes that MediaPackage successfully sends for each request. If MediaPackage doesn't receive any requests for output in the specified interval, then no data is given. Units: Bytes Valid statistics: • Average – Average bytes (Sum/SampleCount) that MediaPackage outputs over the configured interval. • Maximum – Largest individual output request (in bytes) made to MediaPackage. • Minimum – Smallest individual output request (in bytes) made to MediaPackage. • SampleCount – Number of requests that's used in the statistical calculation. • Sum – Total number of bytes that MediaPackage outputs over the configured interval. Valid dimensions: • PackagingConfiguration
EgressRequestCount	Number of content requests that MediaPackage receives. If MediaPackage doesn't receive any requests for output in the specified interval, then no data is given. Units: Count Valid statistics: • Sum – Total number of output requests that MediaPackage receives.

Metric	Description
	Valid dimensions: • PackagingConfiguration • Combination of PackagingConfiguration and StatusCodeRange
EgressResponseTime	The time that it takes MediaPackage to process each output request. If MediaPackage doesn't receive any requests for output in the specified interval, then no data is given. Units: Milliseconds Valid statistics: • Average – Average amount of time (Sum/SampleCount) that it takes MediaPackage to process output requests over the configured interval. • Maximum – Longest amount of time (in milliseconds) that it takes MediaPackage to process an output request and provide a response. • Minimum – Shortest amount of time (in milliseconds) that it takes MediaPackage to process an output request and provide a response. • SampleCount – Number of requests that's used in the statistical calculation. • Sum – Total amount of time that it takes MediaPackage to process output requests over the configured interval. Valid dimensions: • PackagingConfiguration

AWS Elemental MediaPackage VOD dimensions

You can filter the AWS/MediaPackage data using the following dimensions.

Dimension	Description
No Dimension	Metrics are aggregated and shown for all packaging configurations and status codes.
PackagingConfiguration	Metrics are shown only for the specified packaging configuration. Value: The autogenerated GUID of the configuration. Can be used alone or with other dimensions: • Alone to show metrics for only the specified configuration. • With the statusCodeRange dimension to show metrics for the specified configuration that's associated with the specified status code.
StatusCodeRange	Metrics are shown for the specified status code range. Value: 2xx, 3xx, 4xx, or 5xx. Can be used alone or with other dimensions: • Alone to show all output requests for the specified status range. • With the channel dimension to show output requests for all endpoints that are associated with the specified channel, with the specified status code range. • With the channel and originEndpoint dimensions to show output requests with a

Dimension	Description
	specific status code range on the specified endpoint that's associated with the specified channel.

Monitoring AWS Elemental MediaPackage with CloudWatch Events

Amazon CloudWatch Events enables you to automate your AWS services and respond automatically to system events such as application availability issues or error conditions. AWS services deliver events to CloudWatch Events in near real-time. You can write simple rules to indicate which events are of interest to you, and what automated actions to take when an event matches a rule. The actions that can be automatically triggered include the following:

- Invoking an AWS Lambda function
- Invoking AWS Systems Manager Run Command
- Relaying the event to Amazon Kinesis Data Streams
- Activating an AWS Step Functions state machine

An example of using CloudWatch Events with MediaPackage is notifying an Amazon SNS topic if you reach the maximum stream input. MediaPackage emits events on a best effort basis.

For more information about creating rules in CloudWatch Events, see the Amazon CloudWatch Events User Guide.

For a list of events that MediaPackage emits, see AWS Elemental MediaPackage events.

AWS Elemental MediaPackage events

AWS Elemental MediaPackage integrates with Amazon CloudWatch Events to notify you of certain events that affect your channels and endpoints. Each event is represented in JSON (JavaScript Object Notation) and contains the event name, the date and time when the event occurred, the channel or endpoint affected, and more. MediaPackage emits events on a best effort basis. You can use CloudWatch Events to collect these events and set up rules that route them to one or

more *targets* such as AWS Lambda functions, Amazon SNS topics, Amazon SQS queues, streams in Amazon Kinesis Data Streams, or built-in targets.

For more information about using CloudWatch Events with other kinds of events, see the Amazon CloudWatch Events User Guide.

The following topics describe the CloudWatch Events that MediaPackage creates.

Event types

- Input notification events
- Key provider notification events
- Harvest job notification events

Input notification events

You get input notification events for live and video on demand (VOD) content. These events notify you when something happens with MediaPackage ingest. These are the input notification events you might receive:

- Maximum input streams exceeded
- Input switch
- VOD ingest status change
- VOD playback readiness

The following sections describe each of these events.

Maximum Input Streams Exceeded Event

For live content, a channel in MediaPackage exceeds the quota for the number of input streams. For information about quotas, see Quotas in AWS Elemental MediaPackage.

Example

```
{
   "id": "7bf73129-1428-4cd3-a780-95db273d1602",
   "detail-type": "MediaPackage Input Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2015-11-11T21:29:54Z",
```

```
   "region": "us-west-2",
   "resources":[
      "arn:aws:mediapackage:us-
west-2:aws_account_id:channels/262ff182d46d4b399fcabea1364df682"
   ],
   "detail":{
      "event": "MaxIngestStreamsError",
      "message": "Parent Manifest [%s] has [23] streams, more than [20] allowed:
 (index_1.m3u8,index_2.m3u8,index_3.m3u8,index_4.m3u8,index_5.m3u8,index_6.m3u8,index_7.m3u8
   }
}
```

Input Switch Event

For live content, MediaPackage switches inputs for one of your endpoints.

One event is sent in a five-minute period. If the input switches multiple times in five minutes (for example, if MediaPackage switches to one input, then back to the other), you receive only one event.

For information about input redundancy and what causes inputs to switch, see Live input redundancy AWS Elemental MediaPackage processing flow.

Example

```
{
   "id": "8f9b8e72-0b31-e883-f19c-aec84742f3ce",
   "detail-type": "MediaPackage Input Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2018-07-16T17:29:36Z",
   "region": "us-east-1",
   "resources":[
      "arn:aws:mediapackage:us-
east-1:aws_account_id:origin_endpoints/82d6b9bc04cb4612b487963d6c8d0f1a"
   ],
   "detail":{
      "event": "InputSwitchEvent",
      "message": "Origin endpoint experienced an Input Switch Event",
      "EventDetails": {
        "Channel": "channel name",
        "PreviousIngestEndpoint": "endpoint uuid before input switch",
        "CurrentIngestEndpoint": "endpoint uuid after input switch",
      }
```

```
   }
}
```

VOD Ingest Status Event

For video on demand (VOD) content, an asset in MediaPackage changes ingest status. You get notifications for the following events:

- `IngestStart`
- `IngestError`
- `IngestComplete`

Example

```
{
   "id": "8f9b8e72-0b31-e883-f19c-aec84742f3ce",
   "detail-type": "MediaPackage Input Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2019-05-03T17:29:36Z",
   "region": "us-west-2",
   "resources":[

      "arn:aws:mediapackage-vod:us-west-2:aws_account_id:assets/asset_name"

   ],
   "detail":{
      "event": "IngestComplete",
      "message": "message text"
   }
}
```

VOD Playback Event

For VOD content, an asset in MediaPackage is available for playback. There is a period of time between when asset ingest is complete, and when the asset can be played back. The event `VodAssetPlayable` means that MediaPackage can now fulfill playback requests for the asset.

You get individual `VodAssetPlayable` events for each packaging configuration in your packaging group. For example, if your packaging group contains one DASH and one HLS packaging configuration, you receive two `VodAssetPlayable` events—one for your DASH packaging configuration, and one for your HLS packaging configuration.

Example

```
{
   "id": "81e896e4-d9e5-ec79-f82a-b4cf3246c567",
   "detail-type": "MediaPackage Input Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2019-11-03T21:46:00Z",
   "region": "us-west-2",
   "resources":[
      "arn:aws:mediapackage-vod:us-west-2:aws_account_id:assets/asset_id",
      "arn:aws:mediapackage-vod:us-
west-2:aws_account_id:packaging_configuration/packaging_configuration_id"
   ],
   "detail":{
      "event": "VodAssetPlayable",
      "message": "Asset 'asset_id' is now playable for PackagingConfiguration
 'packaging_configuration_id'",
      "packaging_configuration_id": "packaging_configuration_id",
      "manifest_urls":[
         "https://555555555555.egress.mediapackage-vod.us-west-2.amazonaws.com/out/
v1/b9cc115bf7f1a/b848dfb116920772aa69ba/a3c74b1cae6a451c/index.m3u8"
      ]
   }
}

{
    "id": "91e896e4-d9e5-ab80-f82a-b4cf3246c568",
    "detail-type": "MediaPackage Input Notification",
    "source": "aws.mediapackage",
    "account": "aws_account_id",
    "time": "2019-11-03T21:47:00Z",
    "region": "us-west-2",
    "resources":[
      "arn:aws:mediapackage-vod:us-west-2:aws_account_id:assets/asset_id",
      "arn:aws:mediapackage-vod:us-
west-2:aws_account_id:packaging_configuration/packaging_configuration_id"
    ],
    "detail":{
      "event": "VodAssetPlayable",
      "message": "Asset 'asset_id' is now playable for PackagingConfiguration
 'packaging_configuration_id'",
      "packaging_configuration_id": "packaging_configuration_id",
      "manifest_urls":[
```

```
          "https://111122223333.egress.mediapackage-vod.us-west-2.amazonaws.com/out/
v1/1234567890abc/021345abcdef6789012345/abcdef0123456789/index.mpd"
      ]
    }
}
```

Key provider notification events

You get key provider notification events when you're using content encryption on an endpoint and MediaPackage can't reach the key provider. For information about DRM and encryption, see https://docs.aws.amazon.com/speke/latest/documentation/.

Example Live key provider notification event

```
{
   "id": "7bf73129-1428-4cd3-a780-98ds273d1602",
   "detail-type": "MediaPackage Key Provider Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2015-11-11T21:29:54Z",
   "region": "us-west-2",
   "resources":[
      "arn:aws:mediapackage:us-west-2:aws_account_id:origin_endpoints/endpoint_id"
   ],
   "detail":{
      "event": "KeyProviderError",
      "message": "message-text"
   }
}
```

Example VOD key provider notification event

```
{
   "id": "7bf73129-1428-4cd3-a780-98ds273d1602",
   "detail-type": "MediaPackage Key Provider Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2015-11-11T21:29:54Z",
   "region": "us-west-2",
   "resources":[
      "arn:aws:mediapackage-vod:us-
west-2:aws_account_id:packaging_configurations/packaging_group_name"
```

```
   ],
   "detail":{
      "event": "KeyProviderError",
      "message": "message-text"
   }
}
```

Harvest job notification events

You get harvest job status events when you export a clip from a live stream to create a live-to-VOD asset. MediaPackage creates notifications when the harvest job succeeds or fails. For information about harvest jobs and live-to-VOD assets, see Creating live-to-VOD assets with AWS Elemental MediaPackage.

Example Successful harvest job event

```
{
   "id": "8f9b8e72-0b31-e883-f19c-aec84742f3ce",
   "detail-type": "MediaPackage HarvestJob Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2019-07-16T17:29:36Z",
   "region": "us-east-1",
   "resources":[
      "arn:aws:mediapackage:us-east-1:aws_account_id:harvest_jobs/harvest_job_id"
   ],
   "detail":{
      "harvest_job": {
          "id": "harvest_job_id",
          "arn": "arn:aws:mediapackage-vod:us-
east-1:aws_account_id:harvest_jobs/harvest_job_id",
          "status": "SUCCEEDED",
          "origin_endpoint_id": "endpoint_id",
          "start_time": "2019-06-26T20:30:00-08:00",
          "end_time": "2019-06-26T21:00:00-08:00",
          "s3_destination": {
              "bucket_name": "s3_bucket_name",
              "manifest_key": "path/and/manifest_name/index.m3u8",
              "role_arn": "arn:aws:iam::aws_account_id:role/S3Access_role",
          },
          "created_at": "2019-06-26T21:03:12-08:00"
      }
   }
```

```
}
```

Example Failed harvest job event

```
{
   "id": "8f9b8e72-0b31-e883-f19c-aec84742f3ce",
   "detail-type": "MediaPackage HarvestJob Notification",
   "source": "aws.mediapackage",
   "account": "aws_account_id",
   "time": "2019-07-16T17:29:36Z",
   "region": "us-east-1",
   "resources":[
      "arn:aws:mediapackage:us-east-1:aws_account_id:harvest_jobs/harvest_job_id"
   ],
   "detail":{
      "harvest_job": {
          "id": "harvest_job_id",
          "arn": "arn:aws:mediapackage-vod:us-
east-1:aws_account_id:harvest_jobs/harvest_job_id",
          "status": "FAILED",
          "origin_endpoint_id": "endpoint_id",
          "start_time": "2019-06-26T20:30:00-08:00",
          "end_time": "2019-06-26T21:00:00-08:00",
          "s3_destination": {
              "bucket_name": "s3_bucket_name",
              "manifest_key": "path/and/manifest_name/index.m3u8",
              "role_arn": "arn:aws:iam::aws_account_id:role/S3Access_role",
          },
          "created_at": "2019-06-26T21:03:12-08:00"
      },
      "message": "Message text"
   }
}
```

Creating event notifications

You can use Amazon CloudWatch Events and Amazon Simple Notification Service (Amazon SNS) to notify you of new events. In CloudWatch Events, the rule describes which events you're notified about. In Amazon SNS, the topic describes what kind of notification you receive. This section provides high-level steps for creating a topic and rule for events from AWS Elemental MediaPackage. For detailed information about topics and rules, see the following:

- Create a topic and Subscribe to a topic in the *Amazon Simple Notification Service Developer Guide*
- Getting started with Amazon CloudWatch Events in the *Amazon CloudWatch Events User Guide*

To create notifications of CloudWatch events

1. Access Amazon SNS and create a topic. Give the topic a descriptive name that you will later recognize.
2. Subscribe to the topic that you just created. Choose what kind of notification you want to receive, and where that notification is sent. For example, for email notifications, choose the **Email** protocol and enter the email address to receive notifications for the endpoint.
3. Access CloudWatch Events and create a rule that uses a **Custom event pattern**. In the pattern preview space, enter the following:

```
{
  "source": [
    "aws.mediapackage"
  ],
  "detail-type": [
    "detail-type from event"
  ]
}
```

For `detail-type`, enter the value for the `detail-type` field from the event. You can use the following values for detail-type:

- **`MediaPackage Input Notification`**
- **`MediaPackage Key Provider Notification`**

For information about the event types, see AWS Elemental MediaPackage events.

Example

The following example rule creates notifications for all events on all detail-types.

```
{
  "source": [
    "aws.mediapackage"
  ],
```

```
  "detail-type": [
    "MediaPackage Input Notification",
    "MediaPackage Key Provider Notification",
    "MediaPackage HarvestJob Notification"
  ]
}
```

4. Add a target to the rule that you just created. Choose **SNS topic**, and then choose the topic that you created in step 1.
5. Configure the details of the rule, and give it a descriptive name. To start using the rule, make sure it's enabled, and then save it.

Logging AWS Elemental MediaPackage API calls with AWS CloudTrail

Logging is available with only live workflows in AWS Elemental MediaPackage.

MediaPackage is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in MediaPackage. CloudTrail captures all API calls for MediaPackage as events. These include calls from the MediaPackage console and code calls to the MediaPackage API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for MediaPackage. 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 MediaPackage, 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.

AWS Elemental MediaPackage information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Elemental MediaPackage, 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 account. For more information, see Viewing events with CloudTrail event history.

For an ongoing record of events in your account, including events for MediaPackage, 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 console, the trail applies to all AWS Regions. The trail logs events from all AWS

Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple regions and Receiving CloudTrail log files from multiple accounts

All MediaPackage actions are logged by CloudTrail and are documented in the AWS Elemental MediaPackage API Reference. For example, calls to the `CreateChannel`, `CreateOriginEndpoint`, and `RotateIngestEndpointCredentials` operations 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 the following:

- Whether the request was made with root user or IAM user credentials
- Whether the request was made with temporary security credentials for a role or federated user
- Whether the request was made by another AWS service

For more information, see the CloudTrail userIdentity element.

Understanding AWS Elemental MediaPackage log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. 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 aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `UpdateChannel` operation:

```
{
    "eventVersion": "1.05",
    "userIdentity": {
```

```
        "type": "AssumedRole",
        "principalId": "ABCDEFGHIJKL123456789",
        "arn": "arn:aws:sts::444455556666:assumed-role/Admin/testUser",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-12-18T00:50:58Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "ABCDEFGHIJKL123456789",
                "arn": "arn:aws:iam::444455556666:role/Admin",
                "accountId": "444455556666",
                "userName": "Admin"
            }
        }
    },
    "eventTime": "2018-12-18T00:50:59Z",
    "eventSource": "mediapackage.amazonaws.com",
    "eventName": "UpdateChannel",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.17",
    "userAgent": "aws-cli/1.15.71 Python/3.6.5 Darwin/17.7.0 botocore/1.10.70",
    "requestParameters": {
        "description": "updated cloudtrail description",
        "id": "cloudtrail-test"
    },
    "responseElements": {
        "description": "updated cloudtrail description",
        "hlsIngest": {
            "ingestEndpoints": [
                {
                    "username": "***",
                    "url": "https://mediapackage.us-west-2.amazonaws.com/in/
v2/8d0ca97840d94b18b37ad292c131bcad/8d0ca97840d94b18b37ad292c131bcad/channel",
                    "password": "***",
                    "id": "8d0ca97840d94b18b37ad292c131bcad"
                },
                {
                    "username": "***",
                    "url": "https://mediapackage.us-west-2.amazonaws.com/in/
v2/8d0ca97840d94b18b37ad292c131bcad/9c17f979598543b9be24345d63b3ad30/channel",
```

```
                    "password": "***",
                    "id": "9c17f979598543b9be24345d63b3ad30"
                }
            ]
        },
        "id": "cloudtrail-test",
        "arn": "arn:aws:mediapackage:us-
west-2:444455556666:channels/8d0ca97840d94b18b37ad292c131bcad"
    },
    "requestID": "fc158262-025e-11e9-8360-6bff705fbba5",
    "eventID": "e9016b49-9a0a-4256-b684-eed9bd9073ab",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
}
```

Access logging

MediaPackage provides access logs that capture detailed information about requests sent to your MediaPackage channel or packaging group. MediaPackage generates *ingress access logs* for requests sent to the channel's input endpoints, and *egress access logs* for requests sent to your channel's endpoints or packaging group's assets. Each log contains information such as the time the request was received, the client's IP address, latencies, request paths, and server responses. You can use these access logs to analyze service performance and troubleshoot issues. They can also help you learn about your customer base and understand your MediaPackage bill.

Access logging is an optional feature of MediaPackage that's disabled by default. After you enable access logging, MediaPackage captures the logs and saves them to the CloudWatch log group that you specify when you create or manage access logging. Typical CloudWatch Logs charges apply.

Topics

Permissions to publish access logs to CloudWatch

When you enable access logging, MediaPackage creates an IAM service-linked role, `AWSServiceRoleForMediaPackage`, in your AWS account. This role allows MediaPackage to publish access logs to CloudWatch. For information about how MediaPackage uses service-linked roles, see Using Service-Linked Roles for MediaPackage.

Enable access logging

You can enable access logs using the AWS Management Console or the AWS CLI.

To enable access logs for an existing channel using the console

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. Select your channel.
3. In the **Configure Access Logs** section, do the following:
   a. Choose **Enable ingress access logs** or **Enable egress access logs**, or both.
   b. You can specify a custom CloudWatch **Log group name**. If left blank, the default group is used.

To enable access logs for an existing packaging group using the console

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.
2. Select **Packaging groups** from the navigation section.
3. Choose your packaging group.
   a. Select **Edit** in the navigation bar.
   b. In the **Access logging** section, select **Enable egress access logs**.
   c. You can specify a custom CloudWatch **Log group name**. If left blank, the default group is used.
4. Choose **Save changes**.

To enable access logs for a channel using the AWS CLI

Use the configure-logs command with the `--ingress-access-logs` parameter, `--egress-access-logs` parameter, or both, to enable access logging. You can include a CloudWatch log

group name for the `--ingress-access-logs` and `--egress-access-logs` parameters. If you don't specify a log group name, then the MediaPackage default log group is used. For ingress logs, the default log group is `/aws/MediaPackage/IngressAccessLogs`, and for egress logs the default log group is `/aws/MediaPackage/EgressAccessLogs`.

Use the following command to enable both ingress and access logs using the default log groups:

```
aws mediapackage configure-logs --id channel-name --ingress-access-logs {} --egress-access-logs {}
```

This command has no return value.

To enable access logs for a packaging group using the AWS CLI

Use the configure-logs command with the `--egress-access-logs` parameter to enable access logging. You can include a CloudWatch log group name for the `--egress-access-logs` parameter. If you don't specify a log group name, then the MediaPackage default log group is used. For ingress logs, the default log group is `/aws/MediaPackage/IngressAccessLogs`, and for egress logs the default log group is `/aws/MediaPackage/EgressAccessLogs`.

Use the following command to enable egress access logs using the default log groups:

```
aws mediapackage configure-logs --id package-name --egress-access-logs {}
```

This command has no return value.

Disable access logging

You can disable access logs for your MediaPackage channel or packaging group at any time.

To disable access logging using the console

1. Open the MediaPackage console at https://console.aws.amazon.com/mediapackage/.

   Select your channel or package group.
2. Choose **Edit**.
3. In the **Access logging** section, deselect **Ingress access logging**, **Egress access logging**, or both.
4. Choose **Save changes**.

To disable access logging for a channel using the AWS CLI

Use the `configure-logs` command to disable access logging. If one or more of the access log parameters aren't declared with the `configure-logs` command, then the corresponding access logs are disabled. For example, in the following command egress access logs are enabled for a channel, and ingress access logs are disabled:

```
aws mediapackage configure-logs --id channel-name --egress-access-logs {}
```

This command has no return value.

To disable access logging for a packaging group using the AWS CLI

Use the `configure-logs` command to disable access logging. If one or more of the access log parameters aren't declared with the `configure-logs` command, then the corresponding access logs are disabled. For example, in the following command `configure-logs` doesn't include `--egress-access-logs` so egress logs are disabled:

```
aws mediapackage configure-logs --id package-group-name
```

This command has no return value.

Access log format

The access log files consist of a sequence of JSON-formatted log records, where each log record represents one request. The order of the fields within the log can vary. The following is an example channel egress access log:

```
{
    "timestamp": "2020-07-13T18:59:56.293656Z",
    "clientIp": "192.0.2.0/24",
    "processingTime": 0.445,
    "statusCode": "200",
    "receivedBytes": 468,
    "sentBytes": 2587370,
    "method": "GET",
    "request": "https://aaabbbcccdddee.mediapackage.us-east-1.amazonaws.com:443/out/
v1/75ee4f20e5df43e5821e5cb17ea19238/hls_7_145095.ts?m=1538005779",
    "protocol": "HTTP/1.1",
    "userAgent": "sabr/3.0 Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US)
 AppleWebKit/528.18 (KHTML, like Gecko) Version/4.0 Safari/528.17",
    "account": "111122223333",
    "channelId": "my_channel",
```

```
    "channelArn": "arn:aws:mediapackage:us-west-2:111122223333:channels/
ExampleChannelID",
    "domainName": "aaabbbcccdddee.mediapackage.us-east-1.amazonaws.com",
    "requestId": "aaaAAA111bbbBBB222cccCCC333dddDDD",
    "endpointId": "my_endpoint",
    "endpointArn": "arn:aws:mediapackage:us-west-2:111122223333:origin_endpoints/
ExampleEndpointID"
}
```

The following list describes the log record fields, in order:

timestamp

The time of day when the request was received. The value is `ISO-8601` date time and is based on the system clock of the host that served the request.

clientIp

The IP address of the requesting client.

processingTime

The number of seconds that MediaPackage spent processing your request. This value is measured from the time the last byte of your request was received until the time the first byte of the response was sent.

statusCode

The numeric HTTP status code of the response.

receivedBytes

The number of bytes in the request body that the MediaPackage server receives.

sentBytes

The number of bytes in the response body that the MediaPackage server sends. This value often is the same as the value of the `Content-Length` header that's included with server responses.

method

The HTTP request method that was used for the request: DELETE, GET, HEAD, OPTIONS, PATCH, POST, or PUT.

request

The request URL.

protocol

The type of protocol used for the request, such as HTTP.

userAgent

A user-agent string that identifies the client that originated the request, enclosed in double quotes. The string consists of one or more product identifiers, product/version. If the string is longer than 8 KB, it is truncated.

account

The AWS account ID of the account that was used to make the request.

channelId

The ID of the channel that received the request.

channelArn

The Amazon Resource Name (ARN) of the channel that received the request.

domainName

The server name indication domain provided by the client during the TLS handshake, enclosed in double quotes. This value is set to - if the client doesn't support SNI or the domain doesn't match a certificate and the default certificate is presented to the client.

requestId

A string that's generated by MediaPackage to uniquely identify each request.

endpointId

The ID of the endpoint that received the request.

endpointArn

The Amazon Resource Name (ARN) of the endpoint that received the request.

The order of the fields in the log can vary.

Read the access logs

MediaPackage writes the access logs to Amazon CloudWatch Logs. Typical CloudWatch Logs charges apply. Use CloudWatch Logs Insights to read the access logs. For information on how to

use CloudWatch Logs Insights, see Analyzing Log Data with CloudWatch Logs Insights in the *AWS CloudWatch Logs User Guide*.

> **Note**
>
> The access logs can take a few minutes to appear in CloudWatch. If you don't see the logs, wait a few minutes and try again.

Examples

This section includes example queries that you can use to read MediaPackage debug log data.

Example View the HTTP status code responses for a channel.

Use this query to view the responses by HTTP status code for a channel. You can use this to view HTTP error code responses to help you to troubleshoot issues.

```
fields @timestamp, @message
| filter channelId like 'my-channel'
| stats count() by statusCode
```

Example Get the number of requests per endpoint on a channel.

```
fields @timestamp, @message
| filter channelId like 'my-channel'
| stats count() by endpointId
```

Example View status codes per asset.

```
fields @timestamp, @message
| filter assetArnlike 'my-asset-id'
| stats count() by statusCode
```

Example Get the P99 response times for a packaging configuration over time

```
fields @timestamp, @message
| filter packagingConfigArn like 'my-dash-config'
| stats pct(processingTime, 99) by bin(5m)
```

Monitoring manifest update time

AWS Elemental MediaPackage playback responses include the following custom headers that indicate when MediaPackage last modified the manifest in non-dynamic ad insertion workflows. These headers are helpful when troubleshooting issues related to stale manifests.

X-MediaPackage-Manifest-Last-Sequence

This is the highest segment sequence number in the manifest.

- For DASH, this is the highest segment number in the lowest rendition of the manifest.
- For HLS and CMAF, this is the highest segment number in the media playlist.
- For MSS, this is the highest segment number in the manifest.

See the following section for manifest examples.

X-MediaPackage-Manifest-Last-Updated

The epoch timestamp in milliseconds when MediaPackage generates the segment referred to in `X-MediaPackage-Manifest-Last-Sequence`.

Manifest examples

DASH manifest examples

For both compact and full DASH manifests, MediaPackage determines the `X-MediaPackage-Manifest-Last-Sequence` value from the highest segment number in the lowest rendition of the manifest. The service calculates the `X-MediaPackage-Manifest-Last-Updated` value based on when it generates the segment referred to in `X-MediaPackage-Manifest-Last-Sequence`.

Number with duration - compact manifest

The following is an example of a compact DASH manifest that uses the number with duration template. MediaPackage determines the `X-MediaPackage-Manifest-Last-Sequence` value from the highest segment number in the lowest rendition in the manifest. For example, in the following manifest, the highest segment number is `index_video_5_0_175232.mp4`, so the value

of X-MediaPackage-Manifest-Last-Sequence is 175232. See duration Attribute in the SegmentTemplate for information about how MediaPackage calculates the sequence $Number$ value. The value of X-MediaPackage-Manifest-Last-Updated is the epoch timestamp in milliseconds when MediaPackage generates the segment referred to in X-MediaPackage-Manifest-Last-Sequence.

```
<?xml version="1.0" encoding="utf-8"?>
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="urn:mpeg:dash:schema:mpd:2011"
 xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 http://standards.iso.org/
ittf/PubliclyAvailableStandards/MPEG-DASH_schema_files/DASH-MPD.xsd" id="201"
 type="dynamic" publishTime="2021-09-08T21:01:38" minimumUpdatePeriod="PT0S"
 availabilityStartTime="2018-11-16T19:08:30Z+00:00" minBufferTime="PT0S"
 suggestedPresentationDelay="PT0.000S" timeShiftBufferDepth="PT116.533S"
 profiles="urn:mpeg:dash:profile:isoff-live:2011">
  <Period start="PT0.000S" id="1">
    <AdaptationSet mimeType="video/mp4" segmentAlignment="true"
 subsegmentAlignment="true" startWithSAP="1" subsegmentStartsWithSAP="1"
 bitstreamSwitching="true">
       <SegmentTemplate timescale="30000" media="index_video_$RepresentationID$_0_
$Number$.mp4?m=1543947824" initialization="index_video_$RepresentationID$_0_init.mp4?
m=1543947824" startNumber="175032" duration="90000"/>
       <Representation id="1" width="640" height="360" frameRate="30/1"
 bandwidth="749952" codecs="avc1.640029"/>
       <Representation id="2" width="854" height="480" frameRate="30/1"
 bandwidth="1000000" codecs="avc1.640029"/>
       <Representation id="3" width="1280" height="720" frameRate="30/1"
 bandwidth="2499968" codecs="avc1.640029"/>
    </AdaptationSet>
</Period>
</MPD>
```

Number with timeline - compact manifest

The following is an example of a compact DASH manifest that uses the number with timeline template. MediaPackage determines the X-MediaPackage-Manifest-Last-Sequence value from the highest segment number in the lowest rendition in the manifest. For example, in the following manifest, the highest segment number is index_video_1_0_7.mp4, so the value of X-MediaPackage-Manifest-Last-Sequence is 7. The value of X-MediaPackage-Manifest-Last-Updated is the is the epoch timestamp in milliseconds when MediaPackage generates the segment referred to in X-MediaPackage-Manifest-Last-Sequence.

```
<?xml version="1.0" encoding="utf-8"?>
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="urn:mpeg:dash:schema:mpd:2011"
 xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 http://standards.iso.org/
ittf/PubliclyAvailableStandards/MPEG-DASH_schema_files/DASH-MPD.xsd" id="201"
 type="static" mediaPresentationDuration="PT72.458S" minBufferTime="PT0S"
 profiles="urn:mpeg:dash:profile:isoff-main:2011">
  <Period start="PT0.000S" id="1" duration="PT74.758S">
    <AdaptationSet mimeType="video/mp4" segmentAlignment="true" startWithSAP="1"
 subsegmentAlignment="true" subsegmentStartsWithSAP="1" bitstreamSwitching="true">
      <SegmentTemplate timescale="48000" media="index_video_$RepresentationID$_0_
$Number$.mp4?m=1621616401" initialization="index_video_$RepresentationID$_0_init.mp4?
m=1621616401" startNumber="1" presentationTimeOffset="108800">
        <SegmentTimeline>
          <S t="110400" d="540000" r="5"/>
          <S t="3350400" d="238000"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="1" width="640" height="480" frameRate="24/1"
 bandwidth="5000000" codecs="avc1.4D401E"/>
    </AdaptationSet>
    <AdaptationSet mimeType="audio/mp4" segmentAlignment="0" lang="eng">
      <Label>eng</Label>
      <SegmentTemplate timescale="48000" media="index_audio_$RepresentationID$_0_
$Number$.mp4?m=1621616401" initialization="index_audio_$RepresentationID$_0_init.mp4?
m=1621616401" startNumber="1" presentationTimeOffset="108800">
        <SegmentTimeline>
          <S t="108800" d="541696"/>
          <S t="650496" d="540672"/>
          <S t="1191168" d="539648" r="1"/>
          <S t="2270464" d="540672"/>
          <S t="2811136" d="539648"/>
          <S t="3350784" d="236544"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="2" bandwidth="192000" audioSamplingRate="48000"
 codecs="mp4a.40.2">
        <AudioChannelConfiguration
 schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011" value="2"></
AudioChannelConfiguration>
      </Representation>
    </AdaptationSet>
```

```
    <SupplementalProperty schemeIdUri="urn:scte:dash:utc-time"
 value="2021-05-21T16:59:47.450Z"></SupplementalProperty>
  </Period>
</MPD>
```

Number with timeline - compact manifest

The following is an example of a compact DASH manifest that uses the number with duration template. MediaPackage determines the `X-MediaPackage-Manifest-Last-Sequence` value from the highest segment number in the lowest rendition in the manifest. For example, in the following manifest, the highest segment number is `index_video_1_0_1675200.mp4`, so the value of `X-MediaPackage-Manifest-Last-Sequence` is `1675200`. See `media` Attribute in `SegmentTemplate` for information about how MediaPackage calculates the sequence number. The value of `X-MediaPackage-Manifest-Last-Updated` is the is the epoch timestamp in milliseconds when MediaPackage generates the segment referred to in `X-MediaPackage-Manifest-Last-Sequence`.

```
<?xml version="1.0" encoding="utf-8"?>
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="urn:mpeg:dash:schema:mpd:2011"
 xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 http://standards.iso.org/
ittf/PubliclyAvailableStandards/MPEG-DASH_schema_files/DASH-MPD.xsd" id="201"
 type="static" mediaPresentationDuration="PT72.458S" minBufferTime="PT0S"
 profiles="urn:mpeg:dash:profile:isoff-main:2011">
  <Period start="PT0.000S" id="1" duration="PT74.758S">
    <AdaptationSet mimeType="video/mp4" segmentAlignment="true" startWithSAP="1"
 subsegmentAlignment="true" subsegmentStartsWithSAP="1" bitstreamSwitching="true">
      <SegmentTemplate timescale="48000" media="index_video_$RepresentationID$_0_
$Time$.mp4?m=1621616401" initialization="index_video_$RepresentationID$_0_init.mp4?
m=1621616401" startNumber="1" presentationTimeOffset="108800">
        <SegmentTimeline>
          <S t="55200" d="270000" r="5"/>
          <S t="1675200" d="119000"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="1" width="640" height="480" frameRate="24/1"
 bandwidth="5000000" codecs="avc1.4D401E"/>
    </AdaptationSet>
    <AdaptationSet mimeType="audio/mp4" segmentAlignment="0" lang="eng">
      <Label>eng</Label>
```

```
      <SegmentTemplate timescale="48000" media="index_audio_$RepresentationID$_0_
$Time$.mp4?m=1621616401" initialization="index_audio_$RepresentationID$_0_init.mp4?
m=1621616401" startNumber="1" presentationTimeOffset="108800">
        <SegmentTimeline>
          <S t="108800" d="541696"/>
          <S t="650496" d="540672"/>
          <S t="1191168" d="539648" r="1"/>
          <S t="2270464" d="540672"/>
          <S t="2811136" d="539648"/>
          <S t="3350784" d="236544"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="2" bandwidth="192000" audioSamplingRate="48000"
 codecs="mp4a.40.2">
        <AudioChannelConfiguration
 schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011" value="2"></
AudioChannelConfiguration>
      </Representation>
    </AdaptationSet>
    <SupplementalProperty schemeIdUri="urn:scte:dash:utc-time"
 value="2021-05-21T16:59:47.450Z"></SupplementalProperty>
  </Period>
</MPD>
```

HLS manifest

MediaPackage determines the `X-MediaPackage-Manifest-Last-Sequence` value from the last segment in the manifest. For example, in the following manifest `index_1_3.ts` is the highest segment sequence number, so the value of `X-MediaPackage-Manifest-Last-Sequence` is `3`. The value of `X-MediaPackage-Manifest-Last-Updated` corresponds to the epoch timestamp in milliseconds when MediaPackage generates the last segment in the manifest.

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-TARGETDURATION:8
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:7.500,
index_1_0.ts?m=1583172400
#EXTINF:7.500,
index_1_1.ts?m=1583172400
#EXTINF:7.500,
index_1_2.ts?m=1583172400
```

```
#EXTINF:7.500,
index_1_3.ts?m=1583172400
#EXT-X-ENDLIST
```

CMAF manifest

Similar to HLS, MediaPackage determines the `X-MediaPackage-Manifest-Last-Sequence` value from the last segment in the manifest. For example, in the following manifest `../cmafseg_video_1_10.mp4?m=1621616399` is the highest segment sequence number, so the value of `X-MediaPackage-Manifest-Last-Sequence` is `10`. The value of `X-MediaPackage-Manifest-Last-Updated` corresponds to the epoch timestamp in milliseconds when MediaPackage generates the last segment in the manifest.

```
#EXTM3U
#EXT-X-VERSION:6
#EXT-X-INDEPENDENT-SEGMENTS
#EXT-X-TARGETDURATION:12
#EXT-X-MEDIA-SEQUENCE:1
#EXT-X-MAP:URI="../cmafseg_video_1_track_1098178399_csid_aaa_2_init.mp4"
#EXTINF:11.250,
../cmafseg_video_1_1.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_2.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_3.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_4.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_5.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_6.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_7.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_8.mp4?m=1621616399
#EXTINF:11.250,
../cmafseg_video_1_9.mp4?m=1621616399
#EXTINF:0.542,
../cmafseg_video_1_10.mp4?m=1621616399
#EXT-X-ENDLIST
```

MSS manifest

MediaPackage determines the `X-MediaPackage-Manifest-Last-Sequence` value from the highest segment in the manifest, as indicated by `Fragments(a_2_0={start time})`. For example, in the following manifest `Fragments(a_2_0=380533333)` is the highest sequence number, so the value of `X-MediaPackage-Manifest-Last-Sequence` is `380333333`. The value of `X-MediaPackage-Manifest-Last-Updated` corresponds to the epoch timestamp in milliseconds when MediaPackage generates the last segment in the manifest.

```
<SmoothStreamingMedia MajorVersion="2" MinorVersion="2" TimeScale="10000000"
 CanSeek="TRUE" CanPause="TRUE" IsLive="TRUE" LookAheadFragmentCount="2"
 DVRWindowLength="3000000000" Duration="0">
  <CustomAttributes>
    <Attribute Name="ProducerReferenceTime" Value="2017-06-14T22:07:01.967Z"/>
  </CustomAttributes>
  <StreamIndex Type="video" Name="video" Subtype="" Chunks="3" TimeScale="10000000"
 Url="Events(203_0)/QualityLevels({bitrate})/Fragments(v={start time})"
 QualityLevels="1">
    <QualityLevel Index="0" Bitrate="4000000"
 CodecPrivateData="00000001274D401F924602802DD8088000003008000001E7220007A120000895477BDC07C22
 FourCC="H264" MaxWidth="1280" MaxHeight="720"/>
    <c d="120000000" t="20333333"/>
    <c d="120000000"/>
    <c d="120000000"/>
  </StreamIndex>
  <StreamIndex Type="audio" Name="fra_1" Language="fra" Subtype=""
 Chunks="3" TimeScale="10000000" Url="Events(203_0)/QualityLevels({bitrate})/
Fragments(a_2_0={start time})">
    <QualityLevel Index="0" Bitrate="128460" CodecPrivateData="1190" FourCC="AACL"
 AudioTag="255" Channels="2" SamplingRate="48000" BitsPerSample="16" PacketSize="4"/>
    <c d="120533333" t="20000000"/>
    <c d="119893333"/>
    <c d="120106667"/>
  </StreamIndex>
</SmoothStreamingMedia>
```

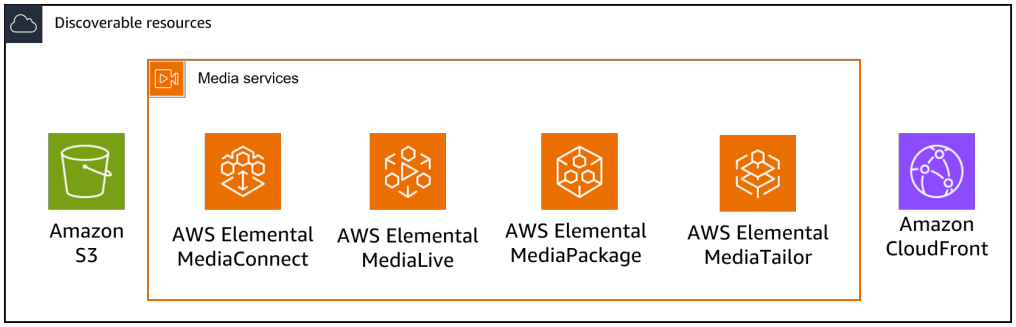
Monitoring AWS media services with workflow monitor

Workflow monitor is a tool for the discovery, visualization, and monitoring of AWS media workflows. Workflow monitor is available in the AWS console and API. You can use workflow monitor to discover and create visual mappings of your workflow's resources, called *signal maps*.

You can create and manage Amazon CloudWatch alarm and Amazon EventBridge rule templates to monitor the mapped resources. The monitoring templates you create are transformed into deployable AWS CloudFormation templates to allow repeatability. AWS recommended alarm templates provide predefined best-practice monitoring.

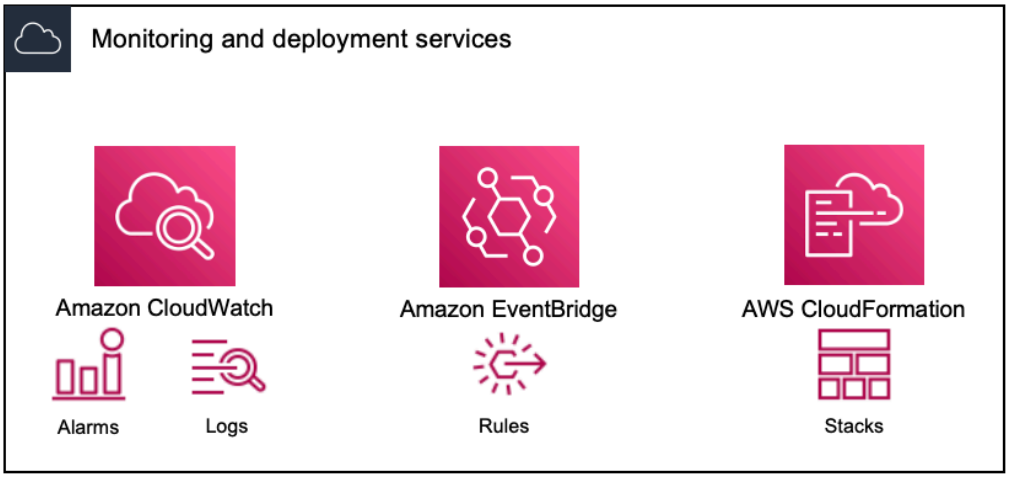
Discover

Utilize signal maps to automatically discover interconnected AWS resources associated with your media workflow. Discovery can begin at any supported service resource and creates an end-to-end mapping of the workflow. Signal maps can be used as stand-alone visualization tools or enhanced with monitoring templates.



Monitor

You can create custom CloudWatch alarm and EventBridge rule templates to monitor the health and status of your media workflows. Best practice alarm templates are available to import into your workflow monitor environment. You can use the best practice alarm templates as they are, or edit them to better fit your workflow. Any templates you create are transformed into CloudFormation templates for repeatable deployment.



Note

There is no direct cost for using workflow monitor. However, there are costs associated with the resources created and used to monitor your workflow.
When monitoring is deployed, Amazon CloudWatch and Amazon EventBridge resources are created. When using the AWS Management Console, prior to deploying monitoring to a signal map, you will be notified of how many resources will be created. For more information about pricing, see: CloudWatch pricing and EventBridge pricing.
Workflow monitor uses AWS CloudFormation templates to deploy the CloudWatch and EventBridge resources. These templates are stored in a standard class Amazon Simple Storage Service bucket that is created on your behalf, by workflow monitor, during the deployment process and will incur object storage and recall charges. For more information about pricing, see: Amazon S3 pricing.
Previews generated in the workflow monitor signal map for AWS Elemental MediaPackage channels are delivered from the MediaPackage Origin Endpoint and will incur Data Transfer Out charges. For pricing, see: MediaPackage pricing.

Components of workflow monitor

Workflow monitor has four major components:

- CloudWatch alarm templates - Define the conditions you would like to monitor using CloudWatch. You can create your own alarm templates, or import predefined templates created by AWS. For more information, see: CloudWatch alarm groups and templates for monitoring your AWS media workflow
- EventBridge rule templates - Define how EventBridge sends notifications when an alarm is triggered. For more information, see: EventBridge rule groups and templates for monitoring your AWS media workflow
- Signal maps - Use an automated process to create AWS Elemental workflow maps using existing AWS resources. The signal maps can be used to discover resources in your workflow and deploy monitoring to those resources. For more information, see: Workflow monitor signal maps
- Overview - The overview page allows you to directly monitor the status of multiple signal maps from one location. Review metrics, logs, and alarms for your workflows. For more information, see: Workflow monitor overview

Supported services

Workflow monitor supports automatic discovery and signal mapping of resources associated with the following services:

- AWS Elemental MediaConnect
- AWS Elemental MediaLive
- AWS Elemental MediaPackage
- AWS Elemental MediaTailor
- Amazon S3
- Amazon CloudFront

Topics

Configuring workflow monitor to monitor AWS media services

To setup workflow monitor for the first time; you create the alarm and event templates, and discover signal maps that are used to monitor your media workflows. The following guide contains the steps necessary to setup both Administrator and Operator level IAM roles, create workflow monitor resources, and deploy monitoring to your workflows.

Topics

Getting started with workflow monitor

The following steps provide a basic overview of using workflow monitor for the first time.

1. Setup workflow monitor IAM permissions for administrator and operator level roles: Workflow monitor IAM policies

2. Build alarm templates or import predefined templates created by AWS: CloudWatch alarms
3. Build notification events that will be delivered by EventBridge: EventBridge rules
4. Discover signal maps using your existing AWS Elemental resources: Signal maps
5. Attach the alarm templates and notification rules to your signal map: Attaching templates
6. Deploy the templates to begin monitoring the signal map: Deploying monitoring templates
7. Monitor and review your workflow monitor resources using the overview section of the AWS console: Overview

Workflow monitor IAM policies

Workflow monitor interacts with multiple AWS services to create signal maps, build CloudWatch and EventBridge resources, and CloudFormation templates. Because workflow monitor interacts with a wide range of services, specific AWS Identity and Access Management (IAM) policies must be assigned for these services. The following examples indicate the necessary IAM policies for both administrator and operator IAM roles.

Administrator IAM policy

The following example policy is for an administrator-level workflow monitor IAM policy. This role allows for the creation and management of workflow monitor resources and the supported service resources that interact with workflow monitor.

JSON

```
{

  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:List*",
        "cloudwatch:Describe*",
```

```
        "cloudwatch:Get*",
        "cloudwatch:PutAnomalyDetector",
        "cloudwatch:PutMetricData",
        "cloudwatch:PutMetricAlarm",
        "cloudwatch:PutCompositeAlarm",
        "cloudwatch:PutDashboard",
        "cloudwatch:DeleteAlarms",
        "cloudwatch:DeleteAnomalyDetector",
        "cloudwatch:DeleteDashboards",
        "cloudwatch:TagResource",
        "cloudwatch:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:List*",
        "cloudformation:Describe*",
        "cloudformation:CreateStack",
        "cloudformation:UpdateStack",
        "cloudformation:DeleteStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudfront:List*",
        "cloudfront:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeNetworkInterfaces"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": [
        "events:List*",
        "events:Describe*",
        "events:CreateEventBus",
        "events:PutRule",
        "events:PutTargets",
        "events:EnableRule",
        "events:DisableRule",
        "events:DeleteRule",
        "events:RemoveTargets",
        "events:TagResource",
        "events:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:Describe*",
        "logs:Get*",
        "logs:TagLogGroup",
        "logs:TagResource",
        "logs:UntagLogGroup",
        "logs:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediaconnect:List*",
        "mediaconnect:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "medialive:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": [
        "mediapackage:List*",
        "mediapackage:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediapackagev2:List*",
        "mediapackagev2:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediapackage-vod:List*",
        "mediapackage-vod:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediatailor:List*",
        "mediatailor:Describe*",
        "mediatailor:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "resource-groups:ListGroups",
        "resource-groups:GetGroup",
        "resource-groups:GetTags",
        "resource-groups:GetGroupQuery",
        "resource-groups:GetGroupConfiguration",
        "resource-groups:CreateGroup",
        "resource-groups:UngroupResources",
        "resource-groups:GroupResources",
        "resource-groups:DeleteGroup",
        "resource-groups:UpdateGroupQuery",
```

```
        "resource-groups:UpdateGroup",
        "resource-groups:Tag",
        "resource-groups:Untag"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:*"
      ],
      "Resource": "arn:aws:s3:::workflow-monitor-templates*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:TagResource",
        "sns:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "tag:Get*",
        "tag:Describe*",
        "tag:TagResources",
        "tag:UntagResources"
      ],
      "Resource": "*"
    }
  ]
}
```

Operator IAM policy

The following example policy is for an operator-level workflow monitor IAM policy. This role allows for limited and read-only access to the workflow monitor resources and the supported service resources that interact with workflow monitor.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:List*",
        "cloudwatch:Describe*",
        "cloudwatch:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:List*",
        "cloudformation:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudfront:List*",
        "cloudfront:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeNetworkInterfaces"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "events:List*",
        "events:Describe*"
      ],
```

```
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:Describe*",
        "logs:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediaconnect:List*",
        "mediaconnect:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "medialive:List*",
        "medialive:Get*",
        "medialive:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediapackage:List*",
        "mediapackage:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediapackagev2:List*",
        "mediapackagev2:Get*"
      ],
      "Resource": "*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "mediapackage-vod:List*",
        "mediapackage-vod:Describe*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mediatailor:List*",
        "mediatailor:Describe*",
        "mediatailor:Get*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3:::workflow-monitor-templates*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "tag:Get*",
        "tag:Describe*"
      ],
      "Resource": "*"
    }
  ]
}
```

Workflow monitor groups and templates

Before you can deploy workflow monitoring to a signal map, you must create the groups and templates for CloudWatch alarms and EventBridge notifications. The CloudWatch templates define what scenarios and thresholds will be used to trigger the alarms. The EventBridge templates will determine how these alarms are reported to you.

If you only want mappings of your connected resources and do not want to use the monitoring template capabilities of workflow monitor, signal maps can be used without CloudWatch and EventBridge templates. For more information about using signal maps, see: Signal maps

Topics

CloudWatch alarm groups and templates for monitoring your AWS media workflow

Workflow monitor alarms allow you to use existing CloudWatch metrics as the foundation of alarms for your signal maps. You can create an alarm template group to sort and classify the types of alarming that is important to your workflow. Within each alarm template group, you create alarm templates with specific CloudWatch metrics and parameters that you want to monitor. You can create your own alarm templates or import recommended alarm templates created by AWS. After creating an alarm template group and alarm templates within that group, you can attach one or more of these alarm template groups to a signal map.

You must create an alarm template group first. After you have created an alarm template group, you can create your own templates or use recommended templates created by AWS. If you want to create your own alarm templates, continue on this page. For more information about importing recommended templates, see: Recommended templates

This section covers the creation of CloudWatch alarms using workflow monitor. For more information about how the CloudWatch service handles alarms and details of the alarm components, see: Using CloudWatch alarms in the *Amazon CloudWatch User Guide*

Creating alarm template groups

Alarm template groups allow you to sort and classify the types of alarms that are important to your workflow.

To create an alarm template group

1. From the workflow monitor console's navigation pane, select **CloudWatch alarm templates**.
2. Select **Create alarm template group**.
3. Give the alarm template group a unique **Group name** and optional **Description**.
4. Select **Create**, You will be taken to the newly created alarm template group's details page.

Creating alarm templates

You can create alarm templates with the CloudWatch metrics and parameters that you want to monitor.

To create an alarm template

1. From the alarm template group's details page, select **Create alarm template**.
2. Give the alarm template a unique **Template name** and optional **Description**.
3. In the **Choose metric** section:
    1. Select a **Target Resource Type**. The target resource type is a resource for the respective service, such as a channel for MediaLive and MediaPackage or a flow for MediaConnect.
    2. Select a **Metric Name**. This is the CloudWatch metric that acts as the foundation for the alarm. The list of metrics will change depending on the selected **Target Resource Type**.
4. In the **Alarm settings** section:

    > **Note**
    >
    > For more information about how the CloudWatch service handles alarms and details of the alarm components, see: Using CloudWatch alarms in the *Amazon CloudWatch User Guide*

    1. Select the **Statistic**. This is a value such as a **Sum** or an **Average** that will be used to monitor the metric.
    2. Select the **Comparison Operator**. This field references the **Threshold** that you set in the next step.
    3. Set a **Threshold**. This is a numeric value that the **Comparison Operator** uses to determine greater than, less than, or equal to status.
    4. Set a **Period**. This is a time value, in seconds. The **Period** is the length of time that the **Statistic**, **Comparison Operator**, and **Threshold** interact to determine if the alarm gets triggered.
    5. Set the **Datapoints**. This value determines how many datapoints are needed to trigger the alarm.

6. Select how to **Treat Missing Data**. This selection determines how this alarm reacts to missing data.

5. Select **Create** to complete the process.

An example of a completed alarm template could have the following parameters: A MediaConnect flow **Target Resource Type** is monitored for the Disconnections **Metric Name**. The **Statistic** value is set to Sum with a **Comparison Operator** of "greater than or equal to" and a **Threshold** of 10. The **Period** is set to 60 seconds, and only requires 1 out of 1 **Datapoints**. **Treat Missing Data** is set to "ignore."

The result of these settings is: workflow monitor will monitor for disconnections on the flow. If 10 or more disconnections occur within 60 seconds, the alarm will be triggered. 10 or more disconnections in 60 seconds only needs to happen one time for the alarm to be triggered.

Recommended alarm templates for monitoring your AWS media workflow

Workflow monitor's recommended templates are a curated selection of AWS Elemental service metrics with predefined alarm settings appropriate for the metric. If you do not want to create customized alarm templates, recommended templates provide you with best-practice monitoring templates that are created by AWS.

Workflow monitor contains recommended template groups for each supported service. These groups are designed to apply best-practice monitoring to specific types of workflows. Each template group contains a curated selection of alarms configured from service-specific metrics. For example, a recommended template group for a MediaLive multiplex workflow will have a different set of preconfigured metrics than a MediaConnect CDI workflow.

To use recommended alarm templates

1. Follow the steps to create an alarm template group, or select an existing one.
2. In the **Alarm templates** section, select **Import**. You will need to import the AWS recommended templates into your template group.
3. Use the **CloudWatch alarm template groups** dropdown to select an AWS recommended group. These groups contain curated alarms for specific services.
4. Select the templates to import using the check boxes. Each template will list its metrics, preconfigured monitoring values, and provide a description of the metric. When you are done selecting templates, select the **Add** button.

5. The selected templates will move to the **Alarm template(s) to import** section. Review your choices and select **Import**.
6. After the import is complete, the selected templates will be added to the template group. If you want to add more templates, repeat the import process.
7. Imported templates can be customized after import. Alarm settings can be modified to fit your alarming needs.

EventBridge rule groups and templates for monitoring your AWS media workflow

CloudWatch uses Amazon EventBridge rules to send notifications. You begin by creating an event template group. In that event template group, you create event templates that determine what conditions create a notification and who is notified.

This section covers the creation of EventBridge rules using workflow monitor. For more information about how the EventBridge service uses rules, see: EventBridge rules in the *Amazon EventBridge User Guide*

Creating event template groups

Event template groups allow you to sort and classify events based on your use case.

To create an event template group

1. From the workflow monitor console's navigation pane, select **EventBridge rule templates**.
2. Select **Create event template group**.
3. Give the alarm template group a unique **Group name** and optional **Description**.
4. Select **Create**, You will be taken to the newly created alarm template group's details page.

Creating event templates

You can send notifications based on event templates you create.

To create an event template

1. From the event template group's details page, select **Create event template**.
2. Give the event template a unique **Template name** and optional **Description**.
3. In the **Rule settings** section:

1. Select an **Event type**. When selecting an event type, you can choose between several events created by AWS or select **Signal map active alarm** to use an alarm created by an alarm template.
2. Select a **Target service**. This determines how you would like to be notified of this event. You can select Amazon Simple Notification Service or CloudWatch logs.
3. After selecting a target service, select a **Target**. This will be a Amazon SNS topic or a CloudWatch log group, depending on your target service selection.

4. Select **Create** to complete the process.

Workflow monitor signal maps

Signal maps are visual mappings of AWS resources in your media workflow. You can use workflow monitor to start the signal map discovery on any of the supported resource types. During the discovery process, workflow monitor will automatically and recursively map all connected AWS resources. After the signal map has been created, you can use the workflow monitor console to do things like deploy monitoring templates, view metrics, and view details of the mapped resources.

Topics

Creating signal maps for AWS media workflows

You can use workflow monitor signal maps to create a visual mapping of all connected AWS resources in your media workflow.

To create a signal map

1. From the workflow monitor console's navigation pane, select **Signal maps**.
2. Select **Create signal map**.

3. Give the signal map a **Name** and **Description**.
4. In the **Discover new signal map** section, resources in the current account and selected region are displayed. Select a resource to begin signal map discovery. The selected resource will be the starting point for discovery.
5. Select **Create**. Allow a few moments for the discovery process to complete. After the process is complete, you will be presented with the new signal map.

> **Note**
>
> Previews generated in the workflow monitor signal map for AWS Elemental MediaPackage channels are delivered from the MediaPackage Origin Endpoint and will incur Data Transfer Out charges. For pricing, see: MediaPackage pricing.

Viewing signal maps of AWS media workflows

Workflow monitor signal maps allow you to see a visual mapping of all connected AWS resources in your media workflow.

Signal map views

After selecting a signal map, you have two views that can be used to monitor or configure the signal map. **Monitor signal map** and **Configure signal map** is a context-sensitive button found in the upper-right of the signal map console section.

If you select the signal map using the **Signal maps** section of the navigation pane, your signal map will be displayed in the configuration view. The configuration view allows you to make changes to the template groups attached to this signal map, deploy the attached templates, and view the basic details and tags of the signal map.

If you select the signal map using the **Overview** section of the navigation pane, your signal map will be displayed in monitoring view. The monitoring view displays the CloudWatch alarms, EventBridge rules, alerts, logs, and metrics for this signal map.

The view can be changed at any time by selecting the **Monitor/Configure signal map** button in the upper-right. The configuration view requires administrator-level IAM permissions. Required IAM permissions can be viewed here: Workflow monitor IAM policies

Navigating the signal map

A signal map will contain nodes for every supported AWS resource discovered by workflow monitor. Certain resources, such as MediaLive channels and MediaPackage endpoints can display thumbnail previews of the content, if thumbnail previews are available.

Selecting a resource node, and selecting **View selected resource details** from the **Actions** dropdown menu will take you to the associated service's details page. For example, selecting a MediaLive channel and selecting **View selected resource details** will open the MediaLive console's details page for that channel.

Selecting a resource node will filter the list of active alarms to only that node. If you select the resource's **Target ARN** in the active alarm, you will be taken to the associated service's details page, with the selected resource open.

Attaching alarm and event templates to the signal map of your AWS media workflow

After you have created alarm and event templates, you need to attach these to a signal map. Any of the alarm and event templates you have created can be attached to any discovered signal maps.

To attach alarm and event templates to your signal map

1. From the workflow monitor console's navigation pane, select **Signal maps** and select the signal map you want to work with.
2. In the upper-right of the signal map page, in the **CloudWatch alarm template groups** tab, select **Attach CloudWatch alarm template groups**.

    1. In the new section that opens, choose all of the alarm template groups that you want to apply to this signal map, then select **Add**. This will cause the selected alarm template groups to move to the **Attached CloudWatch alarm template groups** section.
    2. Selecting **Save** will save your changes and return you to the signal map page.
3. At the right of the signal map page, select the **EventBridge rule template groups** tab then select **Attach EventBridge rule template groups**.

    1. In the new section that opens, choose all of the event template groups that you want to apply to this signal map, then select **Add**. This will cause the selected rule template groups to move to the **Attached EventBridge rule template groups** section.
    2. Selecting **Save** will save your changes and return you to the signal map page.
4. You have assigned CloudWatch alarm and EventBridge rule templates to the signal map, but the monitoring is not yet deployed. The next section will cover the deployment of the monitoring resources.

Deploying templates to the signal map of your AWS media workflow

After you have attached the alarm and event templates to your signal map, you must deploy the monitoring. Until the deployment is complete, the monitoring of your signal map will not be active.

Workflow monitor will only deploy alarms that are relevant to the selected signal map. For example, the attached alarm template group might contain alarms for multiple services, such as MediaLive, MediaPackage, and MediaConnect. If the selected signal map only contains MediaLive resources, no MediaPackage or MediaConnect alarms will be deployed.

To deploy the monitoring templates

1. After attaching alarm and event template groups to your signal map and saving your changes, select **Deploy monitor** in the **Actions** dropdown menu.
2. You will be asked to confirm the deployment and presented with the number of CloudWatch and EventBridge resources that will be created. If you would like to proceed, select **Deploy**.

   > **Note**
   >
   > There is no direct cost for using workflow monitor. However, there are costs associated with the resources created and used to monitor your workflow.
   > When monitoring is deployed, Amazon CloudWatch and Amazon EventBridge resources are created. When using the AWS Management Console, prior to deploying monitoring to a signal map, you will be notified of how many resources will be created. For more information about pricing, see: CloudWatch pricing and EventBridge pricing.
   > Workflow monitor uses AWS CloudFormation templates to deploy the CloudWatch and EventBridge resources. These templates are stored in a standard class Amazon Simple Storage Service bucket that is created on your behalf, by workflow monitor, during the deployment process and will incur object storage and recall charges. For more information about pricing, see: Amazon S3 pricing.

3. The status of the deployment is displayed next to the name of the signal map. The deployment status is also visible in the **Stacks** section of the CloudFormation console. After a few moments of resource creation and deployment, your signal map monitoring will begin.

Updating the signal map of your AWS media workflow

If a change is made to your workflow, you might need to rediscover the signal map and redeploy monitoring resources. Workflow monitor is a visualization and monitoring tool that does not have the ability to make any changes to your workflow. Signal maps represent a point-in-time visualization of your workflow. In the event that you add, remove, or significantly modify parts of your media workflow, we recommend that you rediscover the signal map. If you have monitoring resources attached to the signal map, we recommend you redeploy monitoring after the rediscovery process.

To rediscover a signal map

1. From the workflow monitor console's navigation pane, select **Signal maps** and select the signal map you want to work with.
2. Verify that you are in the **Configure signal map** view. For more information about changing views, see: Viewing signal maps
3. In the upper-right of the signal map page, select the **Actions** dropdown menu. Select **Rediscover**.
4. You will be presented with the rediscovery screen. Select a resource that is a part of the workflow you are rediscovering. Select the **Rediscover** button.
5. The signal map will be rebuilt according to the current workflow. If you need to redeploy monitoring resources, stay on this signal map's page. Any previously attached monitoring templates will remain attached, but will need to be redeployed.

To redeploy monitoring templates after a signal map rediscovery

1. After the rediscovery, you will be directed to the updated signal map. To redeploy the monitoring templates, select **Deploy monitor** from the **Actions** dropdown menu.
2. You will be asked to confirm the deployment and presented with the number of any CloudWatch and EventBridge resources that will be created. If you would like to proceed, select **Deploy**.
3. The status of the deployment is displayed next to the name of the signal map. After a few moments of resource creation and deployment, your signal map monitoring will begin.

Deleting the signal map of your AWS media workflow

If you not longer need a signal map, it can be deleted. If you have monitoring templates deployed on the signal map, the deletion process will ask you to delete any CloudWatch and EventBridge resources that have been deployed to this signal map. Deleting the deployed resources does not affect the templates that created them. This resource deletion is to ensure that you do not have CloudWatch and EventBridge resources that are deployed but not used.

To delete a signal map

1. From the workflow monitor console's navigation pane, select **Signal maps** and select the radio button next to the signal map you want to delete.
2. Select the **Delete** button. You will be asked to confirm the deletion of the monitoring resources. Select **Delete** to begin the monitoring resource deletion process.
3. The **Monitor deployment** column will display the current status. When the status has changed to **DELETE_COMPLETE**, select the **Delete** button again.
4. You will be asked to confirm deletion of the signal map. Select **Delete** to proceed and delete the signal map.

Workflow monitor quotas

The following section contains quotas for workflow monitor resources. Each quota is on a "per account" basis. If you need to increase a quota for your account, you can use the AWS Service Quotas console to request an increase, unless otherwise noted in the following table.

Quotas

Resource type	Quota
CloudWatch alarm template groups	20
CloudWatch alarm templates	200
EventBridge rule template groups	20
EventBridge rule templates	200
Signal maps	30

Resource type	Quota
Signal maps: CloudWatch alarm template groups attached to a single signal map	5 You cannot increase this quota.
Signal maps: EventBridge rule template groups attached to a single signal map	5 You cannot increase this quota.

Using workflow monitor

Use the **overview** and **signal maps** sections of the workflow monitor console to review the current status of the workflows and any associated alarms, metrics, and logs.

Topics

Workflow monitor overview

The **Overview** section of the workflow monitor console is a dashboard that provides at-a-glance information about your signal maps. In the overview section, you can see the current state of each signal map's monitoring, as well as CloudWatch metrics and any associated CloudWatch logs. You can select any signal map to be taken to that signal maps console page.

Overview filtering

Using the **Search** bar in the overview section, you can filter the list of signal maps using context sensitive constraints. After selecting the search bar, you will be presented with a list of **Properties** to filter by. Selecting a property will present **Operators** such as Equals, Contains, Does not equal, and Does not contain. Selecting an operator will create a list of resources from the selected property type. Selecting one of these resources will cause the signal map list to only display signal maps that fit the constraint you defined.

Overview logs and metrics for workflow monitor

To view CloudWatch metrics and logs for a signal map, select the radio button next to the name of the signal map. A tabbed interface for both metrics and logs will appear beneath the signal map list.

CloudWatch Metrics

CloudWatch metrics for the selected signal map will be context-sensitive and only display metrics associated with the services used in that signal maps workflow. You can use the on-screen metrics tools to customize the displayed metric periods and time ranges.

CloudWatch Logs

If you associated a CloudWatch log group with the signal map, that group will be displayed here.

Using workflow monitor signal maps

From the **overview** section of the console, you can select a specific signal map to view more information about that signal map and its attached monitoring resources.

After selecting a signal map, you will be presented with the signal map and a number of tabbed section containing more information:

- CloudWatch alarms
- EventBridge rules
- AWS Elemental alerts
- Metrics
- Logs
- Basic details

Navigating the signal map

A signal map will contain nodes for every supported AWS resource discovered by workflow monitor. Certain resources, such as MediaLive channels and MediaPackage endpoints can display thumbnail previews of the content, if thumbnail previews are available.

Selecting a resource node, and selecting **View selected resource details** from the **Actions** dropdown menu will take you to the associated service's details page. For example, selecting a

MediaLive channel and selecting **View selected resource details** will open the MediaLive console's details page for that channel.

Selecting a resource node will filter the list of active alarms to only that node. If you select the resource's **Target ARN** in the active alarm, you will be taken to the associated service's details page, with the selected resource open.

Tagging AWS Elemental MediaPackage resources

A tag is a label that you assign to an AWS resource. Each tag consists of a *key* and a *value*, both of which you define. For example, the key might be "stage" and the value might be "test". You can use tags for a variety of purposes. One common use is to control access to AWS resources using tags. For information, see the Controlling access to AWS resources using tags topic in the *IAM User Guide*.

Another common use of tags is to categorize and track your MediaPackage costs. When you apply cost allocation tags to MediaPackage channels, endpoints, and packaging configurations, AWS generates a cost allocation report as a comma-separated value (CSV) file with your usage and costs aggregated by your tags. You can apply tags that represent business categories (such as cost centers, application names, or owners) to organize your costs across multiple services. For more information about using tags for cost allocation, see Using cost allocation tags in the AWS Billing User Guide.

Tag restrictions

The following restrictions apply to tagging AWS Elemental MediaPackage resources:

- Cost allocation tagging is only available for channel, endpoint, and packaging configuration resources. You can't use cost allocation tags for asset or packaging group resources.
- Maximum number of tags that you can assign to a resource – 50.
- Maximum key length – 128 Unicode characters.
- Maximum value length – 256 Unicode characters.
- Valid characters for key and value – a-z, A-Z, 0-9, space, and the following characters: _ . : / = + - and @.
- Keys and values are case sensitive.
- Don't use `aws:` as a prefix for keys; it's reserved for AWS use.
- Can't be used for harvested live-to-VOD assets.

Managing tags

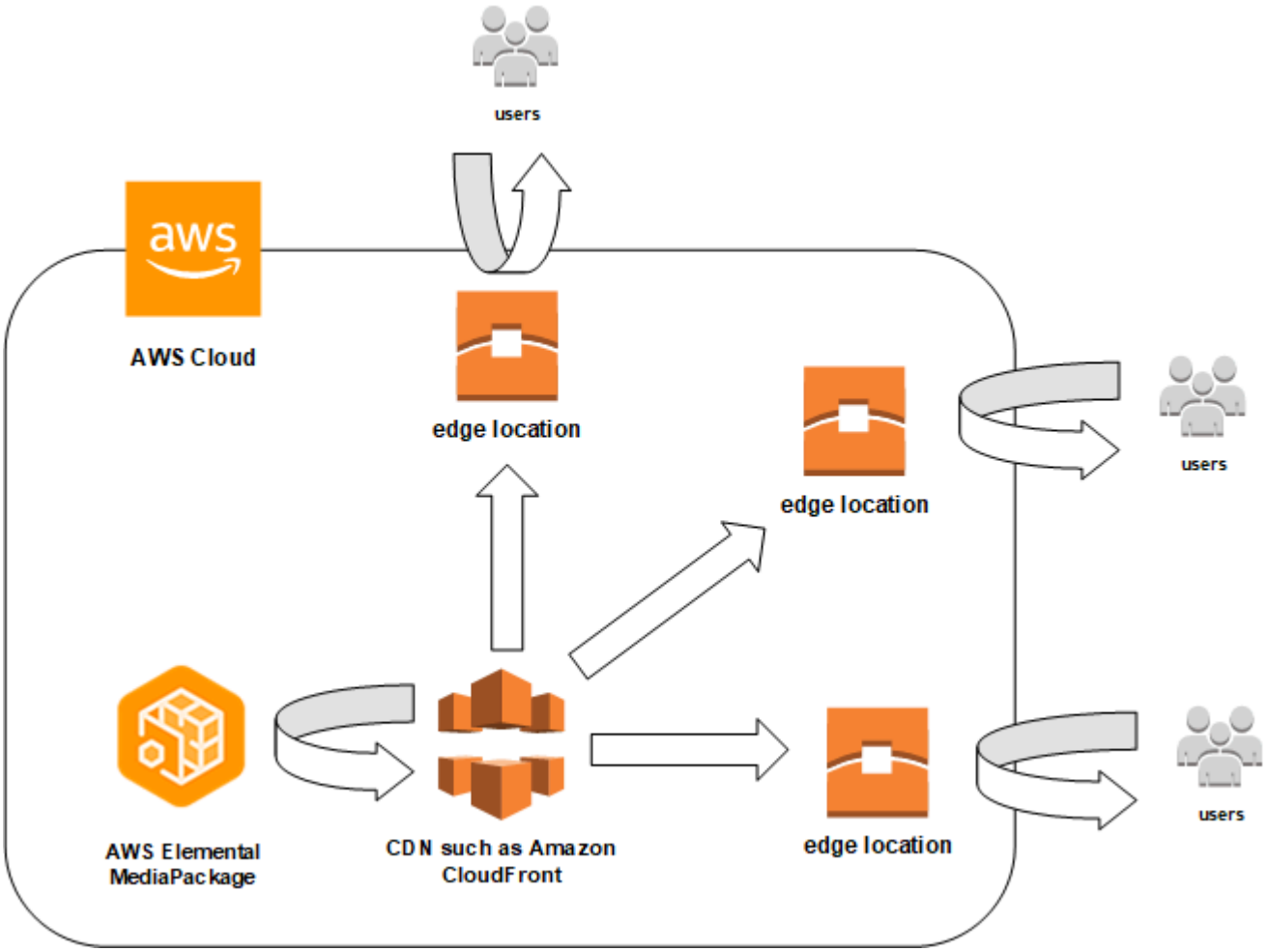
You can use the AWS Elemental MediaPackage API or the AWS CLI to add, edit, or delete the values for these properties.

For more information, see the actions related to tags in the following reference documentation:

- Tags resource-arn in the *AWS Elemental MediaPackage live API reference*.
- Tags resource-arn in the *AWS Elemental MediaPackage VOD API reference*.
- tag-resource in the AWS CLI *MediaPackage reference*.

Working with CDNs

You can use a content delivery network (CDN) such as Amazon CloudFront to serve the content that you store in AWS Elemental MediaPackage. A CDN is a globally distributed set of servers that caches content such as videos. When a user requests your content, the CDN routes the request to the edge location that provides the lowest latency. If your content is already cached in that edge location, the CDN delivers it immediately. If your content is not currently in that edge location, the CDN retrieves it from your origin (in this case, the MediaPackage endpoint) and distributes it to the user. The following illustration shows this process.



The following sections provide procedures for working with distributions from Amazon CloudFront.

Topics

- Creating a Distribution
- Viewing a Distribution

- Editing a Distribution
- Deleting a Distribution

Creating a Distribution

A distribution in Amazon CloudFront holds all information about content delivery, including where content is coming from and how it's tracked and managed. The distribution holds origins (where content is originating from) and behaviors (where content requests are routed based on specified patterns in the request).

You can create a distribution from the CloudFront console. The following section describes this approach.

Topics

- Creating a Distribution from Amazon CloudFront

Creating a Distribution from Amazon CloudFront

After you create a channel and its endpoints in AWS Elemental MediaPackage, note the URLs for each of the endpoints. These URLs are what you use for the origin domain names for your CloudFront distribution. You need one origin for each endpoint on the channel in MediaPackage.

For detailed steps about creating a distribution in Amazon CloudFront with AWS Elemental MediaPackage endpoints as the origins, see Delivering Live Streaming Video in the *Amazon CloudFront Developer Guide*.

Viewing a Distribution

As described in Viewing channel details, you can view basic information about a distribution that was created in MediaPackage, such as the distribution ID and description. Note that the ID links to the CloudFront management console.

Access more detailed information about the distribution from the Amazon CloudFront console. For help accessing this information, see Viewing and Updating Distribution in the *Amazon CloudFront Developer Guide*.

Editing a Distribution

Edit an Amazon CloudFront distribution from the CloudFront console.

The only edit that AWS Elemental MediaPackage can make to an origin is to create an origin when you add an endpoint to a channel in MediaPackage. You can't edit a distribution from the MediaPackage console.

To access the distribution in CloudFront, choose the distribution's ID on the channel's details page. For more information about editing a distribution in CloudFront, see Viewing and Updating Distribution in the *Amazon CloudFront Developer Guide*.

Important

When you're editing a distribution, do not change the default on the **Tagging** page. CloudFront uses the AWS Elemental MediaPackage channel ID in this tag to link the distribution and the channel together. If the tag is modified, then you will no longer be able to view or manage the distribution from MediaPackage.

Deleting a Distribution

Delete an Amazon CloudFront distribution from the CloudFront console. You can't delete a distribution from the AWS Elemental MediaPackage console.

To access the distribution in CloudFront, choose the distribution's ID on the channel's details page. For more information about deleting a distribution in CloudFront, see Deleting a Distribution in the *Amazon CloudFront Developer Guide*.

Using this service with an AWS SDK

AWS software development kits (SDKs) are available for many popular programming languages. Each SDK provides an API, code examples, and documentation that make it easier for developers to build applications in their preferred language.

SDK documentation	Code examples
AWS SDK for C++	AWS SDK for C++ code examples
AWS CLI	AWS CLI code examples
AWS SDK for Go	AWS SDK for Go code examples
AWS SDK for Java	AWS SDK for Java code examples
AWS SDK for JavaScript	AWS SDK for JavaScript code examples
AWS SDK for Kotlin	AWS SDK for Kotlin code examples
AWS SDK for .NET	AWS SDK for .NET code examples
AWS SDK for PHP	AWS SDK for PHP code examples
AWS Tools for PowerShell	AWS Tools for PowerShell code examples
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) code examples
AWS SDK for Ruby	AWS SDK for Ruby code examples
AWS SDK for Rust	AWS SDK for Rust code examples
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP code examples
AWS SDK for Swift	AWS SDK for Swift code examples

For examples specific to this service, see Code examples for MediaPackage using AWS SDKs.

Example availability

Can't find what you need? Request a code example by using the **Provide feedback** link at the bottom of this page.

Code examples for MediaPackage using AWS SDKs

The following code examples show how to use MediaPackage with an AWS software development kit (SDK).

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios.

For a complete list of AWS SDK developer guides and code examples, see Using this service with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Code examples

- Basic examples for MediaPackage using AWS SDKs
  - Actions for MediaPackage using AWS SDKs
    - Use ListChannels with an AWS SDK or CLI
    - Use ListOriginEndpoints with an AWS SDK or CLI

Basic examples for MediaPackage using AWS SDKs

The following code examples show how to use the basics of AWS Elemental MediaPackage with AWS SDKs.

Examples

- Actions for MediaPackage using AWS SDKs
  - Use ListChannels with an AWS SDK or CLI
  - Use ListOriginEndpoints with an AWS SDK or CLI

Actions for MediaPackage using AWS SDKs

The following code examples demonstrate how to perform individual MediaPackage actions with AWS SDKs. Each example includes a link to GitHub, where you can find instructions for setting up and running the code.

The following examples include only the most commonly used actions. For a complete list, see the AWS Elemental MediaPackage API Reference.

Examples

- Use ListChannels with an AWS SDK or CLI
- Use ListOriginEndpoints with an AWS SDK or CLI

Use `ListChannels` with an AWS SDK or CLI

The following code examples show how to use `ListChannels`.

CLI

AWS CLI

To list all channels

The following `list-channels` command lists all of the channels that are configured on the current AWS account.

```
aws mediapackage list-channels
```

Output:

```
{
    "Channels": [
        {
            "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/584797f1740548c389a273585dd22a63",
            "HlsIngest": {
                "IngestEndpoints": [
                    {
                        "Id": "584797f1740548c389a273585dd22a63",
                        "Password": "webdavgeneratedpassword1",
                        "Url": "https://9be9c4405c474882.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",
                        "Username": "webdavgeneratedusername1"
                    },
                    {
                        "Id": "7d187c8616fd455f88aaa5a9fcf74442",
                        "Password": "webdavgeneratedpassword2",
```

```
                        "Url": "https://7bf454c57220328d.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",
                        "Username": "webdavgeneratedusername2"
                    }
                ]
            },
            "Id": "test",
            "Tags": {}
        }
    ]
}
```

For more information, see Viewing Channel Details in the *AWS Elemental MediaPackage User Guide*.

- For API details, see ListChannels in *AWS CLI Command Reference*.

Rust

SDK for Rust

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

List channel ARNs and descriptions.

```
async fn show_channels(client: &Client) -> Result<(), Error> {
    let list_channels = client.list_channels().send().await?;

    println!("Channels:");

    for c in list_channels.channels() {
        let description = c.description().unwrap_or_default();
        let arn = c.arn().unwrap_or_default();

        println!("  Description : {}", description);
        println!("  ARN :         {}", arn);
        println!();
```

```
    }

    Ok(())
}
```

- For API details, see ListChannels in *AWS SDK for Rust API reference*.

For a complete list of AWS SDK developer guides and code examples, see Using this service with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `ListOriginEndpoints` with an AWS SDK or CLI

The following code examples show how to use `ListOriginEndpoints`.

CLI

AWS CLI

To list all origin-endpoints on a channel

The following `list-origin-endpoints` command lists all of the origin endpoints that are configured on the channel named `test`.

```
aws mediapackage list-origin-endpoints \
    --channel-id test
```

Output:

```
{
    "OriginEndpoints": [
        {
            "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/247cff871f2845d3805129be22f2c0a2",
            "ChannelId": "test",
            "DashPackage": {
                "ManifestLayout": "FULL",
                "ManifestWindowSeconds": 60,
                "MinBufferTimeSeconds": 30,
                "MinUpdatePeriodSeconds": 15,
                "PeriodTriggers": [],
```

```
                "Profile": "NONE",
                "SegmentDurationSeconds": 2,
                "SegmentTemplateFormat": "NUMBER_WITH_TIMELINE",
                "StreamSelection": {
                    "MaxVideoBitsPerSecond": 2147483647,
                    "MinVideoBitsPerSecond": 0,
                    "StreamOrder": "ORIGINAL"
                },
                "SuggestedPresentationDelaySeconds": 25
            },
            "Id": "tester2",
            "ManifestName": "index",
            "StartoverWindowSeconds": 0,
            "Tags": {},
            "TimeDelaySeconds": 0,
            "Url": "https://8343f7014c0ea438.mediapackage.us-
west-2.amazonaws.com/out/v1/247cff871f2845d3805129be22f2c0a2/index.mpd",
            "Whitelist": []
        },
        {
            "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/869e237f851549e9bcf10e3bc2830839",
            "ChannelId": "test",
            "HlsPackage": {
                "AdMarkers": "NONE",
                "IncludeIframeOnlyStream": false,
                "PlaylistType": "EVENT",
                "PlaylistWindowSeconds": 60,
                "ProgramDateTimeIntervalSeconds": 0,
                "SegmentDurationSeconds": 6,
                "StreamSelection": {
                    "MaxVideoBitsPerSecond": 2147483647,
                    "MinVideoBitsPerSecond": 0,
                    "StreamOrder": "ORIGINAL"
                },
                "UseAudioRenditionGroup": false
            },
            "Id": "tester",
            "ManifestName": "index",
            "StartoverWindowSeconds": 0,
            "Tags": {},
            "TimeDelaySeconds": 0,
            "Url": "https://8343f7014c0ea438.mediapackage.us-
west-2.amazonaws.com/out/v1/869e237f851549e9bcf10e3bc2830839/index.m3u8",
```

```
            "Whitelist": []
        }
    ]
}
```

For more information, see Viewing all Endpoints Associated with a Channel in the *AWS Elemental MediaPackage User Guide*.

- For API details, see ListOriginEndpoints in *AWS CLI Command Reference*.

Rust

SDK for Rust

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

List your endpoint descriptions and URLs.

```
async fn show_endpoints(client: &Client) -> Result<(), Error> {
    let or_endpoints = client.list_origin_endpoints().send().await?;

    println!("Endpoints:");

    for e in or_endpoints.origin_endpoints() {
        let endpoint_url = e.url().unwrap_or_default();
        let endpoint_description = e.description().unwrap_or_default();
        println!("  Description: {}", endpoint_description);
        println!("  URL :        {}", endpoint_url);
        println!();
    }

    Ok(())
}
```

- For API details, see ListOriginEndpoints in *AWS SDK for Rust API reference*.

For a complete list of AWS SDK developer guides and code examples, see Using this service with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Quotas in AWS Elemental MediaPackage

The following sections provide information about the quotas in AWS Elemental MediaPackage.

Topics

Live content quotas

This section describes the quotas for live content in AWS Elemental MediaPackage. For information about requesting an increase to soft quotas, see AWS service quotas. Hard quotas can't be changed.

Live soft quotas

The following table describes quotas in AWS Elemental MediaPackage for live content that can be increased. For information about changing quotas, see AWS Service Quotas.

For some customers, your account quota might be below these published quotas. If you believe that you encountered a Resource limit exceeded error wrongfully, use the Service Quotas console to request quota increases.

Resource	Default quota
Maximum Channels	30 **Note** Increasing your channel quota doesn't always mean that you also need to increase your endpoints. For example, if you need 34 channels and want to serve HLS, HLS encrypted, and DASH content from each channel, you need only 3 endpoints for each channel (one for each output type). The default

Resource	Default quota
	endpoint quota is 10 so, although you do need a channel quota increase, you don't need to increase your endpoint quota. You won't exceed the quota of 10 endpoints *per channel*.
Maximum Endpoints per Channel	10 This is a *per channel* quota. Each endpoint represents the output package that you use. If one channel serves HLS, HLS encrypted, DASH, DASH encrypted, Microsoft Smooth, and Microsoft Smooth encrypted content, then that channel has 6 endpoints and falls within the 10 endpoints quota. If you have 10 channels set up this same way, then you still haven't exceeded the quota because each channel uses only 6 endpoints.
Maximum Concurrent Harvest Jobs	10
Maximum Live Manifest Length	5 minutes

Live hard quotas

The following table describes quotas in AWS Elemental MediaPackage for live content that can't be increased.

Resource or operation	Quota
Ingest streams per Channel	20 streams per channel
Maximum Content Age for Time-shifted Viewing	336 hours (14 days)

Resource or operation	Quota
Maximum Time-shifted Manifest Length	24 hours for all supported output formats
Maximum Live-to-VOD Manifest Length	24 hours for all supported output formats
Request Rates per Channel	Input: 50 requests per second
Request Rates per Endpoint	• Media segments output: 300 requests per second • Manifests output: 5000 requests per second **Note** The per Endpoint origination request rate quotas are indicative only and based on typical traffic patterns when using a properly configured CDN. The request rate quotas are applicable for live events, linear channels, and time-shifted viewing. The request rate quotas may be lower under certain conditions like misconfigured CDNs or players generating abnormal levels of origin requests with unique HTTP headers values, or unique query strings values appended to the playback URLs.
REST API Requests	• Steady state: 5 requests per second • Bursting: 50 requests per second
Tracks per Ingest Stream	10 The maximum number of tracks (audio, video, subtitle, etc.) per stream that you can ingest.

VOD content quotas

This section describes the quotas for video on demand (VOD) content in AWS Elemental MediaPackage. For information about requesting an increase to soft quotas, see AWS Service Quotas. Hard quotas can't be changed.

VOD soft quotas

The following table describes quotas in AWS Elemental MediaPackage for VOD content that can be increased. For information about changing quotas, see AWS Service Quotas.

For some customers, your account quota might be below these published quotas. If you believe that you encountered a Resource limit exceeded error wrongfully, use the Service Quotas console to request quota increases.

Resource	Default quota
Maximum Packaging Groups	10 **Note** Increasing your packaging group quota doesn't always mean that you also need to increase your assets or packaging configurations. For example, if you need 14 groups and want to serve HLS, HLS encrypted, and DASH content from each asset, you need only 3 packaging configurations for each asset (one for each output type). You do need to increase your packaging group quota, but not the packaging configuration quota because you have fewer than 10 configurations per packaging group.
Maximum Packaging Configurations per Packaging Group	10

Resource	Default quota
	This is a *per packaging group* quota. Each packaging configuration represents the output package that you use. If one packaging group has configurations for HLS, HLS encrypted, DASH, DASH encrypted, Microsoft Smooth, and Microsoft Smooth encrypted content, then that group has 6 packaging configura tions and falls within the 10 configurations quota. If you have 10 packaging groups set up this same way, then you still haven't exceeded the quota because each group uses only 6 configurations.
Maximum Assets per Packaging Group	10,000 This is a *per packaging group* quota. For example, if you have 10,500 assets spread out over multiple packaging groups, then you still haven't exceeded the quota if each group has no more than 10,000 assets.

VOD hard quotas

The following table describes quotas within AWS Elemental MediaPackage for VOD content that can't be increased.

Resource or operation	Quota
Ingest streams per Packaging Configuration	20
Request Rates per Packaging Configuration	• Media segments output: 600 requests per second • Manifests output: 300 requests per second

Resource or operation	Quota
	Note The per Packaging Configuration origination request rate quotas are indicative only and based on typical traffic patterns when using a properly configured CDN. The request rate quotas may be lower under certain conditions like misconfigured CDNs or players generating abnormal levels of origin requests with unique HTTP headers values, or unique query strings values appended to the playback URLs.
REST API Requests	• Steady state: 5 requests per second • Bursting: 50 requests per second
Tracks per Ingest Stream	10 The maximum number of tracks (audio, video, subtitle, etc.) per stream that you can ingest.

AWS Elemental MediaPackage related information

The following table lists related resources that you'll find useful as you work with MediaPackage.

Resource	Description
Classes and Workshops	Links to role-based and specialty courses and self-paced labs to help sharpen your AWS skills and gain practical experience.
AWS Developer Tools	Links to developer tools, SDKs, IDE tool kits, and command line tools for developing and managing AWS applications.
AWS Whitepapers	Links to a comprehensive list of technical AWS whitepapers, covering topics such as architect ure, security, and economics and authored by AWS Solutions Architects or other technical experts.
AWS Support Center	The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.
AWS Support	The primary web page for information about AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
Contact Us	A central contact point for inquiries concernin g AWS billing, account, events, abuse, and other issues.

Resource	Description
AWS Site Terms	Detailed information about our copyright and trademark; your account, license, and site access; and other topics.

Document history for User Guide

The following table describes important changes in each release of the *AWS Elemental MediaPackage User Guide* after May 2018. For notification about updates to this documentation, you can subscribe to an RSS feed.

- **API version:** latest

Change	Description	Date
Added DRM query param info	Added topic about query parameters to remove `EXT-X-SESSION-KEY` tags from manifest responses.	January 10, 2025
Added accessibility support information	Added *Accessibility support* to the list of MediaPackage features, descriing requirements for audio and subtitles accessibility signaling.	December 17, 2024
Workflow monitor	Analyze AWS media services and create signal maps, visualizations of the media workflow, between those services. Use the signal maps to generate monitoring alarms and notifications using CloudWatch, EventBridge, and CloudFormation.	April 11, 2024
New DASH manifest setting	Added the **DRM top level compact** manifest settings option.	January 26, 2024
Removed cenc encryption option from CMAF for VOD	CMAF for VOD workflows using SPEKE Version 2.0	April 24, 2023

	supports only cbcs encryption. Removed cenc encryption from the table.	
Added `trickplay_type` value	Updated `trickplay_type` to include none, which filters out all trickplay tracks.	April 24, 2023
Added missing quotation marks	Added missing closing quotation mark to example.	April 24, 2023
Clarified SPEKE support	Updated SPEKE tables to clarify protocol and DRM system support.	February 20, 2023
Updated the IAM guidance	Updated guide to align with the IAM best practices . For more information, see Security best practices in IAM.	February 8, 2023
Removed two fields	Removed two stream selection fields to align with their removal from the console.	January 5, 2023
Corrected typo	Changed instance of "authentication" to "authorization".	December 19, 2022
Updated I-frame only trick-play support	MediaPackage now supports I-frame only trick-play for DASH VOD.	November 7, 2022
New DASH manifest setting	Added the **Include IFrame-only streams** manifest settings option.	November 7, 2022

Updated `trickplay_height` and `video_height` query parameters	Updated `trickplay_height` and `video_height` query parameters when using I-frame only and image-based trick-play.	October 27, 2022
SPEKE Version 2.0 is now available for both VOD and live	MediaPackage now supports SPEKE Version 2.0 with VOD CMAF and DASH workflows.	October 10, 2022
New SPEKE option for DASH	MediaPackage now supports DASH SPEKE Version 2.0 for VOD.	October 10, 2022
New SPEKE option for CMAF	MediaPackage now supports CMAF SPEKE Version 2.0 for VOD.	October 10, 2022
New CMAF encryption option	MediaPackage now supports AES-CTR encryption to encrypted CMAF endpoints.	September 2, 2022
Updated information regarding SPEKE Version 2.0 presets	Updated the CPIX Version to 2.3. Updated the SPEKE Version 2.0 table that describes the support matrix for protocol and DRM system.	July 19, 2022
New documentation of SPEKE Version 2.0 presets	MediaPackage supports SPEKE Version 2.0 presets for unencrypted tracks and encrypted tracks, a single encryption key for all audio and video tracks, and multiple encryption keys for audio and video tracks.	July 19, 2022

New **Include IFrame only stream** option	MediaPackage now supports the **Include IFrame only stream** to include an additional I-frame only stream along with the other tracks in the manifest.	July 19, 2022
Manifest update time	MediaPackage playback responses now include custom headers that specify when MediaPackage last updated the manifest.	January 3, 2022
Add information about image-based trick-play for HLS live	MediaPackage now supports image-based trick-play for HLS live.	November 24, 2021
Added support for `includeAudio` parameter	MediaPackage now supports `includeAudio` in .smil manifests	November 1, 2021
New DVB subtitle option	MediaPackage can now passthrough DVB subtitles into packaging configuration outputs.	October 20, 2021
New DVB subtitle option	MediaPackage can now passthrough DVB subtitles into the HLS outputs.	October 20, 2021
New trick-play topic	MediaPackage now supports trick-play for DASH.	October 15, 2021

Asset playback status	You can now view playable status information about an asset. This lets you tell whether or not an asset is ready for playback, or if processing has failed.	October 7, 2021
MediaPackage now supports SPEKE Version 2.0 with live workflows	Added information about SPEKE Version 2.0 with live CMAF and DASH workflows.	September 7, 2021
New guidance for integrating with SPEKE Version 2.0	MediaPackage now supports SPEKE Version 2.0.	September 6, 2021
New metadata passthrough topic	MediaPackage now supports ID3 and KLV metadata passthrough. When ID3 or KLV metadata is present in a channel's input stream, MediaPackage automatically passes through the metadata to the output stream.	June 30, 2021
New CMAF encryption options	MediaPackage now offers constant IV and multiple system IDs for CMAF packaging configurations.	June 30, 2021
Updated cost allocation tagging content	Clarified availability of cost allocation tagging.	May 27, 2021
Added DASH endpoints for live-to-VOD	MediaPackage now supports DASH endpoint requirements for live-to-VOD.	May 14, 2021

New DASH and CMAF manifest setting	Added the **Include encoder configuration in segments** manifest settings option. If you turn on this option, MediaPackage places Sequence Parameter Set (SPS), Picture Parameter Set (PPS), and Video Parameter Set (VPS) metadata in each video segment instead of in the init fragment.	April 28, 2021
Added `audio_bitrate` query parameter	`audio_bitrate` can now be used as a manifest filter query parameter.	March 22, 2021
Access logging	MediaPackage now supports access logging for VOD.	February 24, 2021
Added information about MediaPackage event handling behavior for CloudWatch events	MediaPackage emits events to CloudWatch on a best effort basis.	January 7, 2021
Access logging	MediaPackage now supports access logging, which provides detailed records for the requests that are made to a channel. This feature is available for live workflows.	October 21, 2020
New UTC Timing option for DASH endpoints	MediaPackage now supports UTC timing for DASH endpoints.	October 20, 2020
Support for SCTE-35 EXT-x-DATERANGE tag	Added a new EXT-X-DAT ERANGE section to the SCTE-35 Ad Marker topic.	August 7, 2020

Moved the maximum live manifest length from hard quota to soft quota	The maximum live manifest length is a soft quota. Moved this entry from hard quotas to soft quotas.	June 24, 2020
Added information about CDN authorization for VOD	Added *CDN Authorization in AWS Elemental MediaPack age* topic to describe how to add authorization to requests from your CDN.	May 29, 2020
Updated manifest filtering topic	Added 6 new parameters and updated the character limit to 1024.	May 15, 2020
Remove VOD tagging restricti on	MediaPackage now supports tagging for VOD.	April 23, 2020
New manifest filtering topic	Added a new *Manifest Filtering* topic.	April 8, 2020
Updated maximum time-shif ted and live-to-VOD manifest length	The maximum manifest length is now 24 hours for all supported output formats.	March 9, 2020
New VOD DASH-ISO manifest console settings	Added new DASH-ISO manifest console settings for VOD packaging configura tions. Compact DASH, new segment template formats, and period trigger options are now available.	February 25, 2020
Multi-period DASH is now available for both live and VOD.	Removed references to "live only" support for multi-period DASH.	February 25, 2020

Compact DASH manifests are now available for both VOD and live	Removed reference to "live only" support for compact DASH.	February 25, 2020
Fixed bug	Successful harvest jobs CloudWatch events return `"status": "SUCCEEDED"` .	February 10, 2020
Added information about CDN authorization	Added *CDN Authorization in AWS Elemental MediaPack age* topic to describe how to add authorization to requests from your CDN.	December 23, 2019
Added information about VOD playback events.	Added example playback ready notification events for ingested VOD content.	November 8, 2019
Added information about live-to-VOD CloudWatch Events.	Added example harvest job notification events for live-to-VOD content harvesting.	October 15, 2019
Added .smil manifest information	Added *Requirements for .smil manifests* topic to describe the supported `.smil` manifest format for VOD ingest.	October 10, 2019
Added live-to-VOD (video on demand) topics	Throughout guide, added and updated topics about creating live-to-VOD assets, including *Creating Live-to-VOD Assets* and *Live-to-VOD Content Delivery*.	October 1, 2019

Updated time-shifted manifest length limit.	AWS Elemental MediaPackage can now produce time-shifted manifests up to 18 hours for DASH with compact manifest, HLS, and CMAF.	August 21, 2019
Added supported inputs and outputs info	Added *Supported Inputs and Outputs* topic that describes what input types, containers, and codecs MediaPackage supports.	June 21, 2019
Added configurable SCTE-35 options.	Added *SCTE-35 Message Options in AWS Elemental MediaPackage* topic that describes how you can configure MediaPackage behavior when there are SCTE-35 markers in your input content.	June 21, 2019
Added security chapter.	Added the *Security* chapter to enhance and standardize security topics for MediaPackage.	June 5, 2019
Added video on demand (VOD) topics	Throughout guide, added topics about working with VOD content: *VOD Content Processing*, *Allowing MediaPackage to Access Amazon Simple Storage Service*, *VOD Content Delivery*, *Delivering VOD Content*, *VOD Content Metrics*, and *VOD Content Limits*.	May 17, 2019

Added further information about DASH manifest SegmentTemplate format options.	Added the *Duration Attribute* topic to discuss how to include duration information in `SegmentTemplate` instead of using `SegmentTimeline`.	May 10, 2019
Updated time-shifted manifest length limit	AWS Elemental MediaPackage can now produce time-shifted manifests up to 9 hours.	May 1, 2019
Added information about live and VOD manifests	Added the *Live and VOD Manifest Reference* topic that explains when MediaPackage serves a live or VOD manifest.	April 16, 2019
Added tagging information	Added *Tagging Resources* topic to discuss how tagging channels and endpoints works in AWS Elemental MediaPackage.	March 4, 2019
Added information about DASH manifest SegmentTemplate format options.	Added the *DASH Manifest Segment Template Format* topic to discuss how to change variables in the media URL in the SegmentTemplate object of the DASH manifest.	February 6, 2019
Added DASH manifest treatment information	Added *DASH Manifest Options* topic to discuss the ways that you can modify output DASH manifests.	February 6, 2019

Added AWS CloudTrail logging information.	Added *Logging AWS Elemental MediaPackage API Calls with AWS CloudTrai l* topic to discuss using CloudTrail to log actions in the AWS Elemental MediaPackage API.	December 21, 2018
Added information about compact DASH manifests	Added a *Compacted DASH Manifests* topic to discuss how compacting DASH output manifests works in AWS Elemental MediaPackage.	December 18, 2018
Updated content retention window limit	AWS Elemental MediaPackage now retains content for 336 hours (14 days).	November 13, 2018
Added content key encryption to DRM encryption	Added the option to encrypt content keys. Prior to this, AWS Elemental MediaPack age supported clear key delivery only. To use content key encryption, your DRM key provider must support encrypted content keys. If you enable this feature for a key provider that doesn't handle content key encryption, the operation fails.	November 8, 2018
Added input redundancy information	Added *How Input Redundanc y Works* topic to discuss how MediaPackage can receive two identical streams for back-up purposes.	August 28, 2018

Added Amazon CloudFront console integration information	Added sections about working with distributions in CloudFront, including how to create a distribution from the AWS Elemental MediaPackage console.	August 3, 2018
Added information about multi-period DASH.	Added *Multi-period DASH in AWS Elemental MediaPackage* topic to discuss the purpose and functionality of multiple periods in DASH manifests.	July 18, 2018
Added CDN information	Added *Working with CDNs* topic to discuss how AWS Elemental MediaPackage works with CDNs such as Amazon CloudFront.	May 31, 2018
Added information about creating event notifications	Added the *Creating Event Notifications* topic that describes how to use Amazon CloudWatch Events and Amazon Simple Notification Service to notify you of new events.	January 22, 2018

Earlier updates

The following table describes important changes in each release of the *AWS Elemental MediaPackage User Guide* before May 2018.

Change	Description	Date
Initial document creation	New document.	November 27, 2017

Change	Description	Date
Corrected links and added whitelisting	Corrected links to the AWS Elemental MediaPackage console and AWS Elemental MediaPackage API Reference. In Working with Endpoints , added reference to access control fields.	December 1, 2017
Added IAM policy informati on specific to AWS Elemental MediaPackage	In Setting up MediaPack age, added instructions for creating non-admin roles with limited permissions.	December 13, 2017
Added hard limit information	In Quotas in AWS Elemental MediaPackage, added information about limits that can't be changed (hard limits).	December 20, 2017
Updated IAM policy informati on	In Setting up MediaPackage, added information about policies specific to AWS Elemental MediaPackage.	January 5, 2018
Added CMAF endpoint information	Added Creating a CMAF endpoint section for new output type.	April 6, 2018
Updated feature functionality	In Features of AWS Elemental MediaPackage, added feature support for HDR-10.	April 30, 2018

Change	Description	Date
Added CDN information	Added topic Working with CDNs to discuss how AWS Elemental MediaPackage works with CDNs such as Amazon CloudFront.	May 31, 2018

Note

- The AWS Media Services are not designed or intended for use with applications or in situations requiring fail-safe performance, such as life safety operations, navigation or communication systems, air traffic control, or life support machines in which the unavailability, interruption or failure of the services could lead to death, personal injury, property damage or environmental damage.

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.