User Guide

AWS Elemental Live

AWS Elemental Live: User Guide

Table of Contents

What is AWS Elemental Live?

AWS Elemental Live is video encoding solution that runs on hardware or on virtual machines in your premises. It encodes and transcodes real-time (live) video for broadcast and streaming delivery.

You use Elemental Live to transform live video content from one format and package into other formats and packages. You typically need to transform the content in order to provide a format and package that a playback device can handle. Playback devices include smartphones and set-top boxes attached to televisions.

This high-level introduction to Elemental Live includes workflow diagrams, detailed feature specifications, and a link to Sales.

Topics

- Information about using Elemental Live
- Interfaces for Elemental Live
- How Elemental Live works
- Elemental Live terminology

Information about using Elemental Live

Information about AWS Elemental Live is available in the following guides and resources.

This guide

This guide offers conceptual and procedural information for specific features of Elemental Live. For a list of the features covered in this guide, see the topic list previous to this section.

Elemental Live API guide

This guide is intended for system integrators and Elemental Live operators. It contains the following information:

- An outline of the interfaces for machine and human control, configuration, and monitoring. For a summary of the interfaces covered in the guide, see the next section.
- An overview of how to work with transcoding events, event profiles, and presets.

- A list and explanation of event and system parameters.

This guide is available on the Support tab of the web interface of your Elemental Live appliance.

AWS Elemental Live Installation Guide

AWS Elemental Live Upgrade Guide

AWS Elemental Live Configuration Guide

Interfaces for Elemental Live

Elemental Live can be controlled, configured, and monitored through the following interfaces.

Interface	Description
Web browser via HTML	Using a web browser is the easiest way to control, configure, and monitor Elemental Live. This interface is used when a human is interacting with the server or when no automation or integration with other systems is required.
Web Services REST Interface	The REST-based interface supports all features of the web interface as well as automation features.
SNMP Interface	The SNMP interface allows basic monitorin g and control of Elemental Live. It allows a management system to query the state of the service and content it manages.

How Elemental Live works

From the point of view of Elemental Live, a live streaming workflow that includes Elemental Live involves three systems:

- An Elemental Live *event*, which ingests and transcodes source content.

- One or more *upstream systems* that provide the *source content* (the video) to Elemental Live.

  Examples of an upstream system are a streaming camera or appliance that is directly connected to the internet, or a contribution encoder that is located in a sports stadium where a sports event is being held.

  The source content is in a specific package format and protocol. For example, the source content might be available as streaming HLS or streaming TS (transport stream). The source content contains video, audio, and optional captions streams that are in specific codecs or formats.
- One or more *downstream systems* that are the destinations for the output that Elemental Live produces.

  A typical downstream system consists of an origin service or a packager that is connected to Elemental Live, a content distribution network (CDN) that is downstream of the origin service or the packager, and a playback device or website where the users view the content. .

To create an Elemental Live workflow, you create an event. Broadly speaking this event contains two sets of configuration information:

- A list of inputs (sources) and information about how to ingest those sources.

- A list of output groups that specifying packaging and encoding information.

To start processing the content, you start the event. When the event is running, it ingests the source content from the upstream system that is identified by the input. The event then transcodes that video (and the related audio, captions, and metadata) and creates outputs. Elemental Live sends the outputs to the specified downstream systems.

Elemental Live terminology

CDN

A content distribution network (CDN) is a network of servers that is downstream of the origin server or packager. The CDN distributes the content from the origin server to dozens or hundreds of networked servers that serve the content to your viewing users. This distributed network ensures that content can be delivered to thousands or millions of viewing users simultaneously.

Downstream system

The *downstream system* is a set of one or more servers that is positioned after Elemental Live in the workflow. The downstream system handles the content that is output from Elemental Live.

Encode

An encode exists within an output. There are three types of encodes: video, audio, and captions. Each encode contains the instructions for one video stream, one audio stream, or one captions track that the transcoding process will create. Different encodes have different characteristics. For example, one video encode produced from the input might be high resolution while another is low resolution.

Event

An Elemental Live event ingests and transcodes (decodes and encodes) source content from the inputs that are attached to that event, and packages the new content into outputs.

Event configuration

An Elemental Live event configuration contains information about how the event ingests, transcodes, and packages content into output.

Origin service

An origin service might be part of the downstream system that is positioned after Elemental Live in the workflow. It accepts the video output from Elemental Live.

Output

An output exists within an output group. It is a collection of encodes that you want to handle as one set.

Output Group

An output group is a collection of outputs within the Elemental Live event.

Packager

A packager might be part of the downstream system. It accepts the video output from Elemental Live and repackages it. AWS Elemental MediaPackage is a packager.

Playback device

A playback device is the final component of the downstream system. It is the device that the people who are your audience use to view the video.

Source content

The video content that Elemental Live transcodes. The content typically consists of video, audio, captions, and metadata.

Upstream system

The system that is in front of Elemental Live in the workflow and that holds the source content. Examples of an upstream system are a streaming camera or appliance that is directly connected to the internet, or a contribution encoder that is located in a stadium at a sports event.

Performance of Elemental Live appliances

This section looks at the factors that affect the performance of an AWS Elemental Live appliance, with particular emphasis on the newest appliances. It provides guidance about how to maximize performance while achieving the preferred balance between density, speed, and quality.

- **Density** is the number of output encodes or events that you can run on an appliance.

  The density is affected by the following:
  - The capabilities of the appliance.
  - The compute demands of the individual events, especially the demands to achieve the desired video quality in the outputs.
- **Speed** is the rate at which the workflow content can be processed or transmitted. With live events, there must be enough compute power applied to achieve real-time ingest of the input and real-time encoding of the output.
- **Quality** refers primarily to the video quality.

This section describes how to obtain the balance that suits your requirements.

Topics

Recommended testing procedure

We strongly recommend that you perform this performance testing before you deploy a new appliance into production.

1. Design the lab tests as follows:
   - Set up all of the workflows that you expect to run on the appliance.

- Obtain input that is a good example of the most complex source that you expect to handle. A complex source is one that has video that includes the following:
  - A lot of movement and change of scene.
  - Possibly a ticker tape or other strip of text.
  - Broad areas of the same color, like a grassy area, because high video quality of these areas is difficult to obtain.
- Set up the events with the most complex outputs that you expect to run on the appliance. Include any high-demand features that you plan to implement. See the section called “Features and performance”.
- Run the events for several days.

2. Monitor the events as described in this section:
   - Frequently measure the CPU usage, RAM usage, I/O bandwidth, and system bandwidth. See the section called “Measure performance”.
   - Continually monitor the logs that Elemental Live produces. See the section called “Log messages for performance”.
3. Revise the workflows in terms of density and video quality, and continue testing. You should revise the workflows incrementally.

Recommendation: Continually upgrade Elemental Live

We strongly recommend that you always upgrade to the latest version of Elemental Live. Improvements are continually being made to Elemental Live to increase the density on the appliances, and to enhance the video quality.

Assessing performance by measuring

There are several features of the appliance that affect performance:

- CPU usage
- RAM usage
- I/O bandwidth
- Memory bandwidth

You should look at all these features when measuring performance and attempting to increase density. For example, it is possible that your workflows aren't stressing the CPU capabilities, but they are making maximum demands on the I/O bandwidth.

Topics

CPU usage

CPU usage is the percentage of CPU power being used. CPU usage is affected primarily by the complexity of the output encode.

Measuring CPU usage from the command line

To measure CPU usage, use one of these utilities:

- `top`, which is the default command line utility in the Linux operating system.
- `htop`, which is a Linux utility that you must install.

Measuring CPU usage on the web interface

The status bar at the top of the Elemental Live web interface includes a CPU usage indicator. Hover over the indicator to view the percentage of CPU power that's being used.

Expected CPU usage for 4K workflows

4K workflows can be run only on specific L8xx appliances. Don't run 4K workflows on unsupported appliances.

When you monitor the CPU usage (using top or htop), you will notice that the CPU load is always below the maximum. When you monitor core usage (using htop), you will notice that the load for individual cores is also below the maximum. However, do not assume that you can increase density.

The CPU on these appliances is optimized for 4K workflows. As the event runs, there will be spikes in the CPU usage, due to the nature of a 4k workflow. Therefore, there must always be a buffer of CPU for these peaks.

You might be able to add one more 4K encode if the appliance is showing under the following CPU usage:

- 40% to 45% per CPU.

Expected usage for SD and HD workflows in L8xx appliances

With these workflows, the measured CPU usage is a good guide to the possible number of events. You might be able to add one or more events if the appliance is showing under the following CPU usage:

- 80–85% on GPU-based machines.
- 70% on CPU only encoders (L8xxx)

If you add more events, consider refining the encoding on the existing events. See the section called “Encoding and performance”.

RAM usage

RAM usage is the percentage of the memory being used on the appliance memory cards.

You can measure free memory using the `top` or `free` commands.

In Elemental Live versions 2.19.1 and later, the status bar at the top of the Elemental Live web interface includes a memory usage indicator.

Typically, RAM usage doesn't create performance problems on Elemental Live appliances. However, you should still monitor RAM usage to make sure that you always have enough free memory. If the appliance runs out of memory and starts to swap memory, there will be performance and video quality issues.

I/O bandwidth

I/O bandwidth is the rate at which the CPU reads from input or writes to output through a network interface card. Therefore, network inputs and outputs affect I/O bandwidth. SDI inputs don't affect I/O bandwidth.

Impact of I/O bandwidth issues

Lack of the required input bandwidth typically results in dropped frames in the output encodes.

You might not notice output bandwidth problems in Elemental Live. Instead, the downstream systems will notice the problem.

Measuring I/O bandwidth

- To measure I/O bandwidth on an Ethernet card, use the `iftop` command. This command shows the bandwidth for one card over a range of time (the last 2 seconds, 10 seconds, and 40 seconds). It also shows a running total of bandwidth used since you entered the `iftop` command.

  Enter the `iftop` command. For example, to show the results for `eth0`, enter this command:

  ```
  [elemental@encoder ~]$ sudo iftop -t -i eth0
  ```

  To stop collecting data, press **Ctrl-C**.

- To measure I/O bandwidth on a Mellanox card, use the `mlnx_perf` command.

  Following are the expected bandwidths on the Mellanox card:

  - Workflows with SMPTE 2110 inputs or outputs, and workflows with SMPTE 2022-6 inputs require a Mellanox card. This card has a declared maximum bandwidth of 25 Gbps. In practice, the maximum bandwidth (using the latest versions of Elemental Live) is as follows:
  - For SMPTE 2110 workflows, expect approximately 24.5 Gbps per Mellanox port.

Expected bandwidth for individual workflows

The following table illustrates the difference in bandwidth that's required for two different inputs (SMPTE 2110 and SMPTE 2022-6) with two different output resolutions (HD and 4K).

Notice that the 4K output requires approximately four times as much bandwidth as the HD output.

Type of input	Resolution of the output	Expected bandwidth (Gbps) for the workflow
SMPTE 2110 (video only)	HD (1080p60)	2.6
	4Kp60	10.5
SMPTE 2022-6	1080p60	3
	4Kp60	12

System bandwidth

System bandwidth is the rate at which network and data traffic moves between process on each core, and between those cores and memory.

Impact of system bandwidth issues

Typically, system bandwidth rates don't create performance problems on Elemental Live appliances.

One of the ways that you know there are memory bandwidth problems on the appliance is through error messages in the logs. For more information, see the section called “Log messages for performance”.

If you suspect that system bandwidth is causing these problems, reduce density on the appliance.

Measuring system bandwidth

The `amd_bandwidth` utility is included in Elemental Live versions 2.18.6 and later.

It shows the outbound bandwidth for each socket on the appliance. For example:

```
$ sudo amd_bandwidth
Collecting bandwidth data for 10 seconds...
Socket0 bandwidth: 7.43645 GB/s
Socket1 bandwidth: 19.0827 GB/s
```

Expected rates

The following guidelines apply for system bandwidth:

- Single-socket L8xx appliances have a maximum bandwidth of 90 GBps
- Dual-socket L8xx appliances have a maximum bandwidth of 140 GBps

Assessing performance with logging messages

Elemental Live continually writes to log files to capture status information about the health of each event.

When there are encoding problems, the first recourse of Elemental Live is to automatically lower the output quality. The next recourse is to drop or repeat frames. Elemental Live never chooses to slow down output, because then it would not be running in real time.

Viewing relevant logging messages

You can parse the log files for information that is useful for identifying density, speed, or quality issues.

For example, the following command pulls specific log messages:

```
grep -irE "SyncRepeat|low quality|dropped| C " /opt/elemental_se/web/log/10000/
```

This command performs a case insensitive (-i), recursive (-r) search for extended regular expressions (-E). The messages it finds are those in the following table. The command searches in the specified path on the appliance. Note that `10000` is the name of the folder where logs are stored.

Messages and their meaning

The following table lists the Elemental Live messages that relate to performance, and identifies possible causes of the message.

Message	Meaning	Possible performance issue
Low quality	The encoder is automatically lowering quality of encoding in an attempt to keep up with the speed at which frames are being delivered to the encoder.	CPU: Not enough available CPU power

Message	Meaning	Possible performance issue
Dropped	The encoder is dropping frames, which typically indicates that Elemental Live can't keep up with the rate at which it is receiving frames to encode.	CPU: Not enough available CPU power
SyncRepeat	The encoder is repeating frames.	CPU: Not enough CPU power to encode the current frame. The current video might be very complex. Or the CPU usage might generally be at its maximum. I/O: Not enough available I/O bandwidth Network: Elemental Live can't ingest the input. Perhaps there are network issues. One way that Elemental Live recovers from an input failure is through repeat frames.
C	A **C** beside a message identifie s a critical error.	

Dealing with issues

The solution to dropped or repeated frames is typically to decrease density on the appliance. You might also fine tune the video quality. See the section called “Encoding and performance”.

Encoding parameters that affect performance

You can tune the quality, speed, and density of the video encodes in each event.

CABAC

For information about this parameter, see the section called "Miscellaneous video tuning parameters".

Codec of the output encode

Some codecs make more demands on the CPU than other codes. Here is a comparison of the processing demands of codecs, from least to most demanding:

- JPEG XS
- MPEG
- AVC
- HEVC

For more information, see the section called "Video quality".

Density versus quality (SVQ)

You can modify the density and quality balance of each individual encode by setting the **Density vs Quality** field. You can favor quality at the cost of density, or you can favor density at the cost of quality.

For more information, see the section called "Miscellaneous video tuning parameters".

Encode order

If you are using a version of AWS Elemental Live before version 2.24.4, AWS Elemental Live assigns CPU resources to the encodes (streams) in the order in which they appear in the web interface or in the XML of the event. You should list your video encodes (streams) from highest resolution to lowest resolution. In this way, Elemental Live will first assign CPU resources to the most difficult encodes.

Starting with version 2.24.4, Elemental Live automatically assigns resources to the most difficult encodes.

Frame rate, resolution, bitrate, and color depth

A key factor in performance is the pixels per second being produced by all outputs in all events.

The following characteristics of each video encode affect pixels per second:

Characteristic	Field in the Video section of the Output section	Comment
Resolution	**Resolution**	
Frame rate	**Frame Rate**	
Bitrate	**Rate Control Mode**	See the section called "QVBR and rate control mode".
Color depth: 8-bit or 10-bit	Part of the profile of the codec: **Video Stream**, then **Advanced**, then **Profile**	

Group of pictures

For more information, see the section called "Group of pictures (GOP) configuration".

Lookahead

For more information, see the section called "Miscellaneous video tuning parameters".

Quantization

For more information, see the section called "Quantization controls".

Slices

This parameter affects CPU usage.

The number of slices in each video frame affects the CPU usage. You must balance the desired quality against the processing demands of that quality.

We recommend that you set the slices to automatic, to let Elemental Live set the value that works best for the video resolution. For more information, see the section called "Miscellaneous video tuning parameters".

Features that affect performance

Motion graphic overlays

Motion graphic overlays in the video output can add up to 50% density, compared to an output without overlays.

For more information, see the section called “Graphic overlay: Motion overlay”.

Color space conversions

Color space conversions add compute complexity and require more CPU usage.

For more information, see the section called “Color space”.

Noise reducer filters

Noise reducer filters add up to 10% density when compared to an output without noise reduction.

For more information, see the section called “Noise reduction”.

Working with Inputs

This chapter describes how to set up the different types of inputs that AWS Elemental Live supports.

Topics

Reference: Supported live inputs

The following tables describe the supported types of source content (input) that an AWS Elemental Live event can ingest. Read this section to find the types of source content that you must request from your content provider.

Topics

Supported types

This table describes the different live media types that Elemental Live can ingest, and the type of upstream systems that support each media type.

The rows are sorted by media type. The *Released* column identifies the version of Elemental Live that introduced the input type.

Media type	Upstream system	Use case	Type of input	Protocol	Released
HLS	HTTP or HTTPS server	Pull an HLS stream from an external endpoint using the HTTP protocol, with or without a secure connection.	HLS network input	http:// or https://	Before 2.14.0
MPTS	RTP host	Receive an MPTS or SPTS stream, using the RTP protocol or UDP protocol.	Network input	rtp:// or udp://	Before 2.14.0
RTMP Pull	RTMP server	Pull a stream using the RTMP pull protocol.	Network input	rtmp:// or rtmps://	RTMP supported before 2.14.0 RTMPS introduced in 2.22.2
RTMP Push	RTMP server	Receive a stream using the RTMP push protocol.	Network input	rtmp:// or rtmps://	RTMP supported before 2.14.0

Media type	Upstream system	Use case	Type of input	Protocol	Released
					RTMPS introduced in 2.22.2
RTSP	A device that supports RTSP	Receive an RTSP stream using the TCP or UDP protocol. RTSP streaming is popular with devices such as security cameras.	RTSP input	rtsp://	2.22.4
SMPTE 2110 with compressed video	RTP or UDP host that supports SMPTE 2110 sources	Receive a JPEG XS live stream that is compliant with SMPTE 2110. Redundant inputs using SMPTE 2022-7 are supported but optional.	SMPTE 2110 input	rtp://	2.21.3

Media type	Upstream system	Use case	Type of input	Protocol	Released
Transport stream	AWS Elemental MediaConn ect	Receive an encrypted or unencrypt ed transport stream (TS) from AWS Elemental MediaConn ect. The source into MediaConn ect uses the SRT protocol.	AWS Elemental MediaConn ect	Not applicabl e	2.21.1
Transport stream	AWS Elemental MediaConn ect	Receive an encrypted or unencrypt ed transport stream (TS) from an SRT listener output on AWS Elemental MediaConn ect Elemental Live is the SRT caller.	Secure Reliable Transport (SRT)	srt://	2.21.3

Media type	Upstream system	Use case	Type of input	Protocol	Released
Transport stream	RTP host	Receive a transport stream (TS), using the RTP protocol.	Network input	rtp:// or udp://	Before 2.14.0
Transport stream	RTP host that is compliant with SMPTE 2022-7	Receive a transport stream (TS) over RTP. This input type specifica lly supports redundant inputs using SMPTE 2022-7. Elemental Live expects to receive two sources.	SMPTE 2022-7 network input	rtp://	2.17.3
Transport stream	SRT caller	Receive a transport stream (TS) from an SRT caller. The content can be encrypted using AES.	Secure reliable transport (SRT)	srt://	2.21.1

Media type	Upstream system	Use case	Type of input	Protocol	Released
Transport stream	UDP host	Receive a transport stream (TS), using the UDP protocol.	Network input	rtp:// or udp://	Before 2.14.0
Uncompressed content – SDI	Standard SDI source device	Receive SD-SDI, HD-SDI, or 3G-SDI content. The Elemental Live appliance must be configured with an SDI interface.	One of the SDI options	Not applicable	Before 2.14.0
Uncompressed content – 4K SDI	12G SDI source device	Receive 4K SDI content using 12G. The Elemental Live appliance must be configured with a 12G SDI interface.	One of the SDI options	Not applicable	2.23.3

Media type	Upstream system	Use case	Type of input	Protocol	Released
Uncompres sed content – 4K Quadrant SDI	Quad-comp liant SDI source device	Receive 4K SDI content that is formatted as Quadrant. The Elemental Live appliance must be configure d with an 8-port SDI interface.	One of the Quadrant options	Not applicabl e	Before 2.14.0
Uncompres sed content – 4K 2SI SDI	2SI-compl iant SDI source device	Receive 4K SDI content that is formatted as 2SI (2 Sample Interleav e). The Elemental Live appliance must be configure d with an 8-port SDI interface.	Interleave 4K (HD-2SI)	Not applicabl e	Before 2.14.0

Media type	Upstream system	Use case	Type of input	Protocol	Released
Uncompressed content – 4K 2SI SDI	2SI-compliant SDI source device	Receive 4K SDI content that is formatted as 2SI (2 Sample Interleave). Receive the content over 12G. The Elemental Live appliance must be configured with a 12G SDI interface.	Interleave 4K (HD-2SI)	Not applicable	2.23.3

Media type	Upstream system	Use case	Type of input	Protocol	Released
Uncompressed content – SMPTE 2022-6	RTP or UDP host that supports SMPTE 2022-6 sources	Receive an uncompressed live SMPTE 2022-6 stream over UDP or RTP. Redundant inputs using SMPTE 2022-7 are supported but optional. Support for SMPTE 2022-7 was added in version 2.20.3.	SMPTE 2022-6 input	rtp:// or udp://	2.20.3

Media type	Upstream system	Use case	Type of input	Protocol	Released
Uncompressed content – SMPTE 2110	RTP or UDP host that supports SMPTE 2110 sources	Receive an uncompressed live stream that is compliant with SMPTE 2110. Redundant inputs using SMPTE 2022-7 are supported in version 2.20.3 and later.	SMPTE 2110 input	rtp://	2.17.3

Supported codecs

This table specifies the codecs that are supported for each input media type that Elemental Live supports.

Media Type	Video Codecs	Audio Codecs
HLS	H.264 H.265	AAC
MPTS	H.264 H.265 MPEG-2	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos

Media Type	Video Codecs	Audio Codecs
		MPEG-1, layer II PCM
RTMP	H.264	AAC
RTSP	H.264 H.265	AAC
Transport stream	H.264 H.265 J2K (only in a TS that is compliant with TR-01) MPEG-2	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos MPEG-1, layer II PCM
SDI	Uncompressed	Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos Dolby E frames carried in PCM streams tagged with SMPTE-337 PCM

Media Type	Video Codecs	Audio Codecs
HDMI	Uncompressed	Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos Dolby E frames carried in PCM streams tagged with SMPTE-337 PCM
SDI Quad-compliant SDI 2SI-compliant SDI	Uncompressed	Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos Dolby E frames carried in PCM streams tagged with SMPTE-337 PCM
Uncompressed SMPTE 2110	Uncompressed JPEG XS (starting with version 2.21.3)	Dolby Digital Dolby Digital Plus PCM
Uncompressed SMPTE 2022-6	Uncompressed	Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos PCM

Rules for ingesting Apple HLS TS sources

For video, each AWS Elemental Live event can extract only one video from only one rendition. Elemental Live will not reject inputs that contain multiple renditions, but it will handle only one of the renditions. There are fields in the event for specifying which video to extract.

For audio, each Elemental Live event can extract audio from the same rendition as the selected video. It can extract more than one audio from that rendition. It cannot extract audio from two different renditions. There are fields in the event for specifying which audio or audios to extract.

Elemental Live cannot extract audio from a rendition that contains only audio; it does not support ingest of audio rendition groups.

In all cases, the incoming HLS stream must include a manifest.

Rules for ingesting MPEG-TS programs

For video, each AWS Elemental Live event can extract only one video from only one program. Elemental Live will not reject MPTS inputs, but it will handle only one program. There are fields in the event that specify which video to extract.

For audio, each Elemental Live event can extract audio that is in the same program as the video. It can extract more than one audio from that program. It cannot extract audio from another program. There are fields in the event that specify which audio to extract.

Reference: Supported file inputs

The following tables describe the supported types of source content (input) that an Elemental Live event can ingest. Read this section to find the types of source content that you must request from your content provider.

Topics

Supported upstream systems for file inputs

This table describes the different upstream systems that can provide file content to Elemental Live.

The rows are sorted by type of input.

Upstream system	Use case	Type of input	Protocol	Push or pull?
Amazon S3 bucket	Pull a VOD asset from an Amazon S3 bucket, using a secure or unsecure connection.	File input	custom protocol s3:// or s3ssl://	Pull
Aspera server	Pull a VOD asset from a server that supports the Aspera data transfer protocol.	File input	Custom protocol aspera://	Pull
A directory on the Elemental Live appliance	Pull a VOD asset that is stored on a directory on the Live appliance.	File input	Not applicable	Pull
FTP or SFTP server	Pull a VOD asset from a server that supports FTP or SFTP.	File input	ftp:// or sftp://	Pull
SCP server	Pull a VOD asset from a server that supports SCP.	File input	scp://	Pull
Amazon S3 bucket	Pull an HLS VOD asset from an Amazon	HLS file input	custom protocol s3:// or s3ssl://	Pull

Upstream system	Use case	Type of input	Protocol	Push or pull?
	S3 bucket, using a secure or unsecure connection.			
HTTP or HTTPS server	Pull an HLS VOD asset from a server that supports HTTP or HTTPS.	HLS file input	http:// or https://	Pull

Supported containers

This table specifies the types of containers that are supported for file inputs. These containers apply to all the types of upstream systems.

Input type	Container	Media Type	Supported extensions for the file
File input	Adobe Flash	F4V	.f4v, .flv
File input	Apple HLS	HLS	.m3u8
File input	Audio Video Interleave	AVI	.avi, .divx, .xvid
File input	Matroska	MKV	.mkv
File input	MPEG Transport Streams	MPEG TS	.m2ts, .m2t, .mts, .ts, .trp, .mp
File input	MPEG-1 System Streams	MPEG SS	.mpg, .mpeg
File input	MPEG-4	MPEG-4	.mp4, .m4v, .f4v

Input type	Container	Media Type	Supported extensions for the file
File input	MXF	MXF	.mxf
File input	No Container	MPEG-1 or MPEG-2 video	.m2v, .m1v
File input	QuickTime	QuickTime	.mov
File input	WAV	WAV	.wav
HLS file input	Apple HLS	HLS	.m3u8

Supported codecs

This table specifies the codecs that are supported for each type of file input container.

Container	Media Type	Video Codecs	Audio Codecs
Adobe Flash	F4V	H.264	AAC
Apple HLS	HLS	H.264 H.265	AAC
Audio Video Interleave	AVI	Uncompressed DivX/Xvid DV/DVCPRO	Dolby Digital Dolby Digital Plus up to coding mode 7.1 Dolby Digital Plus with Atmos Dolby E frames carried in PCM streams tagged with SMPTE-337 MPEG-1, layer II

Container	Media Type	Video Codecs	Audio Codecs
			PCM
Matroska	MKV	H.264 MPEG-2 MPEG-4 part 2	AAC Dolby Digital Dolby Digital Plus up to coding mode 7.1 Dolby Digital Plus with Atmos
MPEG Transport Streams	MPEG TS	H.264 H.265 MPEG-2	AAC AIFF Dolby Digital Dolby Digital Plus up to coding mode 7.1 Dolby Digital Plus with Atmos Dolby E frames carried in non-PCM streams MPEG-1, layer II PCM

Container	Media Type	Video Codecs	Audio Codecs
MPEG-1 System Streams	MPEG SS	MPEG-1 MPEG-2	AAC AIFF Dolby Digital Dolby Digital Plus up to coding mode 7.1 Dolby Digital Plus with Atmos MPEG-1, layer II PCM
MPEG-4	MPEG-4	Uncompressed AVC Intra 50/100 DivX/Xvid H.262 H.264 JPEG 2000 MJPEG MPEG-2 MPEG-4 part 2	AAC Dolby Digital Dolby Digital Plus up to coding mode 7.1 Dolby Digital Plus with Atmos PCM

Container	Media Type	Video Codecs	Audio Codecs
MXF	MXF	Uncompressed AVC Intra 50/100 DV/DVCPRO DV25 DV50 DVCPro HD H.264 JPEG 2000 MPEG-2	AAC AIFF MPEG-1, layer II PCM Dolby E frames carried in PCM streams tagged with SMPTE-337
No Container		DV/DVCPRO H.264 H.265 MPEG-1 MPEG-2	

Container	Media Type	Video Codecs	Audio Codecs
QuickTime		Uncompressed Apple ProRes AVC Intra 50/100 DivX/Xvid DV/DVCPRO H.262 H.264 JPEG 2000 MJPEG MPEG-2 MPEG-4 part 2	AAC
WAV	WAV	None	Dolby E frames carried in PCM streams tagged with SMPTE-337

Interleave 4K inputs

AWS Elemental Live supports 4K uncompressed sources that are formatted as 2SI (2 Sample Interleave).

To configure an event for this input, do the following:

1. Make sure that the source is formatted as 2SI.
2. In the **Input** section of the event, set the **Select Type** field to **Interleave 4K (HD-2SI)**.

The **Interleave 4K (HD-2SI)** setting is applied to fours ports on the appliance: ports 1-4, or ports 5-8, or ports 9-12, or ports 13-16.

When you start the event, the **Media Info** in the output will report the input as either 2SI or SDI.

Note

If you choose the **Interleave 4K** input type, you must make sure that the source is formatted as 2SI, not as Quadrant. If you mismatch the source format and the input type, Elemental Live will ingest the source, but there will be readily observable video issues.

Ingesting SMPTE 2022-6 content

AWS Elemental Live supports video sources that are compliant with the SMPTE 2022-6 standard.

Topics

- Appliance hardware requirements
- Supported content
- Get ready: Remove bonded interfaces
- Get ready: Reserve cores for SMPTE 2022-6
- Setting up a SMPTE 2022-6 input with or without SMPTE 2022-7

Appliance hardware requirements

A SMPTE 2022-6 input requires an Elemental Live appliance with a high-speed network interface card (NIC). Therefore, to set up a SMPTE 2022-6 input, you must create the event on one of the following appliances.

Appliance	Network interface card (NIC)
L800 series Elemental Live appliance	25 GbE NIC
A bare-metal appliance	25 GbE Mellanox NIC. You must make sure that the NIC is licensed for use with the RiverMax SDK.

Supported content

With SMPTE 2022-6, the video, audio, and ancillary data are muxed into one feed. Compare this design to SMPTE 2110, where the content is each in a separate essence.

The following table describes the content that Elemental Live supports in SMPTE 2022-6 inputs.

Type	Details
Video	Uncompressed Resolutions – SD and HD Scan types – Progressive and interlaced Sampling – 4:2:2 Bit format – 10-bit
Audio	PCM audio Uncompressed Sample rates: 44.1kHz and 48.0 kHz
	Dolby Digital (AC3) Coding modes – 1.0, 1+1, 2.0, 3.2 (with LFE)
	Dolby Digital Plus (EAC3) Coding modes – 1.0, 2.0, 3.2
Ancillary data – Captions (optional)	EIA-608 embedded captions CEA-708 embedded captions Teletext as OP42 teletext format. SMPTE 2031 field is unchecked in source.

Type	Details
	Teletext as OP47 teletext format wrapped in a SMPTE-2031 envelope. SMPTE 2031 field is checked in source. Teletext as OP47 teletext format, also known as SMPTE RDD-08 (compliant with ITU-R BT.1120-7). SMPTE 2031 field is unchecked in source.
Ancillary data – Ad avail messages (optional)	SCTE 104 messages. Elemental Live will automatically convert these messages to SCTE 35 messages during ingest.

Get ready: Remove bonded interfaces

Read this section if you plan to implement redundant outputs with SMPTE 2022-6.

SMPTE 2022-6 inputs support resiliency, but they do so using SMPTE 2022-7. You can't implement input resiliency by bonding the two interfaces on the appliance. This resiliency implementation has the following impact:

- If you currently have a bonded interface on the appliance (on a Mellanox card or on a 25Gb card), you must remove the bond.
- In addition, if you have non-SMPTE 2022-6 (or non-SMPTE 2110) events set up on the appliance that use this bonded interface, you must modify the event configuration. You might be able to bond other interfaces on other cards in the appliance, and then use that bonded interface in those events. If you can't do that, you must change the event configuration to not use a bonded interface.

If you don't plan to implement redundant SMPTE 2022-6 inputs (or SMPTE 2110 inputs or outputs), you can retain the bonded interface on the appliance.

Get ready: Reserve cores for SMPTE 2022-6

Before you can run an event that has SMPTE 2022-6 inputs, you must reserve cores on the appliance NIC. When you reserve these cores, Elemental Live uses them only for processing SMPTE 2022-6 and/or SMPTE 2110.

> **Important**
>
> Ensure that no events are running before you follow these steps.

To enable an interface for SMPTE 2022-6

1. Find out which Ethernet interfaces on the appliance apply to your NIC. For example, eth4 and eth5.
2. Stop all events that are currently running on the appliance.
3. In the Elemental Live web interface, go to **Settings**, and select **Network Devices**.
4. Choose the **edit** icon (pencil) next to the device that you want to enable. For L800 series appliances and bare metal appliances, this can only be enabled for 25 GbE NICs.
5. Select the **SMPTE 2110 and SMPTE 2022-6 Enabled** check box, and then choose **Save**.
6. Stop and restart the service for your changes to take effect. You can do this in the web interface or in the command line interface (CLI).

    - In the web interface, go to the **Settings** tab, select **Stop Service** and then select **Start Service**.

      OR
    - In the CLI, run `sudo service elemental_se restart`

Setting up a SMPTE 2022-6 input with or without SMPTE 2022-7

You can create a SMPTE 2022-6 input in Elemental Live in order to ingest a source that is compliant with the SMPTE 2022-6 specification.

You can optionally configure the input to use SMPTE 2022-7, if the source offers it. This standard describes a resiliency solution that uses seamless protection switching. *Seamless protection switching* refers to a method of transmission in which two identical packet streams are transmitted over separate network routes at the same time. If packets are lost from the first stream, the second stream can be used to reconstruct the data. This switching between the two streams is instantaneous, so the content is not affected. This is what makes it seamless.

To set up a SMPTE 2022-6 input

1. From the **Input** menu in your event, select **SMPTE 2022-6 Input**.
2. Complete the following fields:

   - In **Network Location**, provide the UDP or RTP network location.
   - For **Interface**, enter the network interface to use for this particular stream. For example, `eth4`.

     > **Note**
     >
     > The network interface must be on a 25 GbE Mellanox card.

   - In **IGMP Source**, provide the source-specific multicast streams. Only packets from the specified source are processed. All others are discarded.
3. If you are implementing seamless protection switching (SMPTE 2022-7), complete the following fields:

   - **Secondary Network Location**
   - **Secondary Interface**. The secondary interface must be different from the primary interface
   - **Secondary IGMP Source**

Handling ancillary data in SMPTE 2038

Whenever you plan to set up a transport stream (TS) source, you should find out if the content provider has set up the TS so that some data is included in SMPTE 2038 packets, rather than in the standard locations. The content provider typically sets up the TS in this way because they have converted an SDI stream to a TS.

If the content provider has provided data in SMPTE 2038, set up the input to extract that data from the SMPTE 2038 packets. Doing so ensures that Elemental Live will handle the source correctly. Don't ignore the data.

Topics

Supported ancillary data

Elemental Live can ingest several types of ancillary data. After the data has been ingested, Elemental Live can handle it in one of two ways—use it, following the instructions you set up in the channel, or passing it through in the output, so that a downstream system can use it.

Elemental Live can use the following ancillary data:

- Time code – If a time code is included in the SMPTE 2038, it is always an embedded time code.

  You can choose to configure the event to use the extracted time code as the timecode source for video processing.
- Captions – The SMPTE 2038 might include embedded, Teletext, or ARIB captions.

  You can choose to set up the extracted captions as the source for output captions, in the usual way.
- AFD signals – The SMPTE 2038 might include AFD signals.

  You can choose to set up one or more outputs so that Elemental Live uses the signals to modify the video.
- SCTE 104 messages – The SMPTE 2038 might include SCTE 104 messages. Elemental Live automatically converts the messages to SCTE 35 messages.

  You can choose to handle these messages in the usual way.

Elemental Live can pass through the following ancillary data:

- Custom data – Elemental Live considers ancillary data to be *custom data* if it is not any of the four types listed above.

  You can choose to identify this custom data so that Elemental Live can extract it.

You can then set up SMPTE 2110 outputs to include that data. Elemental Live doesn't read or use the ancillary data in any way.

Well-formed SMPTE 2038 source

For Elemental Live to handle the ancillary data, the SMPTE 2038 must meet certain criteria:

- The SMPTE 2038 packet must be present in every PMT.
- The PID in which the SMPTE 2038 packet is located must not change in the stream. There is no support for changing the PID and sending a new PMT identifying that PID.
- The stream should contain the SMPTE 2038 packet in only one PID. If it is present in more than one PID, there is no guarantee that Elemental Live will identify the PID that appears first. It could choose another PID, with results you do not intend.

Enable SMPTE 2038

You must set up the input to ingest the SMPTE 2038 ancillary data.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To set up the event to look for ancillary data in SMPTE 2038

Follow these steps for both types of handling—extract and use, and passthrough.

1. Speak to the content provider to obtain information about the SMPTE 2038 ancillary data in the source:

    - Find out which of the four types of ancillary data are in the SMPTE 2038. If the ancillary data includes captions, find out which format of captions is present.

    - Find out whether the SMPTE 2038 includes custom data. Elemental Live can ingest custom data only if it is set up in a DID/SDID pair. It can't ingest data that's set up in a DID/DBN pair.

Obtain the values of the DID and SDID for all the custom data pairs.

Obtain a description of the data in each pair.

2. On the Elemental Live web interface, display the details for the event that you want to set up.
3. In **Input**, make sure you have set the **Input type** field to one of the following. These types are all valid for TS inputs:

   - **Network Input**
   - **File Input**
   - **HLS File Input**
   - **HLS Network Input**
   - **SMPTE 2022-7 Network Input**

   The **Prefer SMPTE 2038** field appears in the **Advanced** section of the **Input** section.
4. In **Input** > **Advanced**, set **Prefer SMPTE 2038**:

   - **Checked** – You should choose this option if the source content contains SMPTE 2038 ancillary data. If the content provider has included SMPTE-2038, they intend for you to use it.
   - **Unchecked** – If you do not want Elemental Live to look at the SMPTE 2038, uncheck this field. Choose this option if the source content doesn't include SMPTE 2038 ancillary data. Elemental Live looks for ancillary data in the native TS. Even if a SMPTE 2038 PID is present, Elemental Live ignores that PID.

Setting up the event to use the ancillary data

After you have enabled SMPTE 2038, you should specify how you want Elemental Live to use the timecode, captions, AFD signals, and SCTE 104 messages that it detects.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

- For timecode, set the **Timecode Source** (in **Video Selector**) using one of the following options:
  - If the SMPTE 2038 contains a timecode and you want to use it – Set the source to **Embedded**. Elemental Live will extract the timecode from the SMPTE 2038, and never from the native TS.
  - Choose another value in order to use a different timecode source, such as system clock. These other timecode sources are never found in the SMPTE 2038.
- For captions, in **Input** > **Caption Selector**, set up the captions in the usual way.
  - If you don't want to use the captions from the SMPTE 2038 – Don't create any caption selectors that specify the captions of that type.
  - If you want to use the captions from the SMPTE 2038 – Create a caption selector that specifies that type of captions—**Embedded**, **Teletext**, or **ARIB**. Elemental Live will extract those captions from the SMPTE 2038. It won't look for those captions in the native TS.

    Perform the setup of the captions in the output in the usual way. For more information about working with captions, see Working with captions.
- For AFD signals, you can choose to use the AFD signals to modify the video:
  - If you don't want to use the signals – Set **Respond to AFD** and **Insert AFD signaling** to **None**.
  - If the SMPTE 2038 contains AFD signals and you want to use them – In **Output** > **Video Stream** > **Advanced** set the **Respond to AFD** field and the **Insert AFD signaling** field in the usual way. For information about the fields, click the question mark icon for the field.
  - If the SMPTE 2038 doesn't contain AFD signals, Elemental Live ignores the values in the two fields.
- For SCTE 104 messages, Elemental Live automatically converts the messages to SCTE 35 messages.
  - If you don't want to include the SCTE 35 messages in the output – There is nothing you need to do because omitting the messages is the default behavior.
  - If you want to work with the messages – See the section called “SCTE-35 and SCTE-104 message processing”.

Setting up the event to pass through custom data

After you have enabled SMPTE 2038, you can identify the custom data to extract, and then set up to include that data in your SMPTE 2110 outputs.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To pass through custom data

1. Decide which custom data you want to pass through to the SMPTE 2110 outputs. Make a note of the DID/SDID values.
2. On the Elemental Live web interface, display the details for the event that you want to set up.
3. In the **Input** section of the web interface, find the input that contains the SMPTE 2038. Find the **Add Custom DID/SDID Pair** button below the **Prefer SMPTE 2038** field. The button appears only if you selected the **Prefer** field.
4. Click the button and enter the appropriate values in the DID and SDID fields that appear. Repeat to add more pairs.
5. In the web interface, find the SMPTE 2110 output where you want to pass through the custom data. Find the ancillary output for that output. If you are not sure how to set up an ancillary output for SMPTE 2110, see the section called "Step 3: Create output group".
6. In the **Ancillary Data Settings** section, select **SMPTE 2038**.

   Elemental Live will extract each custom data pair that you identified in the input, convert it to a SMPTE 291 packet, and include it in the ancillary data stream of the SMPTE 2110 output.

Ingesting SMPTE 2110 content

You can create a SMPTE 2110 input in AWS Elemental Live in order to ingest a source that is compliant with the SMPTE 2110 specification. You can optionally configure the input to use SMPTE 2022-7, if the source offers it.

You can configure the SMPTE 2110 input by specifying an SDP file that contains information about the SMPTE 2110 content. Or you can configure the input to use NMOS to provide that information.

For more information about SMPTE 2110 video content, SMPTE 2022-7, NMOS, and SDP files, see the section called "SMPTE 2110 inputs and outputs".

Topics

- Get ready
- Setting up a SMPTE 2110 input using NMOS
- Setting up a SMPTE 2110 input without NMOS

Get ready

Before you set up a SMPTE 2110 input in an event, read the following information:

- the section called “Setup: Remove bonded interfaces”
- the section called “Setup: Reserve processing cores”
- the section called “Setup: Enable PTP”

Setting up a SMPTE 2110 input using NMOS

Follow this procedure if you have a SMPTE 2110 source and your organization uses NMOS IS-04 and IS-05.

> **Note**
>
> This section assumes that you have read the section called “SMPTE 2110 inputs and outputs” and are familiar with how SMPTE 2110 works and with its prerequisites.

You must enable NMOS on the appliance, as a one-time action. Then the setup of a SMPTE 2110 source is a two-step process. First, you create a 2110 receiver group to replicate the SMPTE 2110 receivers in Elemental Live. Then in the event, you create an input that uses the 2110 receiver group. In this way, a SMPTE 2110 source is similar to an SDI source, which is represented in Elemental Live by an SDI device that is used by an SDI input.

Topics

- Enable NMOS
- Obtain information from the NMOS operator
- Determine the SDPs to create
- Create the receiver group
- Create a receiver group input

- Configure the resiliency scenario

Enable NMOS

You must enable NMOS on your appliance. If you are using an NMOS registry, you must configure AWS Elemental Live to connect to it. Perform this configuration once, before you create the first SMPTE 2110 NMOS input.

You perform this step once on each appliance where you will use SMPTE 2110 NMOS inputs.

1. In the Elemental Live web interface, choose **Settings**. (Don't choose **Input Devices** or **Routers** from the submenu).
2. Complete the fields as described in the following table, then choose **Save**.

Field	
Enable NMOS Access	Choose **Yes**
Use make-before-break	The value that you enter here applies to all the SMPTE 2110 receiver groups in events on the node. There are two ways for Elemental Live to process receiver patching instructions from the NMOS controller: • Make before break: Elemental Live will prepare all the streams in the receiver group before it switches to ingesting this new content. This option requires more network bandwidth because Elemental Live must simultaneously process the original content and prepare the new content. • Break before make: Elemental Live will immediately switch to the new content. There will be a slight discontinuity in the

Field	
	content while Elemental Live makes the switch.
NMOS Node and Device Label **NMOS Node UUID** **NMOS Device UUID**	These fields must uniquely identify this node on the NMOS registry. You can enter values or leave the fields blank to let Elemental Live generate values. The combination of label UUID for the node must be unique on the NMOS registry. The combination of device label and UUID must be unique on the NMOS registry. When you create a SMPTE 2110 receiver group on this node, the values will be included in the data that Elemental Live sends to the NMOS registry.
NMOS Registry Server URI	Obtain the URI of the NMOS registry from your organization’s registry operator. If your organization doesn't use an NMOS registry, leave the auto-populated default value for this field in place.

Obtain information from the NMOS operator

Speak to the NMOS operator in your organization to make sure that you are aligned on the content of the streams.

- Obtain the number and types of the streams — video, audio, and so on.
- For each audio stream, obtain the languages and order of the languages. For example, in an audio stream that contains three languages, you must know the names and the order of the languages. The languages are assigned an index, starting at 0.
- For each embedded captions stream, obtain the number and order of the languages.

You must be aligned so that you can set up Elemental Live to select the audio and ancillary streams correctly, and so that the NMOS controller can successfully perform NMOS patching.

Example 1

You and the NMOS operator might agree that the SDP files and their streams are ordered as follows:

- SDP 1: Video stream
- SDP 2: Audio stream containing English Dolby Digital
- SDP 3: Audio stream containing French Dolby Digital
- SDP 4: Audio stream containing Spanish Dolby Digital
- SDP 5: Ancillary stream containing EIA-608 embedded captions, with English in CC1, French in CC2, Spanish in CC3, and CC4 not used.
- SDP 6: Ancillary stream for SCTE-104 messages.

Example 2

You and the NMOS operator might agree to this setup:

- SDP 1: Video stream
- SDP 2: Audio stream containing three audio descriptions, in this order:
  - English Dolby Digital
  - French Dolby Digital
  - Spanish Dolby Digital
- SDP 3: Ancillary stream containing EIA-608 embedded captions, with English in CC1, French in CC2, and Spanish in CC3. CC4 is not used.

Determine the SDPs to create

Determine the number of SDPs you will create in Elemental Live. Here are the rules:

- The input must include one an only one video SDP.
- Plan to create one SDP for each audio stream you want to extract. Following example 2 above, you must create three audio SDPs.
- Plan to create one SDP for the captions ancillary stream (if any).

- Plan to create one SDP for the SCTE-104 stream (if any).

Create the receiver group

You must create the SMPTE 2110 receiver groups that you need. Each receiver group identifies the set of individual streams of video, audio, and ancillary data (captions and SCTE 104 messages) that you want to treat as one input.

1. In the Elemental Live web interface, hover over **Settings** and choose **Input Devices**. Scroll down to the **SMPTE 2110 Receiver Groups** section.
2. Choose **Create SMPTE 2110 Receiver Groups**. Fields appear.
3. In **NMOS SMPTE 2110 Input**, enter a name for the receiver group. This name will appear in the dropdown list of inputs when you create an Elemental Live event or Conductor Live profile.
4. Adjust the remaining fields to match your requirements. The receiver group must include one video stream. But the audio and ancillary streams are optional.
   - Click **Add Audio SDP** to add one or more rows for audio streams. Or click **Remove** if you don't want any audio in the input.
   - Click **Add Ancillary SDP** to add one or more rows for ancillary streams (captions and SCTE-104 streams).

   Following example 1 above, you will three audio SDPs, and two ancillary SDPs.

   Following example 2 above, you will one audio SDP, and two ancillary SDPs.
5. Complete each stream row as follows:
   - **Label**: Enter a description. For example, in the video stream, enter **`vid_rx`**.

     Elemental Live will combine the receiver group name and the label to create a name for the stream that is unique to that system.
   - **Interface**: Enter the network interface for this stream to connect to the Elemental Live node. For example, **`eth4`**.
   - **Secondary Interface**: If you want to implement SMPTE 2022-7 , enter the interface for the secondary source. For example, **`eth5`**.
   - **GUID**: This field is auto-populated with a system-generated value. You might want to change this value, for example, to make it match values you created outside of Elemental Live. To change the value, choose **Show Advanced** and enter the new value.

6. Follow this step only if you are using Conductor Live. You must import the SMPTE 2110 receiver group input into Conductor Live.

   - On the primary Conductor Live web interface, choose the **Cluster** page, then choose **Nodes**.
   - Choose the down arrow beside the node and select **Import Devices**.

   You will now be able to use this input when you create a channel in Conductor Live. The names of all the imported SMPTE 2110 inputs will appear in the list of devices.

Create a receiver group input

You must create a receiver group input that connects to the appropriate SMPTE 2110 receiver streams.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating a Elemental Live event or Conductor Live channel.

1. From the **Input** menu in the Elemental Live event or Conductor Live profile, select the receiver group that you created. The list of receiver groups appears at the top of the dropdown list.

   Or if you are using AWS Elemental Conductor Live, in the **Input Type** field on the Conductor Live profile, select the receiver group that you created.

   > **Note**
   >
   > Don't select **SMPTE 2110 Input**, which is an option that appears at the bottom of the list.

2. Click **Advanced** to show more fields for this input. Scroll down to the selectors section of this input. Complete the selectors as follows:

   - **Video Selector**: Leave the single selector that appears by default. Ignore the **Program** and **PID** fields, which don't apply to SMPTE 2110 NMOS inputs. Complete the other fields as applicable.

- **Audio Selector**: Click **Add Audio Selector** to display as many selectors as you need. For example, following the examples in the section called "Step 4: Create receiver group", you need three selectors.

  If the receiver group has more than one audio SDP, you must complete these fields:

  - **Selector Type**: Choose **Track**.
  - **Track**: Enter the track. Following the first example in the section called "Step 4: Create receiver group", enter **1** to identify the English audio, **2** to identify the French audio, and **3** to enter the Spanish audio. It doesn't matter if the audios are all in the same SDP or different SDPs, the audio streams are always numbered in SDP order, starting from 1.

3. Click **Add Caption Selector** once. Make sure **Source** specifies **Embedded**. Ignore the other fields on this line, they don't apply to passing through embedded captions.

Configure the resiliency scenario

You can configure the receiver groups and inputs in an Elemental Live event or Conductor Live profile to support three scenarios. Note that all of the scenarios include NMOS patching, because you must always set up inputs for patching.

- Scenario A: NMOS patching only. With this setup, the NMOS controller can send requests to Elemental Live that patch SDP files with new content. Elemental Live switches to processing the new content.
- Scenario B: NMOS patching with hot backup. This scenario combines the patching from scenario A with support for Elemental Live hot backup. Hot backup lets Elemental Live automatically fail over from a failed input to another input.
- Scenario C: NMOS patching with hot backup and network redundancy. This scenario combines scenario A with hot backup and the network redundancy that is set up between the upstream system (the content source) and Elemental Live.

Topics

- Scenario A: Supporting NMOS patching
- Scenario B: Supporting NMOS patching and failover
- Scenario C: Supporting NMOS patching and network failover

Scenario A: Supporting NMOS patching

With this setup, the NMOS controller can send SDP content that provides patching instructions to Elemental Live. For example, in a curling match, it can send a patch that switches the source from the camera on the curling rink to the camera in the studio.

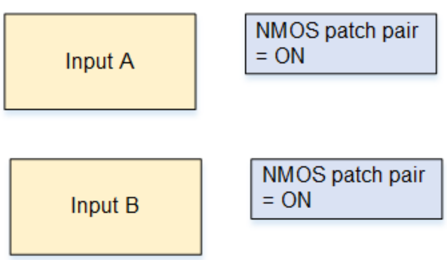
To support NMOS patching, you must attach the same Elemental Live receiver group to two Elemental Live inputs. Each pair of Elemental Live receiver group inputs is a *patching pair*. You can set up more than one patching pair in the Elemental Live event or Conductor Live profile.

To create each patching pair, follow this procedure.

1. In the Elemental Live event or Conductor Live profile that you are creating, create one SMPTE 2110 NMOS input (input A), as described in the section called “Step 5: Create the receiver group input”. In the **Advanced** section, set **NMOS patching pair** to **ON**.
2. After input A, create another input (input B) and select the same source item from the dropdown. In this way, you attach the same receiver group to both inputs. Set **NMOS patching pair** to **ON**.

Result of this setup

Elemental Live sets up a *patching pair* consisting of the first patching-pair-enabled input (input A in this example) and the next input in the list that is patching-pair-enabled (input B in this example). Whenever Elemental Live is ingesting this receiver group content, one of the inputs in the patching pair is *active* and the other input is in *standby*.



The two patching-pair inputs must be next to each other. If necessary, use the up and down arrows on the far right of the web interface to move an input up or down the list.

How it works at runtime

Assume that the NMOS controller sends a patching request by sending new SDP content for the receiver group that is attached to these two inputs. When Elemental Live receives the request, it sets up the standby input (input B, for example) with the new content, then switches from the active input (input A) to the standby input (input B). Input B becomes the active input. Each time Elemental Live receives a patching request that references this receiver group, it switches from the currently active input to the standby input.

Scenario B: Supporting NMOS patching and failover

This scenario adds failover to the patching capabilities of scenario A. First read the information about scenario A.

Elemental Live can automatically fail over to a third input when the active input (from the A/B patching pair) fails. Elemental Live fails over to a hot backup input that you define (input X). You define failure conditions for the first input in the patching pair, which is input A in our example. When these conditions are present, Elemental Live will fail over to input X when there is a problem with either input in the A/B patching pair. You must also define failback conditions in input X to trigger fail back to the A/B patching pair.

Follow this procedure:

1. In the Elemental Live event or Conductor Live profile that you are creating, create one SMPTE 2110 NMOS input (input A), as described in the section called “Step 5: Create the receiver group input”. In the **Advanced** section, set **NMOS patching pair** to **ON**.
2. After input A, create a backup input (input X). There is no requirement for input X to be a SMPTE 2110 input. For example, it might be a file input that displays a slate. A file input is particularly useful if the cause of the input failure is a network failure, because Elemental Live can switch to a file that is stored locally on the node.

   Make sure input X is the first input immediately after input A. To move input X up or down the list of inputs, use the up and down arrows on the far right of the web interface.
3. After input X, create another input (input B) and select the same source item that you already attached to input A. Set **NMOS patching pair** to **ON**.
4. Make these changes in input A:
   - Set **Hot Backup** to **ON**. Ignore the **Error Clear Time** and **Failback Rule** fields.
   - Click **Add Failover Condition** to create as many failover conditions as you want. To create one condition, click **Add Failover Condition**. In **Description**, choose the type of condition, for example **Input Loss**. In **Duration**, enter the length of time the condition must continue before the condition triggers a failover to input X.
5. Make these changes in input X:
   - Set **Hot Backup** to **ON**.

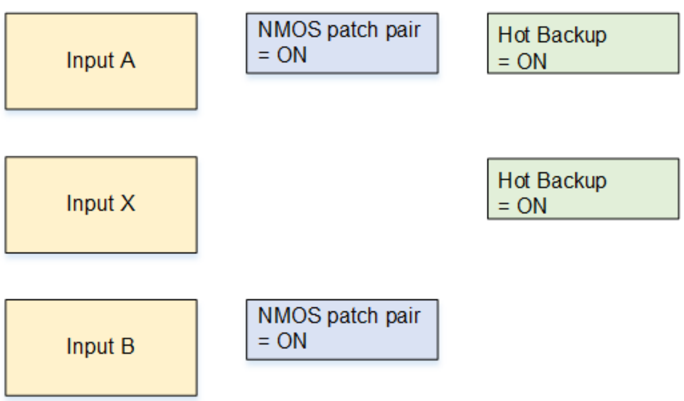
- Enter a time in **Error Clear Time**. After all the failover conditions are no longer applicable Elemental Live waits for the specified time before it fails back to input A.
- Choose a **Failback Rule** to specify how Elemental Live fails back to input A.

Don't enable hot backup on input B. Don't enable NMOS patching pair on input X.

Result of this setup

You now have three inputs in the order A, X, B.

- Patching pair: Inputs A and B each have NMOS patching pair enabled, therefore they are a patching pair, even though they are not next to each other.
- Hot-backup pair: Input A and input X both have hot backup enabled. Input A is set up with failover conditions. Input X is set up with failback rules. Therefore, inputs A and X are a hot-backup pair.



Scenario C: Supporting NMOS patching and network failover

You can set up inputs in a configuration that combines NMOS patching with hot backup and network redundancy. In this scenario, the hot backup configuration allows failover between the redundant networks that are set up between the upstream system (the content source) and Elemental Live. Network redundancy lets Elemental Live fail over to sources that are being delivered over a different network, if the main network fails. Network redundancy lets you fail over to a different live source, instead of failing over to a slate (as shown in scenario B).

Assume that the event is a curling game with two cameras, one on the curling rink and one in the studio. You have set up your upstream system with the following sources:

- Source 1: Content from the camera on the curling rink, routed over network 1.
- Source 2: Content from the camera on the curling rink, routed over network 2.
- Source 3: Content from the studio, routed over network 1.
- Source 4: Content from the studio, routed over network 2.

Setup steps

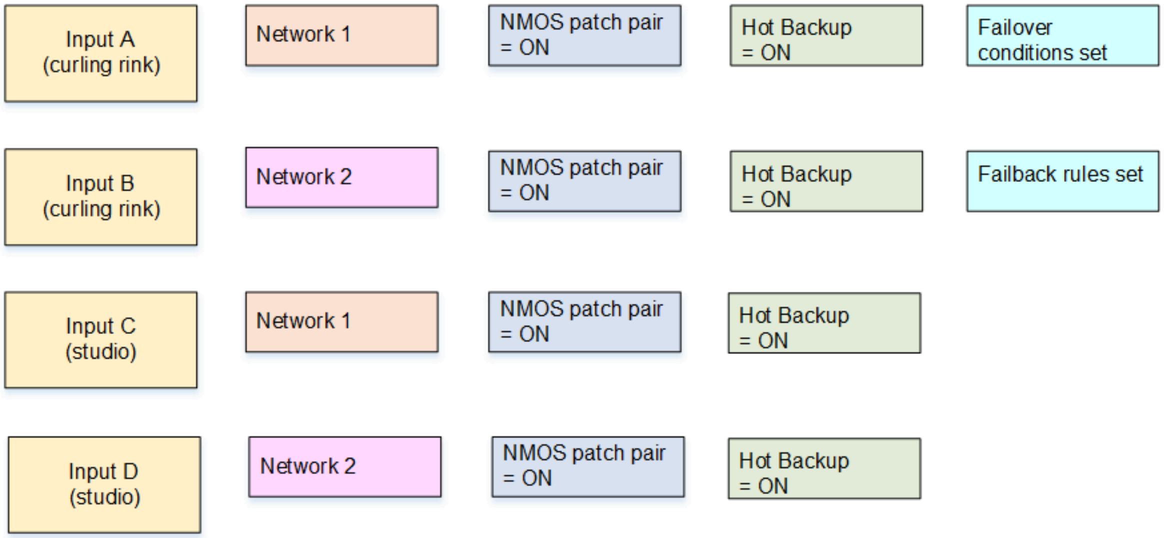
To set up SMPTE 2110 sources in this way, follow this procedure.

1. Create four receiver groups, one for each source. To create a receiver group, see the section called “Step 4: Create receiver group”.
2. In the Elemental Live event or Conductor Live profile that you are creating, create four inputs. Set up each input with the receiver groups, in the following order:
   - Input A uses the receiver group for source 1.
   - Input B uses the receiver group for source 2.
   - Input C uses the receiver group for source 3.
   - Input D uses the receiver group for source 4.
3. Turn on **NMOS patching pair** in all these inputs.
4. Turn on **Hot Backup** in all these inputs.

   You have now set up the four inputs as shown in the diagram after these steps.
5. Make these changes in input A:
   - Create as many failover conditions as you want. To create one condition, click **Add Failover Condition**. In **Description**, choose the type of condition, for example **Input Loss**. In **Duration**, enter the length of time the condition must continue before the condition triggers a failover to input C.
   - Ignore the **Error Clear Time** and **Failback Rule** fields.
6. Make these changes in input B:
   - **Error Clear Time**: Enter a time. After all the failover conditions are no longer applicable Elemental Live waits for the specified time before it fails back to input A.
   - **Failback Rule**: Choose a rule to specify how Elemental Live fails back to input A.
7. On inputs C and D, *don't* set up failover conditions, error clear times or failback rules.

Result of the setup

You now have four inputs in the order A, B, C, D. All inputs have hot backup enabled and all have patching enabled.



Elemental Live proceeds as follows:

- Elemental Live sets up a hot-backup pair.

  Inputs A and B have failover conditions and failover rules set, so Elemental Live considers them to be a hot-backup pair.
- Elemental Live next identifies sets up the patching pairs. It sets up input A with input C, and sets up input B with input D. Inputs C and D have hot backup enabled. But they aren't standard backups. Instead, they shadow the A/B input patching pair. The failover conditions for input A apply to input C, and the failback conditions of input B apply to D.

To summarize:

- Input A. Its hot backup is input B. Its patching pair is input C.
- Input B. Its hot backup is input A. Its patching pair is input D.
- Input C. Its hot backup is input D. Its patching pair in input A.
- Input D. Its hot backup is input C. Its patching pair is input B.

How failover and patching work at runtime

The following table describes how Elemental Live handles input failures and patching requests, depending on which input is currently active. Read across each row.

Current active input	If the active input failure occurs	If a patching request is received
Input A	If input A fails, Elemental Live fails over to input B (so it switches to the other network).	Elemental Live stops ingesting input A and starts to ingest input C.
Input B	If input B fails before the failback rules come into effect, Elemental Live follows the standard input failure behavior, which means it will repeat frames and so on, then finally display a slate.	Elemental Live stops ingesting input B and starts to ingest input D.
Input C	Elemental Live fails over to input B or input D, depending on which upstream source is currently flowing into Live.	Elemental Live stops ingesting input C and starts to ingest input A.
Input D	If input D fails before the failback rules come into effect, Elemental Live follows the standard input failure behavior, which means it will repeat frames and so on, then finally display a slate.	Elemental Live stops ingesting input D and starts to ingest input B.

Setting up a SMPTE 2110 input without NMOS

Follow this procedure if you have a SMPTE 2110 source and your organization doesn't use NMOS IS-04 and IS-05.

> **Note**
>
> This section assumes that you have read the section called "SMPTE 2110 inputs and outputs" and are familiar with how SMPTE 2110 works and with its prerequisites.

When your organization doesn't use NMOS, you must provide information about the SDP that applies to this source, so that Elemental Live can extract content correctly. For example, you must specify where the SDP file for the video is located, and identify the specific video streams to ingest. In addition, if you are using seamless protection switching, you must configure two interfaces for traffic.

Elemental Live supports ingest of SMPTE 2110 sources that implement SMPTE 2022-7. In this case, you can configure the input to ingest one or both instances of the source.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To set up a SMPTE 2110 input

1. From the **Input** menu in the event, select **SMPTE 2110 Input**.
2. In **Video SDP Location**, enter the location of the SDP file for the video. For example:

   http://172.18.8.19/curling_video.sdp

   For more information about SDP files, see the section called "About SDP files".
3. In **Media Index**, enter the index of the video stream that you want Elemental Live to extract. For example, enter `0` if the video is the first stream, **1** if it's the second stream, and so on.
4. Complete the **Interface** fields:

- **Interface**: Enter the network interface for this stream to connect to the Elemental Live node. For example, **`eth4`**.
- **Secondary Interface**: If you want to implement SMPTE 2022-7 , enter the interface for the secondary source. For example, **`eth5`**.

   In either case, choose the interface carefully. In Elemental Live, network interfaces are shared across events. Be careful not to overload the capacity of an interface.

5. Complete the audio fields in the same way – **Audio SDP Location**, **Media Index**, and **Interface** fields.

   Keep in mind that you can support SMPTE 2022-7 for some streams and not for others.

6. If the source includes an ancillary data stream, choose **Add Ancillary SDP +**. Complete the **Ancillary SDP Location**, **Media Index**, and **Interface** fields.

Ingesting SRT content

AWS Elemental Live supports ingest of a transport stream (TS) that is sent from an upstream system that is set up as an SRT listener. In this scenario, the upstream system waits for Elemental Live (the SRT caller) to initiate the SRT connection handshake that precedes the transmission of the source content. The transport stream source can be encrypted with AES.

Roles

With an SRT input, Elemental Live has two roles and the upstream system has two roles:

- For the SRT connection handshake: Elemental Live is the SRT caller (the party that initiates the handshake). The upstream system is the SRT listener.

- For the transmission: Elemental Live is always the receiver of the content. The upstream system is always the sender of the content.

Topics

Get ready

Before you create an SRT input in your event, speak to the administrator of the upstream system, and obtain the following information:

- The IP address and port of the content. For example, 192.168.1.2:5000.
- Whether the content is encrypted. If it is encrypted, find out if encryption uses AES 128, AES 192, or AES 256.

  Obtain the passphrase from the administrator of the upstream system.
- The stream ID, if the upstream system uses this identifier. The sender might require a stream ID, in which case you must obtain it. Otherwise the SRT handshake between the caller and listener might fail.
- The preferred latency (in milliseconds) for implementing packet loss and recovery. Packet recovery is a key feature of SRT.

Setting up an SRT input

You create an SRT input in Elemental Live in order to ingest a transport stream. The upstream system that provides the transport stream is the sender and is set up as an SRT listener.

To set up an SRT input

1. From the **Input** menu in your event, select **Secure Reliable Transport**.
2. Complete the following fields:

   - **Network Location**: The IP address and host that you obtained from the upstream, with the protocol. For example:

     ```
     srt://192.168.1.2:5000
     ```
   - **Interface**: See the tooltip on the web interface.
   - **Latency**: The latency that you want to apply to aid in packet recovery in Elemental Live. You should closely match the latency that the upstream system prefers. For more details, see the tooltip.
   - **Stream ID**: Enter this value if the upstream system provided it.
3. Complete the encryption fields, if applicable:

- **Encryption**: Choose **None**, or choose the encryption level that you obtained from the upstream system.
- **Passphrase**: Enter the passphrase that the upstream system provided.

Ingesting from VSF TR-01 in a TS input

Elemental Live supports VSF TR-01 ingest over a transport stream (TS). Support for TR-01 provides a solution to migrating from SDI infrastructure to an all-IP infrastructure. Handling of IP inputs using TR-01 is almost feature-equivalent to the handling of SDI inputs.

The presence of TR-01 means that the video and audio are TR-01-compliant, as described below. In addition, SMPTE-2038 is required in a TR-01-compliant TS.

Supported Sources

Video in TR-01

The video source must be encoded with JPEG 2000 and can be one of the following:

Definition	Resolutions	Maximum Bitrate	Color Space
SD	576i/25 and 480i/29.97	200 MBps	ITU-R BT 601-6
HD-SDI	Up to 1080i/25/29.97 and 720p/50/59.94	200 MBps	ITU-R BT 709-5
3G-SDI	1080p/50 and 1080p/59.94	200 MBps	ITU-R BT 709-5

Audio codecs in TR-01

The audio source can contain up to eight audio signals. Each audio signal is wrapped in SMPTE-302. At any given point in the source, the SMPTE-302 can contain:

- One AES3 signal pair of Dolby AC3. The AES3 pair is wrapped in SMPTE-337.

- One AES3 signal pair of Dolby E-AC3.The AES3 pair is wrapped in SMPTE-337.
- One AES3 signal pair of Dolby E. The AES3 pair is wrapped in SMPTE-337. There is support for multi-program Dolby E streams, in which case individual programs can be selected
- Or one uncompressed PCM audio stream.

There is support for source monitoring of audio codecs carried in SMPTE-302. If the audio codec (or type) changes, Elemental Live detects the change and processes the new codec appropriately.

Elemental Live enforces the rule about the contents of the SMPTE-302 as follows:

- For the audio codecs other than PCM, if the SMPTE-302 contains invalid content (there is more than one signal pair in the SMPTE-302), then only the first signal pair will be processed and the remaining signal pairs will be discarded.
- For PCM, if the SMPTE-302 contains invalid content (there is more than one signal pair in the SMPTE-302), then Elemental Live will process both. The probable result will be garbled audio.
- If the audio source contains an unsupported codec or format, Elemental Live logs a message, displays an alert on the web interface, and inserts silence. Elemental Live continues monitoring the source and detecting the format.

Setting up a new event

Perform the following steps to set up a new event.

1. In Input, set the Input type to Network Input. In Network Location, specify an IP address.
2. In Input > Advanced, set the Prefer SMPTE-2038 field as appropriate:
   - If you do not want Elemental Live to look at the SMPTE-2038, uncheck this field. When the Prefer SMPTE-2038 field is unchecked (default), then Elemental Live follows the legacy behavior. It looks for ancillary data in the native TS. Even if a SMPTE-2038 PID is present, Elemental Live ignores that PID.
   - If you want to use the SMPTE-2038 ancillary data, and assuming the stream contains well-formed SMPTE-2038 (above), then check this field.

     For information about how SMPTE-2038 ancillary data is extracted and processed, see Handling ancillary data in SMPTE 2038.
3. In Input > Video Selector and Input > Audio Selector, set up the video and audio in the usual way. There are no special steps for setting up TR-01 video or audio inputs.

4. Set up the remainder of the event in the usual way.

The video identified in the video selector and the audio identified in the audio selector are extracted from the TR-01 in the TS. The video and audio are processed in the usual way, as specified in the output sections of the event. The ancillary data (if used) is extracted from the SMPTE-2038 and processed as described in Handling ancillary data in SMPTE 2038.

Migrating from SDI Inputs to TR-01 inputs

If you have an event that is configured to use an SDI input, you can easily adapt it to use an IP network feed that contains TR-01-compliant video and audio. Change the existing event as follows:

- Change the **Input Type** to **Network Input**. In **Network Location**, specify an IP address.
- Set the **Prefer SMPTE-2038** box as appropriate:
  - If you do not want Elemental Live to look at the SMPTE-2038, uncheck this field. When the **Prefer SMPTE-2038** field is unchecked (default), then Elemental Live follows the legacy behavior. It looks for ancillary data in the native TS. Even if a SMPTE-2038 PID is present, Elemental Live ignores that PID.
  - If you want to use the SMPTE-2038 ancillary data, and assuming the stream contains well-formed SMPTE-2038 (above), then check this field.

    For information about how SMPTE-2038 ancillary data is extracted and processed, see Handling ancillary data in SMPTE 2038.

  There is no requirement to change any other fields: the event should now perform the same processing as previously.

The video identified in the video selector and the audio identified in the audio selector are extracted from the TR-01 in the TS. The video and audio are processed in the usual way, as specified in the output sections of the event. The ancillary data (if used) is extracted from the SMPTE-2038 and processed as described in Handling ancillary data in SMPTE 2038.

Working with outputs

This chapter describes how to set up the different types of outputs that AWS Elemental Live supports.

Topics

Reference: Supported outputs

The following tables describe the supported mechanisms for delivering output to the downstream destination. Read this section to find the output group type that applies to your intended downstream destination.

Topics

Output types for delivery to an AWS service

This table describes options that you can use to send output from Elemental Live to a downstream system that is an AWS service. Each row describes a different use case.

The rows are sorted by downstream system (destination AWS service). The *Released* column identifies the version of Elemental Live that introduced the output group type.

Downstream system	Type of output	Description	Type of output group	Output option (if any)	Supported protocol	Live output supported	VOD output supported	Released
Amazon S3	DASH	Send DASH content to Amazon S3 .	DASH		Custom protocol: s3:// or s3ssl://	Yes	Yes	2.14.3
Amazon S3	HLS	Send HLS content to an Amazon S3 bucket.	HLS	A TS container or an fMP4 container	Custom protocol: s3:// or s3ssl://	Yes	Yes	Before 2.14.0
Amazon S3	JPEG frame capture files	Send JPEG files to an Amazon S3 bucket.	Archive	Raw (no container)	Custom protocol: s3:// or s3ssl://	No	Yes	Before 2.14.0
Amazon S3	VOD files	Send compressed VOD output to an Amazon	Archive	Various containers are supported . See the list below.	Custom protocol: s3:// or s3ssl://	No	Yes	Before 2.14.0

Downstream system	Type of output	Description	Type of output group	Output option (if any)	Supported protocol	Live output supported	VOD output supported	Released
		S3 bucket.						
AWS Elemental MediaConnect	Transport stream	Send a redundant transport stream (TS) to a Zixi push flow on MediaConnect	Reliable TS	AWS Elemental MediaConnect option. See Note A.	Not applicable	Yes	No	2.14.3
AWS Elemental MediaConnect	Transport stream	Send a redundant transport stream (TS) to an SRT listener flow on MediaConnect.	Reliable TS	AWS Elemental MediaConnect option. See Note B	Not applicable	Yes	No	2.14.3

Downstream system	Type of output	Description	Type of output group	Output option (if any)	Supported protocol	Live output supported	VOD output supported	Released
AWS Elemental MediaConnect, via Direct Connect to your own AWS VPC.	SMPTE 2110 stream	Send a JPEG XS SMPTE 2110 stream to MediaConnect. SMPTE 2110 requires redundant inputs when sending to MediaConnect. Note that you can't send an uncompressed SMPTE 2110 stream to MediaConnect.	SMPTE 2110		rtp://	Yes	No	2.22.0

Downstream system	Type of output	Description	Type of output group	Output option (if any)	Supported protocol	Live output supported	VOD output supported	Released
AWS Elemental MediaStore	DASH	Send DASH content to MediaStore.	DASH		Custom protocol: ems:// or emsssl://	Yes	Yes	2.14.3
AWS Elemental MediaStore	HLS	Send HLS content to a container on MediaStore.	HLS	A TS container or an fMP4 container	Custom protocol: ems:// or emsssl://	Yes	Yes	2.14.3

Downstream system	Type of output	Description	Type of output group	Output option (if any)	Supported protocol	Live output supported	VOD output supported	Released
AWS Elemental MediaPackage	HLS	Send HLS content to a MediaPackage channel using the HTTPS protocol. The MediaPackage channel must be on your own AWS account.	HLS	A TS container	https://	Yes	No	Before 2.14.0

Downstream system	Type of output	Description	Type of output group	Output option (if any)	Supported protocol	Live output supported	VOD output supported	Released
AWS Elemental MediaPackage v2	HLS	Send HLS content to a channel in MediaPackage v2 using the HTTPS protocol. The output is typically (but not necessarily) part of a glass-to-glass low latency workflow.	HLS	A TS container	https://	Yes	No	2.25.0

Note A. You could send to a MediaConnect Zixi flow using the Zixi option, but the AWS Elemental MediaConnect option provides a seamless integration with MediaConnect. The Zixi option is designed for destinations other than MediaConnect.

Note B. You could send to a MediaConnect SRT flow using the SRT option, but the AWS Elemental MediaConnect option provides a seamless integration with MediaConnect. The SRT option is designed for destinations other than MediaConnect.

Output types for delivery to non-AWS destinations

This table describes options that you can use to send output from Elemental Live to a destination that is not an AWS service. Each row describes a different use case.

The rows are sorted by type of output group. The Version column identifies the version of Elemental Live that introduced the output group type.

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
A folder on the appliance	VOD files	Send VOD output to a local folder	Archive	Various containers are supported. See the list below.	Not applicable	No	Yes	Before 2.14.0
A folder on the appliance	VOD files	Send uncompressed VOD output to a local folder	Archive	Raw (no container)	Not applicable	No	Yes	Before 2.14.0
A folder on the appliance	JPEG frame capture files	Send JPEG files to a local folder	Archive	Raw (no container)	Not applicable	No	Yes	Before 2.14.0

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
An HTTP or HTTPS server	DASH	Send DASH content to an HTTP or HTTPS server	DASH		http:// or https://	Yes	Yes	Before 2.14.0
An HTTP or HTTPS server	HLS	Send HLS content to an HTTP or HTTPS server	HLS	A TS container or an fMP4 container	http:// or https://	Yes	Yes	Before 2.14.0
Akamai CDN	HLS	Send HLS content to an Akamai CDN.	HLS	A TS container or an fMP4 container	http:// or https://	Yes	No	Before 2.14.0

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
A supported CDN, over HTTP or HTTPS	Microsoft Smooth	Send content to an origin server or CDN that supports Microsoft Smooth Streaming.	Microsoft Smooth		http:// or https://	Yes	Yes	Before 2.14.0
A Zixi broadcaster	Transport stream	Send redundant or non-redundant transport stream (TS) content to an endpoint that supports the Zixi protocol.	Reliable TS	Zixi option	zixi://	Yes	No	2.14.3

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
An SRT caller endpoint or listener endpoint	Transport stream	Send redundant or non-redundant RTP content to an SRT endpoint that is set up as either a caller or a listener.	Reliable TS	SRT option	srt://	Yes	No	2.22.2
RTMP or RTMPS server	RTMP	Send a stream to a server that supports the RTMP protocol.	RTMP		rtmp:// or rtmps://	Yes	No	Before 2.14.0

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
A system that supports SMPTE 2110	SMPTE 2110 stream	Send an uncompressed SMPTE 2110 stream to a device that supports SMPTE 2110. Redundant streams using SMPTE 2022-7 are supported but optional.	SMPTE 2110		rtp://	Yes	No	2.19.3

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
A system that supports SMPTE 2110	SMPTE 2110 stream	Send a JPEG XS compressed SMPTE 2110 stream to a device that supports SMPTE 2110. Redundant streams using SMPTE 2022-7 are supported but optional.	SMPTE 2110		rtp://	Yes	No	2.21.3

Downstream system	Type of output	Description	Type of output group	Output group option	Supported protocol	Live output supported	VOD output supported	Version
A system that supports RTP or UDP over unicast or multicast	Transport stream	Send a transport stream (TS) to a server that supports unicast or multicast RTP or unicast or multicast UDP.	UDP/TS		rtp:// or udp://	Yes	No	Before 2.14.0

Live outputs: Supported codecs

This table specifies the audio codecs that are supported in output groups that support live outputs.

Use this table if you identified a use case in the section called "Types for delivery to AWS" or the section called "Types for delivery to other" and your output is live output.

In the table, find the output group and container (if applicable) for the user case that you identified. Then read across to identify the video and audio codecs that are supported for that output group.

Output group type	Container	Video Codecs	Audio Codecs
DASH		H.264 H.265	AAC Dolby Digital Dolby Digital Plus

Output group type	Container	Video Codecs	Audio Codecs
			Dolby Digital Plus with Atmos (converted) Dolby Digital Plus with Atmos (as passthrough)
HLS	Standard	H.264 H.265	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converted) Dolby Digital Plus with Atmos (as passthrough)
HLS	fMP4	H.264 H.265	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converted) Dolby Digital Plus with Atmos (as passthrough)

Output group type	Container	Video Codecs	Audio Codecs
Microsoft Smooth		H.264	AAC Dolby Digital Dolby Digital Plus with Atmos (as passthrough)
Reliable TS		H.264 H.265 MPEG2	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Plus with Atmos (as passthrou gh)
RTMP		H.264	AAC
SMPTE 2110		Uncompressed JPEG XS (Version 2.21.3 and later)	PCM Dolby Digital Dolby Digital Plus

Output group type	Container	Video Codecs	Audio Codecs
UDP/TS		H.264 H.265 MPEG2	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus with Atmos (as passthrough) DTS Express MPEG-1, layer II

VOD outputs: Supported codecs

This table specifies the codecs that are supported in output groups that support VOD outputs.

Use this table if you identified a use case in the section called "Types for delivery to AWS" or the section called "Types for delivery to other" where your output is VOD output.

In the table, find the output group and container (if applicable) for the user case that you identified. Then read across to identify the video and audio codecs that are supported for that output group.

Output group type	Container (if applicable)	Video Codecs	Audio Codecs
Archive	Raw (No container)	Frame Capture (MJPEG) H.264 H.265	AAC AIFF Dolby Digital

Output group type	Container (if applicable)	Video Codecs	Audio Codecs
		MJPEG MPEG2	Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus with Atmos (as passthrough) Archive output in TS and Raw container (no container) DTS Express MPEG Audio WAV

Output group type	Container (if applicable)	Video Codecs	Audio Codecs
Archive	Raw (No container)	Uncompressed For information about the 4CC codes that are supported, and the pixel formats for each 4CC code, see the table below	AAC AIFF Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh) DTS Express MPEG Audio WAV
Archive	3GPP	H.264	AAC
Archive	MXF	MPEG-2	WAV

Output group type	Container (if applicable)	Video Codecs	Audio Codecs
Archive	MPEG-2 Transport Stream	H.264 H.265 MPEG2	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh) MPEG Audio
Archive	MPEG-4 (.mp4)	H.264 H.265	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh) DTS Express
Archive	MPEG-4 Flash (.f4v)	H.264	AAC

Output group type	Container (if applicable)	Video Codecs	Audio Codecs
Archive	QuickTime	H.264 MPEG2 Apple ProRes Uncompressed (For information about the 4CC codes that are supported, and the pixel formats for each 4CC code, see the table below)	AAC AIFF Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh) WAV
HLS	Standard	H.264 H.265	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh)

Output group type	Container (if applicable)	Video Codecs	Audio Codecs
HLS	fMP4	H.264 H.265	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh)
Microsoft Smooth		H.264	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus ATMOS (as passthrou gh)
DASH		H.264 H.265	AAC Dolby Digital Dolby Digital Plus Dolby Digital Plus with Atmos (converte d) Dolby Digital Plus ATMOS (as passthrou gh)

Use this table if you want to deliver uncompressed video in an Archive output group. The table specifies the 4CC codes that are supported for uncompressed video, and the pixel formats for each 4CC code.

4CC Code for the video format	Pixel formats		
YV12	8-bit 4:2:0 planar		
I420	8-bit 4:2:0 planar		
NV12	8-bit 4:2:0 chroma interleaved		
YV16	8-bit 4:2:2 planar		
UYVY	8-bit 4:2:2 planar		
YUYV	8-bit 4:2:2 planar		
S210	10-bit 4:2:2 packed		

Definition of containers and codecs

Codec or Container	Direction	Statement
MXF input container for video	Input	Complete list of supported containers is: AS-02; OP-1a, OP-1b, OP-1c, OP-2a, OP-2b, and OP-2c.
Apple ProRes video codec	Input	Complete list of supported codecs is: Apple Prores 444 (all profiles), Apple Prores 4444 (all profiles), and Apple Prores 422 (all profiles). Apple Prores 444 and 4444 are

Codec or Container	Direction	Statement
		converted to Apple Prores 422 during input handling.
MPEG-2 video codec	Input	Complete list of supported codecs is: MPEG-2, and ATSC (A/53).
AAC audio codec	Input	Complete list of supported profiles is: LC-AAC, HE-AAC v1, and HE-AAC v2.
Dolby Digital audio codec	Input	Dolby Digital is also known as AC-3. Dolby Digital is a licensed codec; however, no license is required to decode this codec in input. For more notes, see Audio: Transcode support and passthrough support for Dolby audio codecs.
Dolby Digital Plus audio codec	Input	Dolby Digital Plus is also known as Enhanced AC-3 and is frequently abbreviated as DD+ or EC-3 and E-AC-3. Decoding of Dolby Digital Plus requires the Elemental Audio Decode Package license option. For more notes, see Audio: Transcode support and passthrough support for Dolby audio codecs.

Codec or Container	Direction	Statement
Dolby E frames carried in PCM audio streams tagged with SMPTE-337	Input	Decoding of Dolby E in PCM stream requires the Elemental Audio Decode Package license option.
Apple ProRes video codec in output	Output	Complete list of supported codecs is: Apple Prores 422 (all profiles).
Dolby Digital audio codec	Output	Encoding with Dolby Digital requires the Elemental Advanced Audio Package license option. For more notes, see Audio: Transcode support and passthrough support for Dolby audio codecs.
Dolby Digital Plus audio codec	Output	Encoding with Dolby Digital Plus requires the Elemental Advanced Audio Package license option. For more notes, see Audio: Transcode support and passthrough support for Dolby audio codecs.
Dolby E pass-through	Output	See Audio: Transcode support and passthrough support for Dolby audio codecs.

Codec or Container	Direction	Statement
DTS Express	Output	Encoding with DTS Express requires the Elemental Advanced Audio Package license option.
MPEG Audio codec	Output	MPEG-1 Audio Layer II only

Details about codecs

Audio codecs and supported conversions

Generally, Elemental Live can convert any audio codec that is supported as a source to any audio codec that is supported as an output. However, there are some constraints, as follows.

- Constraint when Dolby Digital with Atmos is the source. Conversion to another codec isn't supported. You can only pass through this source codec.
- Constraint when converting to another codec (other than Dolby Digital with Atmos) and changing the coding mode. The following rules apply:
  - The source must contain at least as many channels as the output. For example, to produce Dolby 5.1 (6 channels), the source must contain 6 channels.
  - The source can contain fewer channels. For example, you can convert Dolby 5.1 to AAC 2.0.

  In both cases, you might need to remix the channels in the output.

Audio: Transcode support and passthrough support for Dolby audio codecs

Three Dolby codecs are supported and can be transcoded or passed through as follows:

- Dolby Digital (AC-3): Can be re-encoded as Dolby Digital or the original Dolby Digital can be passed through.
- Dolby Digital Plus (E-AC-3): Can be re-encoded as Dolby Digital Plus or the original Dolby Digital Plus can be passed through.
- Dolby E: Can be passed through as Dolby E. Cannot be re-encoded as Dolby E.

The following table specifies the fields on the input side and on the output side that control passthrough versus transcoding.

Input		Output > Stream		Result
Codec Detected in Input	Value in Unwrap SMPTE 337 Field	Value in Output Codec Field	Value in Automatic Passthrough Field	
Dolby Digital	n/a	Dolby Digital	n/a	Re-encoded to Dolby Digital
Dolby Digital	n/a	Dolby Digital Passthrough	n/a	Passthrough of Dolby Digital
Dolby Digital Plus	n/a	Dolby Digital Plus	Checked	Passthrough of Dolby Digital Plus
Dolby Digital Plus	n/a	Dolby Digital Plus	Unchecked	Re-encoded to Dolby Digital Plus
Dolby Digital Plus	n/a	Dolby Digital Passthrough	n/a	Passthrough of Dolby Digital Plus
Dolby Digital Plus and a non-Dolby Digital Plus codec	n/a	Dolby Digital Plus	Checked	The non-Dolby Digital Plus audio is transcoded to Dolby Digital Plus. The Dolby Digital Plus audio is passed through (it will not be re-

Input		Output > Stream		Result
				encoded as Dolby Digital Plus).
Dolby Digital Plus and a non-Dolby Digital Plus codec	n/a	Dolby Digital Plus	Unchecked	The non-Dolby Digital Plus audio is transcoded to Dolby Digital Plus. The Dolby Digital Plus audio is re-encoded to Dolby Digital Plus.
Dolby Digital Plus and a non-Dolby Digital Plus codec	n/a	Dolby Digital Passthrough	n/a	Not valid: setting up like this causes a validation error.
Dolby E in a PCM stream tagged with SMPTE 337	Unchecked	Uncompressed WAV	n/a	Passthrough of Dolby E.
Dolby E in a PCM stream tagged with SMPTE 337	Unchecked	Uncompressed AIFF	n/a	Passthrough of Dolby E.

Video: H.264 (AVC) support

H.264 is supported in the following variations.

Direction	Chroma Sampling	Bit Depth	Profile/Format	Level	License Requirement
Input	4:2:0	8-bit and 10-bit	Baseline, Main, High, High 10, High 4:2:2, High 10 Intra, High 422 Intra, AS-11/RP2 027	1.0-5.2	None
	4:2:2	8-bit and 10-bit			None
Output	4:2:0	8-bit			None
	4:2:0	10-bit			For AVC Intra, purchase the BCE license pack.
	4:2:2	8-bit and 10-bit			For AVC Intra, purchase the BCE license pack. For AVC 4:2:2, purchase the BCE license pack.

Both 4:2:0 and 4:2:2 chroma sampling are supported in raw outputs (.264, .avc, extensions) and MPEG-2 transport streams.

In all other containers, only 4:2:0 chroma sampling is supported.

Video: HEVC (H.265) support

HEVC is supported in the following variations.

Direction	Chroma Sampling	Bit Depth	Profile/Format	Level	License Requirement
Input	4:2:0	8-bit and 10-bit	Main, Main 10, Main 4:2:2 10	1.0-5.2	None
	4:2:2	8-bit and 10-bit			None
Output	4:2:0	8-bit and 10-bit			For HEVC 4:2:0, request the HC license.
	4:2:2	8-bit and 10-bit			For HEVC 4:2:2, purchase the BCE license pack.

Both 4:2:0 and 4:2:2 chroma sampling are supported in raw outputs (.264, .avc, extensions) and MPEG-2 transport streams. In all other containers, only 4:2:0 chroma sampling is supported.

Video: MPEG-2 (H.262) support

MPEG-2 is supported in the following variations.

Direction	Chroma Sampling	Bit Depth	Profile/Format	Level	License Requirement
Input	4:2:0, 4:2:2	8-bit	Simple, Main, 422, IMX/D-10	Low, Main, High1440, High	N/A
Output	4:2:0, 4:2:2	8-bit	Simple, Main, 422, IMX/D-10	Low, Main, High1440, High	N/A

Codec support for CPU-only and GPU-enabled deployments

The codecs listed in the table below are supported with equivalent video quality at equivalent operating points on both CPU-only and GPU-enabled appliances.

Output Codec	Equivalent VQ, CPU-Only and GPU-Enabled Systems	Accelerated Encoding on GPU-Enabled Systems
H.264 (AVC)	Yes	Yes
HEVC (H.265)	Yes	Yes
MPEG-2 (H.262)	Yes	Yes
ProRes	Yes	No
Uncompressed YUV	Yes	No
JPEG (Frame Capture)	Yes	No

Setting up Elemental Live as a Contribution Encoder for AWS Elemental MediaConnect

You can set up an AWS Elemental MediaConnect flow as the output from Elemental Live. In this setup, Elemental Live is the contribution encoder for a MediaConnect flow. You can choose to encrypt the output during delivery to MediaConnect.

Topics

- Assumptions
- Setup procedure
- How delivery from Elemental Live to MediaConnect works at runtime

Assumptions

This section assumes the following:

- We assume that you know how to use the AWS console, AWS Identity and Access Management, and AWS Elemental MediaConnect, and that you have access to the user guides for the AWS services:
  - What Is AWS Elemental MediaConnect? in the *AWS Elemental MediaConnect User Guide*
  - What Is IAM? in the *IAM User Guide*
  - What Is AWS Secrets Manager? in the *AWS Secrets Manager User Guide*.
- We assume that you have already set up permissions for MediaConnect— you have created at least one AWS user and given permissions to those users so that they can use the features of MediaConnect. Specifically, for the purposes of this procedure, the user can create a MediaConnect flow. You have also set up MediaConnect as a trusted entity with Secrets Manager; see Step 3: Create an IAM Role with a Trusted Relationship in the *AWS Elemental MediaConnect User Guide*.
- We don't assume that you have set up Elemental Live with permissions in AWS. Setting up those permissions is one of the steps in this section.

Setup procedure

Step A: Create a role in IAM and attach policies

You must use AWS Identity and Access Management (IAM) to set up AWS Elemental Live as an AWS user (the "Elemental Live user") and give it permissions so that it can communicate with AWS Secrets Manager and AWS Elemental MediaConnect. You must:

- Create policies that contain specific permissions.
- Create the "Elemental Live user" in your AWS account. The user must be in the same AWS account as the user who is operating MediaConnect.
- Associate the Elemental Live user with those policies, which gives the user the permissions of those policies.

Create a policy for Elemental Live to make requests to MediaConnect

Elemental Live must have permissions on MediaConnect. Follow this procedure to set up these permissions:

To create a policy for Elemental Live to make requests to MediaConnect

1. Log into the AWS console and go to the IAM console.
2. On the left menu, choose Policies. Use the filters to determine if there is already a policy with a name similar to "ElementalAccessToMediaConnect".
3. If the policy does not exist, choose Create policy. Click the Visual editor tab and create the policy using the IAM policy generator. This generator lets you choose the service from a list and then choose operations from a list:

   - Service: MediaConnect.
   - Actions: Under List, click DescribeFlow and ListFlows.
   - Resources: If your organization does not have strict rules about accessing containers on MediaConnect, you can ignore this section; you will have access to all flows. Otherwise, follow your internal policies to identify specific flows.
   - Give the policy a name such as "ElementalAccessToMediaConnect"

For more information about how to create and manage IAM policies, see the IAM User Guide.

Create a policy for Elemental Live to make requests to Secrets Manager

If you plan to encrypt the output from Elemental Live when you send it to MediaConnect, then Elemental Live must have permissions on AWS Secrets Manager. Follow this procedure to set up these permissions:

To create a policy for Elemental Live to make requests to Secrets Manager

1. Log into the AWS console and go to the IAM console. Choose Policies and look for a policy that gives MediaConnect the permissions for Secrets Manager. If you or someone else previously followed the procedure in Step 2: Create an IAM Policy to Allow AWS Elemental MediaConnect to Access Your Secret, then this policy will be called **`SecretsManagerForMediaConnect`**.
2. If this policy exists, make sure it contains the following actions:

   - DescribeSecret
   - GetResourcePolicy
   - GetSecretValue
   - ListSecretVersionIds

3. Also make sure that the resources section gives access to the ARN of the secret that you will use. Read the information in IAM Policy Examples for Secrets in AWS Secrets Manager. You may need to edit the policy to include the ARN for this secret in the resources section.
4. If the policy does not exist, follow the procedure in Step 2: Create an IAM Policy to Allow AWS Elemental MediaConnect to Access Your Secret to create the policy.

Create a user

To create a user

1. Log into the AWS console and go to the IAM console.
2. If the user does not exist or it does exist but you want to create separate users for each Elemental product, choose Add User. (Note that you may want separate users for separate products, but there is probably no need to create a separate user for each Elemental node.) Follow the prompts to add the user with this information:
    - Give the user a name such as **`ElementalUser`**.
    - For Access type, choose Programmatic access. Do not choose Console access.
    - In permissions, choose Attach existing policies directly. Attach the policies you created above. For example, **`ElementalAccessToMediaConnect`** and **`SecretsManagerReadSecrets`**.
    - Ignore tags.
3. Create the user and choose Close.
4. Choose the user by clicking the name, for example, click ElementalUser.
5. Choose the Security tab.
6. Click **Create Access Key**.
7. On the Create access key dialog, choose to download the .csv file. Save the file in a safe place, so that you have a permanent record of the access key ID and the secret access key.

    The Access key ID looks like this:AKIAIOSFODNN7EXAMPLE

    The Secret access key looks like this: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
8. Give the Access key ID and the Secret access key to the Elemental Live operator. Do not give the username and password to the operator.

9. How it works: You have created an AWS user with the permissions required to make requests to MediaConnect and optionally to Secrets Manager. When the Elemental Live user sets up the output with MediaConnect as the destination, they will enter the Access key ID and Secret access key. When the Elemental Live event is running, Elemental Live sends these two IDs to the AWS services, instead of sending the user name and password. These IDs provide authorization for the Elemental Live node to make requests to the AWS services.

Step B: Set up for encryption (optional)

If you are encrypting the Elemental Live output, you must generate an encryption key and set it up in Secrets Manager. In this scenario, Secrets Manager is effectively acting as the key server for the encryption key. Secrets Manager serves the key to Elemental Live so it can encrypt and to MediaConnect to it can decrypt. In this scenario, encryption/decryption is supported with a static SHA-256 encryption key and using the AES-256, AES-192, or AES-128 algorithm.

Perform the following steps according to the security policies and procedures for your organization.

To set up for encryption

1. Use a suitable tool for generating a SHA-256 encryption key from a seed that you specify. AWS does not provide a generation tool. Note that you need only one key, even if you are creating two flows.
2. Save the key to the secret, as described in [Setting Up Static Key Encryption Using AWS Elemental MediaConnect](). You must assign a name to the secret, for example, "key_sports". Save the key in the same AWS Region as the flow you plan to create.
3. Make a note of the ARN for this secret. You need this ARN when you create the MediaConnect flow. It looks like the following example, where "key_sports" is the name you assigned to the secret.

```
arn:aws:secretsmanager:us-west-2:111122223333:secret:key_sports-7g8H9i
```

Step C: Create the AWS Elemental MediaConnect flows

You must follow this procedure before you create the Elemental Live outputs because Elemental Live needs data that is generated by this procedure.

To create the MediaConnect flows

1. Create one or two MediaConnect flows. (Create two flows if your have set up Elemental Live for output redundancy using output locking. If you have not set up redundant outputs, create one flow.)
2. Follow the procedure in Creating a Flow in the *AWS Elemental MediaConnect User Guide*.
3. Complete Availability Zone and Name as appropriate. These fields do not relate to using Elemental Live as the source.
4. In the Source section, follow the steps for setting up a standard source. Specifically:
   - Protocol: Zixi push.
   - Whitelist CIDR block: This is the IP address (in CIDR format) of the Elemental node that will be delivering to this flow. It must be a public facing IP address. Speak to your organization's administrator for a value to enter here.
   - Stream ID: You must enter a value when Elemental Live is the source.
5. If you are encrypting the video, check Enable in the Decryption section and complete the fields as described in the MediaConnect documentation. Specifically:
   - Decryption type: Always Static key.
   - Role ARN: The role that has been set up for MediaConnect to be a trusted entity with Secrets Manager. See Step 3: Create an IAM Role with a Trusted Relationship in the *AWS Elemental MediaConnect User Guide*. You must specify this role ARN here so that MediaConnect can obtain the encryption key.
   - To find the ARN for the role, go to the IAM console, choose Roles, click the name of the role, and look at the Role ARN field in the Summary. The role ARN looks like this:

     ```
     arn:aws:iam::111122223333:role/MediaConnectASM
     ```
   - Secret ARN: The ARN you obtained in step A, for example: `arn:aws:secretsmanager:us-west-2:111122223333:secret:key_sports-7g8H9i`
   - Decryption algorithm: Specify the algorithm that you want to use. Elemental Live will be instructed to use this algorithm to encrypt. MediaConnect will read this information and use this algorithm to decrypt.
6. When you create each flow, MediaConnect creates an ARN for that flow. The ARNs look like the following, where "curling_finals_A" and "curling_finals_B" are the flow names you specified in each flow:

```
arn:aws:mediaconnect:us-
west-1:111122223333:flow:1bgf67:curling_finals_A

arn:aws:mediaconnect:us-
west-1:111122223333:flow:9pmlk76:curling_finals_B
```

7. Make a note of these ARNs. You need them to set up the Elemental Live output(s).

Step D: Create the Elemental Live output group

You must create one output group of type "Reliable TS". Inside that group, you must create one or two outputs: create two outputs if you created two MediaConnect flows, create one output if you created only one flow.

To create the Elemental Live output group

1. In the Elemental Live event, go to Output Groups > Reliable TS.
2. Click Add Output to create an output in this Reliable TS output group.
3. Complete the fields in each output as follows:
   - Delivery Protocol: Choose AWS Elemental MediaConnect.
   - Destination/Amazon Resource Name: Enter the ARN for the flow. Following from the example above, enter the following in the first output:

     ```
     arn:aws:mediaconnect:us-
     west-1:111122223333:flow:1bgf67:curling_finals_A
     ```
   - Interface: Optional; see the tooltip.
   - Lock icon: Click this icon. Two more fields appear:
     - Username/Access Key ID: The Access key ID for the user you created in AWS IAM. For example, AKIAIOSFODNN7EXAMPLE
     - Password/Secret Access Key: The Secret access key for this user. For example, wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
4. Note that there is no encryption field. See "How It Works at Runtime", below, to understand how encryption is handled.
5. Repeat these steps to create a second output in this output group, if applicable. Use the same Access key ID and Secret access key.

How delivery from Elemental Live to MediaConnect works at runtime

Here is the data that AWS Elemental Live has: The flow ARN. The Access key ID and Secret access key. Here is the data that MediaConnect has: The flow ARN. The destination IPs and protocol details. The encryption type and algorithm. The role ARN (for obtaining the secret - the encryption key). The secret ARN. When the event starts, Elemental Live authenticates with AWS using the AWS access key ID and AWS secret access key. It then sends the flow ARN to MediaConnect. MediaConnect accepts the request because Elemental Live has permission to make requests to MediaConnect. MediaConnect looks up the flow and determines if the flow is set up for encryption.

- If the flow is set up for encryption, MediaConnect sends the encryption type and algorithm information, and the secret ARN to Elemental Live. Elemental Live uses the secret ARN to get the secret (the encryption key) from Secrets Manager. Secrets Manager accepts the request from Elemental Live because Elemental Live has permission to get this secret.

  Elemental Live uses the encryption key to encrypt the video and sends the encrypted video to MediaConnect.

  MediaConnect in its turn uses the secret ARN to get the secret (the encryption key) from the Secrets Manager. Secrets Manager accepts the request from MediaConnect because MediaConnect has permission to get this secret; it has permission because it has been set up as a trusted entity with Secrets Manager. MediaConnect uses the encryption key to decrypt the video.
- If the flow is not set up for encryption, MediaConnect instructs Elemental Live to deliver the video unencrypted. Secrets Manager is not involved.

Delivering HLS output to MediaPackage version 2

This section describes how to deliver an HLS output from AWS Elemental Live to an AWS Elemental MediaPackage channel that uses MediaPackage v2. You can optionally configure the video output for low latency, to support a glass-to-glass low latency workflow.

The information in this section assumes that you are familiar with the general steps for creating an event.

1. Obtain the following information from the MediaPackage operator:
   - The URL for each destination for the output group. For delivery to MediaPackage v2, the URL will always include the string `mediapackagev2`.

- The credentials that Elemental Live must include to deliver this output to MediaPackage v2. For example:

  An *access key ID* that looks like this: **`AKIAIOSFODNN7EXAMPLE`**

  A *secret access key* that looks like this: **`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`**

2. In the Elemental Live event, go to **Output Groups**, then to **Apple HLS**.
3. Set up the output group in the usual way. Complete the following fields as specified:

Section	Field	Description
Apple HLS Settings	Destination	The URL that you obtained from the MediaPackage operator. When you tab out of this field, two fields automatically appear: **Username or Access Key ID** and **Password or Secret Access Key**.
	Username or Access Key ID	Enter the access key ID that you obtained from the MediaPackage operator.
	Password or Secret Access Key	Enter the secret access key that you obtained from the MediaPackage operator.
	HTTP Push Dialect	Choose **Basic PUT**
	Advanced > Encryption	Choose **Disabled** because MediaPackage doesn't support encryption.
	Advanced > Chunked Transfer	Choose **Disabled**.
Outputs	Segment Type	Choose **TS**.

Section	Field	Description
		You can't send fMP4 to MediaPackage.

4. If you want to implement low latency in the encoder, follow the guidance for these fields:

Section	Field	Description
Apple HLS Settings	Segment Length	We recommend 1 second for better latency.
	Minimum Segment Length	A value is required for delivery to MediaPackage. This value can affect latency.
	Retry Interval	We recommend the same value as the segment length. This value can affect latency.
	Num Retries	This value can affect latency.
	FileCache Size	This value can affect latency. We recommend a lower number.
	Restart Delay	This value can affect latency.
Outputs	Advanced > GOP Size	This value can affect latency because the segment length is a function of the GOP size.
	Advanced > Closed GOP Cadence	This value can affect latency.

Sending Elemental Live output to AWS Elemental MediaStore

In AWS Elemental Live , you can set up a container on AWS Elemental MediaStore as the destination for Apple HLS and DASH outputs.

This article assumes you know how to use the AWS Management Console and AWS Identity and Access Management, and that you have access to IAM User Guide.

This article assumes that you or someone in your organization has created the MediaStore container where Elemental Live will deliver the output. Make sure you have the path to this container or containers.

Step A: Set up Elemental Live in AWS Identity and Access Management

You must use the IAM (AWS Identity and Access Management) service to set up Elemental Live as an AWS user (the "Elemental user") and give it permissions so that it can communicate with MediaStore. You must do the following:

- Create a policy that contains specific permissions.
- Create the Elemental Live user in your AWS account. The user must be in the same AWS account as the user who is operating AWS Elemental MediaStore.
- Associate the Elemental Live user with the policy, which gives the user the permissions of that policy.

You perform this setup only once. You can use the same "Elemental user" every time you want to send output to MediaStore.

Create a policy for Elemental Live to Make Requests to MediaStore

Elemental Live must have permissions on MediaStore. Follow this procedure to set up these permissions:

1. Open the IAM console at https://console.aws.amazon.com/iam/.

2. On the left menu, choose **Policies**. Use the filters to determine if there is already a policy with a name similar to `ElementalAccessToMediaStore`.

3. If the policy does not exist, choose **Create policy**. Choose the **Visual editor** tab and create the policy using the IAM policy generator. This generator lets you choose the service from a list and then choose operations from a list:
   - Service: **MediaStore**
   - **Actions**:Under **List**, choose **DescribeContainer**,
   - **Actions**: Under **Read**, choose **GetObject**, **DescribeObject**, **GetContainerPolicy**.
   - **Actions**: Under **Write**, choose **PutObject**.
   - **Resources**: If your organization does not have strict rules about accessing containers on MediaStore, you can ignore this section; you will have access to all containers. Otherwise, follow your internal policies to identify specific containers.
   - Give the group a name such as `ElementalAccessToMediaStore`.

For detailed instructions on creating a policy, see [IAM User Guide Creating IAM Policies](#).

Create a user

Follow this procedure to create a user:

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. On the left menu, choose **User**. Use the filters to determine if there is already a user for Elemental products. The user might be called `ElementalUser`.
3. If the user does not exist or it does exist but you want to create separate users for each Elemental product, choose **Add User**. (Note that you may want separate users for separate *products*, but there is probably no need to create a separate user for each Elemental *node*.) Follow the prompts to add the user with this information:
   - Give the user a name such as `ElementalUser`.
   - For **Access type**, choose **Programmatic access**. Do not choose **Console access**.
   - In permissions, choose **Attach existing policies directly**. Attach the policy you created above. For example, `ElementalAccessToMediaStore`.
   - Ignore tags.
4. Create the user and choose **Close**.
5. On the left menu, choose **Users** again:
   - Choose the user name, for example, `ElementalUser`.
   - Choose the **Security** tab.

- Choose **Create Access Key**.
- On the **Create access** key dialog, choose to download the `.csv` file. Save the file in a safe place, so that you have a permanent record of the `Access key ID` and the `Secret access key`.

  The Access key ID looks like this:KIAIOSFODNNYEXAMPLE

  The Secret access key looks like this: 94afd1f2e64d908bc90dbca0035a5b567EXAMPLE

6. Give the `Access key ID` and the `Secret access key` to the Elemental Live operator. Do *not* give the username and password to the operator.

This creates an AWS user with the permissions required to let Elemental Live make requests to MediaStore. When the Elemental Live operator sets up the output with MediaStore as the destination, they will enter the `Access key ID` and `Secret access key`. When the Elemental Live event is running, Elemental Live sends these two IDs to the AWS service, instead of sending the user name and password. These IDs provide authorization to AWS for the Elemental Live node to make requests to MediaStore.

Step B: Create the Elemental Live output group

To set up MediaStore as the destination in the HLS or DASH output group:

1. Obtain the endpoint for the MediaStore container where you want to send the output. For more information, see [AWS Elemental MediaStore User Guide Viewing the Details for a Container](#).

   The following is an example of an endpoint:

   ```
   https://w9710g.data.mediastore.us-west-2.amazonaws.com
   ```

2. In the Elemental Live event, go to the HLS or DASH output group and in **Output** > **HTTP Push Dialect** field, choose **AWS Elemental MediaStore**.
3. Complete the fields that appear: **Retry Interval**, **Num Retries**, **FileCache Size**, **Restart Delay**, **Log Uploads**.

   For details on these fields, see the field tooltips.

4. In the output group > **Output** > **Destination** field (above the HTTP Push Dialect field), enter the destination in the format `<endpoint>/path/file/`.

For example: `https://w9710g.data.mediastore.us-west-2.amazonaws.com/sports/curling/`, where `https://w9710g.data.mediastore.us-west-2.amazonaws.com/` is the endpoint for the MediaStore container and `/sports/curling/` is the name of the MediaStore object.

5. Choose the **Lock** icon. Two more fields appear.

   - **Username/Access Key ID**: The Access key ID you created in IAM. For example, AKIAIOSFODNNYEXAMPL
   - **Password/Secret Access Key**: The Secret access key you created in IAM. For example, wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

Repeat the preceding steps to create a second output in this output group, if applicable.

Configuring SMPTE 2110 outputs

You can include one or more SMPTE 2110 outputs in an AWS Elemental Live event. You can optionally configure the output to implement SMPTE 2022-7, so that the output delivers identical content to two destinations, which improves resiliency for the downstream system.

For general information about SMPTE 2110 video content, SMPTE 2022-7, NMOS, and SDP files, see the section called "SMPTE 2110 inputs and outputs".

> **Note**
>
> You can't use AWS Elemental Conductor Live to produce SMPTE 2110 outputs that use NMOS. You can use AWS Elemental Conductor Live to if you're not using NMOS.

Topics

- Step 1: Get ready
- Step 2: Design the workflow
- Step 3: Create SMPTE 2110 output group
- Step 4: Download and post the SDP file

Step 1: Get ready

Before you set up a SMPTE 2110 output in an event, read the following information:

- the section called "Setup: Remove bonded interfaces"
- the section called "Setup: Reserve processing cores"
- the section called "Setup: Enable PTP"

Step 2: Design the workflow

Before you start to create the SMPTE 2110 output group, plan the contents, to make sure that you observe the rules for the number of video, audio, and ancillary data streams that you can include.

Topics

- Design the video
- Design the audio
- Design the ancillary data

Design the video

Decide if you want the output video to be uncompressed or JPEG XS.

The output group can contain only one video output. This Elemental Live *output* represents one SMPTE 2110 *stream*. If you need multiple SMPTE 2110 streams, you must create one SMPTE 2110 output group for each.

Design the audio

> **Note**
>
> Pay attention to the terminology in this section because there is terminology from SMPTE 2110 and there is terminology from Elemental Live.
> A Elemental Live *output* is a SMPTE 2110 *stream*.
> A Elemental Live *stream* is a SMPTE 2110 *essence*.
> A Elemental Live *stream* contains *encodes*.

1. Decide how many audio outputs you need. This Live *output* represents one SMPTE 2110 *stream*.

   Follow these guidelines:

   - You need a separate output for each variation of audio codec and bit depth/sample rate combination. For example:
     - One output for PCM audio, and one output for Dolby Digital.
     - One output for Dolby Digital Plus, and one output for Dolby Digital Passthrough.
     - One output for Dolby Digital with a bit depth of 24 and another output for Dolby Digital with a bit depth of 16.

     You don't need separate outputs for different channel configurations (PCM) or coding mode (Dolby Digital codecs). For example, in a PCM output, you can include one encode that is PCM 6-channel and one encode that is PCM 2-channel.
   - All audio encodes that are in the same output must follow these rules:
     - They must have the same sample rate.
     - They must have the same bit depth.

     Therefore, for example, you need separate outputs if you want to produce one Dolby Digital 5.1 with 24-bit depth and another Dolby Digital 5.1 with 16-bit depth.
2. In each output, decide how many encodes you need. You need one encode for each PCM channel configuration or Dolby Digital coding-mode.

   For example, to include two PCM stereo channels (perhaps one English and one French), you need two encodes.

   Or to include one Dolby Digital 5.1 and one Dolby Digital stereo pair, you need two encodes.

Result of these decisions

You will have a list of outputs, streams, and encodes. For example:

- One PCM output containing one Elemental Live stream. The stream contains eight 2-channel encodes.
- One Dolby Digital output containing one Elemental Live stream. The stream contains one encode, for 5.1 audio.

Design the ancillary data

In the ancillary data, you configure the output to include or exclude embedded captions and SCTE 104 messages.

Embedded captions

1. If your input includes captions, you can convert those captions to embedded captions and include them in the ancillary data output. For detailed information about setting up captions, see the section called "Setting up for captions".
2. If you decide to include embedded captions, identify the line where you want them to appear in the VANC (vertical ancillary data space) of the video stream. If you aren't sure which line to use, speak to the downstream system.

SCTE 104 messages

1. Decide if you want to include ad avail messages in the output. These messages can come from two sources:
   - SCTE 35 or SCTE 104 ad avail messages already present in the event input. When Elemental Live ingests the input, SCTE 104 messages are always converted to SCTE 35 messages.
   - SCTE 35 messages that you insert when the event is running, using the API.

   In both cases, Elemental Live converts the messages to SCTE 104 messages in the SMPTE 2110 output. For general information about ad avail handling in Elemental Live, see the section called "SCTE-35 and SCTE-104 message processing".
2. If you decide to include ad avail messages, identify the line where you want them to appear in the VANC of the video stream.

Step 3: Create SMPTE 2110 output group

In a SMPTE 2110 output group, you create one output for the single video, for each audio, and for the single (optional) ancillary data stream.

Topics

- Set up the ancillary data stream

General procedure

Follow this procedure to create an SMPTE 2110 output. This procedure applies if your organization uses NMOS IS-04 and IS-05, and if it doesn't.

> **Note**
>
> Pay attention to the terminology in this section because there is terminology from SMPTE 2110 and there is terminology from Elemental Live.
> An Elemental Live *output* is a SMPTE 2110 *stream*.
> A Elemental Live *stream* is a SMPTE 2110 *essence*.
> A Elemental Live *stream* contains *encodes*.

To configure a SMPTE 2110 output

To create a SMPTE 2110 output on the web interface, you create one SMPTE 2110 output group that contains one output for video, one output for audio, and (optionally) one output for ancillary data. The video output contains one stream that contains one video.

1. On the web interface for the event, scroll down to the **Additional Global Configuration** section.

   Open the section and choose **Low Latency Mode**. We recommend that you enable this field for any event that includes SMPTE 2110 outputs.
2. Scroll down to the **Output Groups** section and choose the **SMPTE 2110** tab.
3. In the output group section, complete the **NMOS Control** field:
   - Check this field if your organization has an NMOS registry.
   - Otherwise, leave the field unchecked.
4. Choose **Add Output** on the far right of the page. A new output section appears, with a reference to one Elemental Live stream. For example, *Stream 1*. Assume that this stream is for video.
5. Choose **Add Output** again for each audio output you want to include. You should have identified the output you need when you designed the audio output. Each time you add an

output, a new output section appears, with a reference to one Elemental Live *stream*. For example, *Stream 2*.

6. Optionally, choose **Add Output** again to create an output for ancillary data. A new output section appears, with a reference to one Elemental Live stream. For example, *Stream 3*.

   Scroll down to the **Streams** section. There is one Elemental Live stream for each output you created.

7. To set up the video stream (for example, Stream 1), hover over **Audio 1** and click **x**. The stream now contains only a video stream.

8. To set up each audio stream (for example, Stream 2), hover over **Video** and click **x**. The stream now contains only an audio stream. If you scroll back up to the output that references this stream, you'll see that there is now an **Audio Settings** section.

9. To set up the ancillary data stream (for example, Stream 3), hover over **Audio 1** and click **x**. Then click **Caption +**. The stream now contains a video stream and a captions stream. If you scroll back up to the output that references this stream, you'll see that there is now an **Ancillary Data Settings** section.

10. To set up the content of the video, audio, and ancillary data streams, see the sections the section called “Set up the video stream”, the section called “Set up the audio stream”, and the section called “Set up the ancillary data stream”.

11. When you save the event, Elemental Live creates an SDP file for each output you included in the output group. The files are stored on your appliance at this location:

    ```
    http://locahost/<filename>
    ```

    For information about the file name, see the section called “Step 4: Download the SDP file”.

12. Follow the appropriate step with the SDP files:

    - If you enabled NMOS in the output group, Elemental Live automatically sends the SDP file to your registry, and your registry processes the information in each file. You don't need to do anything with the files.

    - If you didn't enable NMOS, you can download this file so that you can make it available to the downstream system for this output. See the section called “Step 4: Download the SDP file”.

      You must put the files on an HTTP server that is accessible to the downstream system.

Set up the video stream

Follow this procedure to set up the video stream.

To set up the video stream

1. Complete the **Outputs** section:

   - **RTP Payload Type –** Enter a number for the type. For example, 96 for a video stream. If you aren't sure what number to enter, speak to the administrator of the downstream system.

     This number will appear in the **m= line** in the SDP file.

   - **Primary Destination** and **Secondary Destination** – These fields appear only if the **NMOS Control** field is unchecked. Complete the two fields in one of these ways:
     - If you are not implementing seamless protection switching, complete only the primary destination.
     - If you are implementing seamless protection switching, complete both destinations fields, one interface for the primary stream, the other for the secondary stream.

     Enter the unicast or multicast address to deliver the video stream to. For example:

     `rtp://239.x.x.x:5000`

     This information will appear in the SDP file.

   - **Interface** – complete these two fields in one of these ways:
     - If you are not implementing seamless protection switching, complete only **Interface**.
     - If you are implementing seamless protection switching, complete both **Interface** fields, one interface for the primary stream, the other for the secondary stream.

2. Complete the **Streams** section for the video (for example, Stream 1):

   - **Video Codec** – Choose the codec that you identified when you designed the video.
   - **Interlace Mode** – Choose **Progressive** or **Interlaced**.
   - **FourCC** (Uncompressed only) – Choose **s210 (10-bit 4:2:2 packed)**.
   - **Compression Ratio** (JPEG XS only) – Choose the compression that the downstream system has requested.
   - **Resolution - Width** – Enter a value that meets one of the following conditions:
     - A value between 352 and 1152. The value must be divisible by 4.

  - A value between 1160 and 2048. The value must be divisible by 8.
  - A value between 2054 and 4096. The value must be divisible by 16.
- **Resolution - Height** – Enter a value that meets one of the following conditions:
  - If **Interlace Mode** is Progressive – a value of 240 or more. The value must be divisible by 4.

    If **Interlace Mode** is Interlaced – a value of 240 or more. The value must be divisible by 8.

- Complete other fields according to your preference.

Many of the fields in this section appear in the `a=ftmp` line in the SDP file.

Set up the audio stream

To set up the audio stream

Follow this procedure to set up each audio stream.

1. Complete the **Outputs** section in the same way that you completed the section for the video. Make sure that you enter an RTP payload type that is unique in this SMPTE 2110 output group.

   For information about the packet time, see the information after this procedure.
2. Add more encodes to each stream, according to [your design](). For example, if your design includes an output that must contain four stereo PCM channels, you need four encodes in the stream.
3. Complete the **Streams** section for the video (for example, Stream 1):

   - **Audio Source** – Choose the input selector that identifies the source for this audio. You previously set up this selector in the Input section of the event.
   - **Audio Codec** – Choose the codec you identified for this stream when you designed the audio. Choose the same audio codec for all the encodes in this Elemental Live stream.

   - **Channels** (PCM only) – Set the number of channels you identified for this audio when you designed the audio.
   - **Coding Mode** (Dolby Digital and Dolby Digital Plus) – Choose the mode you identified for this audio when you designed the audio.

- **Sample Rate** (PCM only) – See the information after this procedure.
- **Bit Depth** – Choose a value. Choose the same value for all the encodes in this Elemental Live stream.
- Complete other fields according to your preference. For Dolby Digital and Dolby Digital Plus, some of the remaining fields relate to Dolby metadata. For more information, see the section called "Dolby metadata".

Many of the fields in this section appear in the `a=ftmp` line in the SDP file.

Setting the sample rate and packet rate for PCM audio

Complete the **Sample Packet** (in the stream section) and the **Packet Time** field (in the output section). Follow this procedure:

- Identify the number of audio channels in all the encodes in this Elemental Live stream.
- Choose a sample rate and a packet time that both fall within the limits for the number of channels, as specified in the following table.
- Set the same sample rate for every encode in the Elemental Live stream.

Number of audio channels	Maximum sample rate (Hz)	Maximum packet time (microseconds)
Up to 4	96,000	1000
Up to 8	48,000	1000
Up to 32	96,000	125
Up to 64	48,000	125

Setting the packet rate for any of the Dolby codecs

Complete the **Packet Time** field (in the output section). Follow this procedure:

- Identify the number of audio tracks in all the encodes in this Elemental Live stream.

- Choose a packet time that falls within the limits for the number of tracks, as specified in the following table.

Number of audio tracks	Packet time (microseconds)
Up to 3	125 or 1000. We recommend 1000
Up to 24	125

Note that the sample rate is always 48,000 Hz.

Set up the ancillary data stream

To set up the ancillary data stream

You set up the ancillary data stream to include embedded captions. You can optionally configure how you want to handle ad avails. These ad avails might be in the source and/or might be ad avails that you insert using the API.

1. Complete the **Outputs** section in the same way that you completed the section for the video. Make sure you enter an RTP payload type that is unique in this SMPTE 2110 output group.
2. Complete the video stream for this output so that it is identical to the video that you set up in the video output. In our example, the video stream in this ancillary data output is stream 3, and the video in the video output is stream 1.
3. If you want to pass through ad avails, complete these fields in the **Outputs** section:
   - **Enable SCTE 35 Passthrough** – Set the check box to checked.
   - **SCTE 104 messages line number** – Enter the line number that you identified when you designed the ancillary data.
4. If you want include embedded captions in the output, follow this procedure:
   - Make sure that you have set up the event so that the captions in the input are converted to embedded captions in the output (or that the embedded source captions are passed through). For information about setting up captions, see the section called "Setting up for captions".
   - In the **Outputs** section for the ancillary data, for **CEA-608-E captions line number**, enter the line number that you identified when you designed the ancillary data.

Step 4: Download and post the SDP file

If your organization doesn't use NMOS, you must make sure that the downstream system (the SMPTE 2110 receiver) can access them:

- If the location is accessible and doesn't need user credentials, provide the downstream system operator with the full path so that they can download the SDP files.
- If the location where the files are stored isn't accessible from the public internet, you must download the files and put them on a location that is accessible.

If your organization uses NMOS, skip this step.

Location and syntax of the files

Elemental Live puts the SDP files on the HTTP root folder:

```
/opt/elemental_se/web/public/
```

The files are named as follows:

```
SmpteSt2110Output.LiveEvent_<event ID>.OutputGroup_<output group>_2110-
<ID>_<media>_<unique number on node>.sdp
```

For example:

```
SmpteSt2110Output.LiveEvent_123.OutputGroup_curling_smpte_2110.2110-20_video_473
```

The key pieces of information are:

- `123` – The number for this event, unique on this Elemental Live node.
- `curling` – The name you assigned to this output group.
- `2110-20_video` – Identifies this file type. In this example, it's an SDP file for uncompressed video.
- `473` – A number that is unique for this Elemental Live node.

To locate and download the SMPTE 2110 SDP file

1. From your output, select **View Logs**.
2. Select the eme log to view its contents.

3. Within the eme log, search for `Writing SDP file to`. You will find a line for the video, and a line for the audio. Copy the portions that follow `/opt/elemental_se/web/public/` for each line. These are your SDP files. For example:

   ```
   SmpteSt2110Output.LiveEvent_123.OutputGroup_curling_smpte_2110.2110-20_video_
   ```

4. If you don't know the IP address of your Elemental Live appliance, find it now. In the Elemental Live web interface, choose **Settings**. (Don't choose **Input Devices** or **Routers** from the submenu).

   On the **Settings** page, choose the **Network** tab, then choose the **Current Settings** tab. Make a note of the IP addresses in the **Domain Name Servers** section.

5. Download the file using a suitable tool to connect from a computer that is on your network to the Elemental Live appliance. For example:

   - On a PC, use Windows Share protocol. Connect to:

     ```
     \\<IP address>\opt\elemental_se\web\public\
     ```

   - On a Mac, use Samba. Connect to:

     ```
     smb://<IP address>/opt/elemental_se/web/public/
     ```

6. Download the files. Then put the files on an HTTP server that you know the receiver can access.

Delivering TS using the SRT protocol

In AWS Elemental Live , you can deliver a transport stream (TS) output from using the SRT protocol. You can choose to encrypt the content with AES. This SRT delivery option is part of the Reliable TS output group.

> **Note**
>
> The SRT option is intended for sending to a downstream system other than AWS Elemental MediaConnect.
> To send to an SRT flow on MediaConnect, we recommend that you use the Reliable TS output group with the AWS Elemental MediaConnect option. See the section called “MediaConnect”.

There are two ways to configure the transmission, one with Elemental Live as the SRT caller, one with Elemental Live as the SRT listener. (In both cases, Elemental Live is the sender. In other words, don't confuse the sender/receiver roles with the caller/listener roles.)

Typically, the decision about the caller/listener roles is made by the downstream destination:

- If the downstream destination is set up as the listener, then Elemental Live must set up as the caller.
- If the downstream destination is set up as the caller, then Elemental Live must set up as the listener.

The SRT caller always initiates the handshake that precedes successful transmission of the output. The SRT listener accepts or rejects the handshake.

Topics

Getting ready

Before you create an SRT output in your event, perform the following preparation.

- Decide if you want to deliver redundant streams in the output. Each stream has a different destination on the downstream system. The downstream destination initially handles one stream. If that stream fails, the downstream destination can switch to the other stream.

  You should find out if the downstream destination can handle this type of resiliency. If they can't then there is no point to delivering two streams.

  If you do want to deliver two streams, you must make sure that your content provider can send you two copies of the source. The two copies must be completely identical.
- Decide if you want to encrypt the content. If you do, discuss the following with the downstream destination:
  - Discuss the encryption level with the administrator of the downstream system. Both sides must use the same level. You can use AES 128, AES 192, or AES 256. You should provide this information to the downstream destination so that they can set up with the same level.

  - Agree on a passphrase that you will both use for encryption. It can be any length because it's not the encryption key itself, it's just used to generate the encryption key. The passphrase can use an ASCII characters including spaces.
- Discuss the following with the administrator of the downstream destination:
  - Find out the role of the downstream destination in the transmission—either the SRT caller or the SRT listener. If you can't obtain this information, then you could assume that the downstream destination is the listener, which is typically the case.
  - Obtain the IP address and port for each destination for the output. For example, `192.168.1.2:5000`.
  - Make sure that the administrator of the downstream system sets up to allow Elemental Live to access the destination. For example, they might need to open ports on the destination, or allow traffic from the public IP address of Elemental Live.
  - Tell the downstream destination the latency (in milliseconds) that you plan to configure into Elemental Live for packet loss and recovery. Packet recovery is a key feature of SRT. The downstream destination should choose a latency value that is close to the value that you plan to use.
  -
  - Note that Elemental Live doesn't use an SRT stream ID when delivering output.

Creating the output

The setup for the output is identical for both roles (caller or listener), except for the **SRT Connection Mode**

1. In the Elemental Live event, go to **Output Groups**, then to **Reliable TS**.
2. Choose **Add Output** to create an output in this Reliable TS output group.
3. Set **Delivery Protocol** to **SRT**.
4. Set **SRT Connection Mode** to the mode for Elemental Live—**Caller** or **Listener**.
5. Complete the fields for the primary destination:
   - **Primary Destination/Amazon Resource Name**: The IP address and port on the downstream system. For example:

     ```
     srt://192.168.1.2:5000
     ```
   - **Interface**: Optional. See the tooltip.

- **Latency**: Enter the value that you decided to use.
- **Encryption**: Choose **None**, or choose an level.
- **Key Value /Passphrase:** If you chose an encryption level, enter the passphrase that you decided to use.

6. Complete the fields for the secondary destination, if you decided to deliver redundant streams.

Delivering TS output using the Zixi protocol

In AWS Elemental Live , you can deliver TS output using the Zixi protocol. The destination is considered to be a *Zixi broadcaster*. You can choose to encrypt the content with AES. This Zixi delivery option is part of the Reliable TS output group.

> **Note**
>
> The Zixi option is intended for sending to a downstream system other than AWS Elemental MediaConnect.
> To send to a Zixi flow on MediaConnect, we recommend that you use the Reliable TS output group with the AWS Elemental MediaConnect option. See the section called “MediaConnect”.

The Zixi protocol involves two roles—the Zixi *feeder* (also known as the caller) and the Zixi *receiver* (the listener). The Zixi feeder always initiates the handshake that precedes successful transmission of the output. The Zixi receiver accepts or rejects the handshake.

With the Zixi option in the Reliable TS output group, Elemental Live is always the Zixi feeder, which means the downstream system must be the Zixi receiver.

Getting ready

Before you create a Zixi output in your event, perform the following preparation.

- Speak to the administrator of the downstream system and make sure that they can work as the Zixi receiver. If they can't, then you can't deliver using the Zixi option.
- Decide if you want to deliver redundant streams in the output. Each stream has a different destination on the downstream system. The downstream destination initially handles one stream. If that stream fails, the downstream destination can switch to the other stream.

You should find out if the downstream destination can handle this type of resiliency. If they can't then there is no point to delivering two streams.

If you do want to deliver two streams, you must make sure that your content provider can send you two copies of the source. The two copies must be completely identical.

- Decide if you want to encrypt the content. If you do, discuss the following with the downstream destination:
  - Discuss the encryption level with the administrator of the downstream system. Both sides must use the same level. You can use AES 128, AES 192, or AES 256. You should provide this information to the downstream destination so that they can set up with the same level.
  - Agree on an encryption key value. The value is an ASCII representation of hexadecimal bytes. The length must be correct for the encryption level. For example, for AES 128, it must be a 32-character string of 16 hexadecimal bytes.
- Discuss the following with the administrator of the downstream destination:
  - Obtain the IP address and port for each destination for the output. For example, `198.51.100.0:2088`.
  - Make sure that the administrator of the downstream system sets up to allow Elemental Live to access the destination. For example, they might need to open ports on the destination, or allow traffic from the public IP address of Elemental Live.
  - Tell the downstream destination the latency (in milliseconds) that you plan to configure into Elemental Live for packet loss and recovery. Packet recovery is a key feature of Zixi. The downstream destination should choose a latency value that is close to the value that you plan to use.
  -

Creating the output

1. In the Elemental Live event, go to **Output Groups** > **Reliable TS**.
2. Choose **Add Output** to create an output in this Reliable TS output group.
3. Complete the fields in each output as follows:
   - **Delivery Protocol**: Zixi
   - **Destination/Amazon Resource Name**: Enter the IP address (that you obtained from the operator of the downstream system), a colon, and the port (default is 2088). For example:

```
zixi://198.51.100.0:2088
```

- **Interface**: Optional. See the tooltip.
- If the downstream system requires that you authenticate, obtain the username and password from them. Then choose the **Lock** icon and complete the fields that appear.
  - **Username/Access Key ID**: The username.
  - **Password/Secret Access Key**: The password.
- **Stream ID**: Enter the stream ID that you and the operator of the downstream system agreed on. It is required.
- **Latency**: See the tooltip.
- **Encryption**: Choose **None**, or choose an algorithm. The downstream system must support the algorithm you choose.
- **Key Value:** Enter the encryption key that you and the operator of the downstream system agreed on. See the tooltip.

Repeat the preceding steps to create a second output in this output group, if applicable. Use the same user name and password. You can also use the same encryption key, if you are encrypting.

Working with Video

This chapter describes how to set up the video features of AWS Elemental Live.

Topics

Working with color space

You can control how Elemental Live handles the color space in a video source. You can set up an event to perform one of these actions on the color space:

- You can set up to *pass through* the color space. Elemental Live doesn't touch the color space or the color space metadata.

  In this case, you might need to adjust the color space metadata in the input. For example, you might want to do this if you know that the color space metadata is incorrect or missing.
- You can set up to *remove* the color space metadata, because you aren't interested in including it in the outputs. Elemental Live doesn't touch the color space but it removes the metadata.
- You can set up to *convert* the color space itself—to change the pixels in the video.Elemental Live changes both the color space and the color space metadata.

You can set up each output in the event for different handling. For example, you can set up one output to remove the color space metadata, one to convert it to a different color space, and another to convert it to a second different color space.

By default, Elemental Live doesn't convert the color space (in the output) or change the color space metadata (in the input). It passes through the source color space and metadata to the output.

Topics

Color space versus video resolution

Color space refers to the range of color. Elemental Live supports the following color spaces:

- SDR (standard dynamic range)
- HDR (high dynamic range)

Resolution refers to the video pixel count. Elemental Live supports the following resolutions:

- SD (standard definition).
- HD (high definition).
- UHD (ultra-high definition). For UHD, Elemental Live supports resolutions up to 4K.

The following combinations of color space and resolution are typically used:

- SDR color space can be associated with SD, HD, and UHD video.
- HDR color space can be associated with HD or UHD video.

HDR isn't typically associated with SD content, but Elemental Live *does* support this combination.

General information about color space

Following is some general information about color space.

Topics

- Requirements for inputs
- Requirements for outputs
- Support for conversion and passthrough

Definitions

There are four aspects to color space:

- The specific *color space* that applies to the video content. The color space specifies a range of pixel colors that can apply to the content.
- The *color space metadata*, which identifies the color space being used. If this metadata is present, the content is said to be *marked* for a color space.
- The *brightness function* that applies to the color space. The brightness function controls the brightness of each pixel. The brightness is also known as gamma tables, lookup tables (LUT), electro-optical transfer function (EOTF), and transfer function.
- The *brightness metadata*, which identifies the brightness function being used.
- The *display metadata* that applies to the color space. Not all standards have this metadata.

The source video might use a specific *color space* and a specific *brightness function*. The source video might also carry *color space metadata* that describes aspects of the color.

Color space standards

To read this table, find a color space in the first column, then read across to identify the three sets of color data for that color space.

Elemental Live term for the color space	Complies with this color space standard	Complies with this brightness function standard	Complies with this standard for display metadata
601 or Rec_601	SDR rec. 601	BT.1886	Not applicable. This color space doesn't include display metadata.

Elemental Live term for the color space	Complies with this color space standard	Complies with this brightness function standard	Complies with this standard for display metadata
709 or Rec_709	SDR rec. 709	BT.1886	Not applicable. This color space doesn't include display metadata.
SDR 2020	rec.2020	BT.1886	Not applicable. This color space doesn't include display metadata.
HDR10	rec.2020	SMPTE ST 2084 (PQ)	SMPTE ST 2086
HLG or HLG 2020	rec.2020	HLG rec. 2020	Not applicable. This color space doesn't include display metadata.
Dolby Vision 5.0. Only a Dolby Vision-compliant downstream player can handle this color space	A Dolby implementation of the ITP color representation format specified in the Rec. ITU-R BT.2100	SMPTE ST 2084 (PQ)	Proprietary Dolby Vision metadata (RPU), on a per-frame basis
Dolby Vision 8.1. Both a Dolby Vision-compliant and an HDR10-compliant downstream player can handle the color space	rec.2020	SMPTE ST 2084 (PQ)	Proprietary Dolby Vision metadata (RPU), on a per-frame basis, and SMPTE ST 2086 on a per-stream basis

Note that HDR10 and HLG use the same color space. They use different brightness functions and display metadata standards.

Requirements for inputs

Supported input types

AWS Elemental Live can work with the color space in all supported input types.

Input requirements for producing Dolby Vision outputs

There are specific requirements for a source that you plan to convert to Dolby Vision. These requirements are stipulated by Dolby Vision, and relate to the minimal video quality required to produce Dolby Vision outputs that meet the Dolby Vision standard::

- The video source must be HD or 4K resolution. In other words, the source must be 1080p or better.
- The video source must be in the HDR10 color space.

Requirements for outputs

Supported output types

You can include any color space in any video in any supported output type. The main consideration in choosing to convert to a color space in an output is whether the intended downstream player can handle the color space.

Output requirements for HDR10 or Dolby Vision

There are specific requirements for converting to HDR10 or Dolby Vision outputs.

Requirement	Applies to converting to HDR10	Applies to converting to Dolby Vision
Codec must be HEVC.	Yes	Yes
Profile must include the term *Main10*.	Yes	Yes

Requirement	Applies to converting to HDR10	Applies to converting to Dolby Vision
For HD outputs, the event must run on an L800 series appliance.	No	Yes
For 4K outputs, the event must run on an appliance in the L730 series, the L840 series, or the L880 series.	No	Yes
You must obtain a license from the AWS Elemental Support Center Activations. Note that pass through of Dolby Vision doesn't require a license.	No	Yes

Support for conversion and passthrough

Handling of supported color spaces

Elemental Live can read the color space information of any supported color space. It can convert the color space or pass through the color space as follows:

Supported color space	Pass through	Convert
601	Yes	Yes, to any supported color space except Dolby Vision.
709	Yes	Yes, to any supported color space except Dolby Vision.
SDR 2020	Yes	Yes, to any supported color space except Dolby Vision.

Supported color space	Pass through	Convert
HDR10	Yes	Yes, to any supported color space. If you want to convert to Dolby Vision, see the section called "Requirements for inputs" and the section called "Requirements for outputs".
HLG	Yes	Yes, to any supported color space except Dolby Vision.
Dolby Vision 5.0	No. See the note after this table.	No.
Dolby Vision 8.1	Yes	No.
Unsupported color space	Yes	No.

Ingesting Dolby Vision 5.0

Elemental Live can't ingest video in the Dolby Vision 5.0 color space. An event with this type of input will fail immediately when ingest starts.

Handling of unsupported color spaces

We can't make any promises about handling of video that uses an unsupported color space. Any of the following might apply:

- Elemental Live might be able to ingest the input and pass through the color space and the color space metadata.
- Or it might ingest the input but produce unacceptable output.
- Or it might fail to ingest the input, so that the event follows the input loss behavior routine (for example, it might display a slate in the output).

Elemental Live is never able to convert an unsupported color space to another color space.

Configuring the handling in the input

You must decide what you need to do with *color space metadata* in the input. You might need to clean up the metadata to ensure that Elemental Live handles the color space correctly in the output.

> **Important**
>
> Keep in mind that the handling on the input side of the event is about changing the color space metadata, not changing the color space itself. It is about changing the metadata to correctly identify the color space in the input, in preparation for planned handling in the outputs.
> The conversion of the video to a different color space occurs in the section called "Configuring output".

If you plan to *pass through* the color space to the outputs, you should do one of the following:

- Clean up the color space metadata, if the content provider tells you that it is missing or inaccurate.
- Leave the metadata as is, if the color space metadata is correct.

If you plan to *convert* the color space in the outputs, you should do one of the following:

- Clean up the color space metadata, if the content provider tells you that it is missing or inaccurate.
- Leave the metadata as is, if the color space metadata is correct.

If you plan to remove the metadata, there is no need to work with the color space metadata in the input.

The following table specifies the handling that is available for color spaces in the input.

Color space	Elemental Live can correct the color space metadata
601	Yes

Color space	Elemental Live can correct the color space metadata
709	Yes
SDR 2020	Yes
HDR10	Yes
HLG	Yes
Dolby Vision 5.0	No
Dolby Vision 8.1	No
Unsupported color space	No

To decide how to handle the color space metadata, use the following three steps.

Topics

Step 1: Decide on the input handling

If you plan to pass through or convert the color space on the output side of the event, you must decide whether you need to change the color space metadata in the input.

> **Note**
>
> If you plan to remove the color space metadata from *all* the outputs, there is no need to review the inputs. Skip to the section called "Converting: Procedure A".

To decide how to handle the color space metadata

Follow this procedure for each input.

1. Contact the content provider of the input source to find out the following information:

   - Whether the input source contains a combination of different color spaces.
   - The name of the color space or color spaces in the input source.
   - Whether the color space metadata is accurate. It is accurate if it correctly identifies the color space and if it isn't missing. The content is most likely to be one of the following:
     - Correctly marked.
     - Unmarked (no color space metadata is present). In this case, try to find out what the probable color space is.

     The content might also be one of the following:

     - Incorrectly marked.
     - Marked as *unknown*.
     - Marked with a color space that Elemental Live doesn't support and therefore doesn't read.
   - If the color space metadata isn't accurate or is missing, and if the color space is HDR10, obtain the values that the content provider intends for the HDR10 master display metadata.
   - If you plan to convert the color space to HDR10 or Dolby Vision, find out whether the source video is Full Range or Video Range. You will need this information in order to correctly set the **Video Range** field in the output.

   If the content provider can't provide accurate information about the color space or its metadata, you might choose at this point to remove the color space metadata. Move on to the next input. Or if this is the only input, stop reading this section and go to the section called "Configuring output".

2. Make a note of the information:

   - The names of the color spaces.
   - If applicable, the values for the HDR10 display metadata.
   - If applicable, whether the source video is full range or video range.

Step 2: Choose a clean-up scenario

Read the following scenarios to decide if you need to clean up the color space metadata in each input.

The following rules apply:

- If you are planning to convert the color space on the output side, keep in mind that the conversion will apply only to marked content. So if you use cleanup to insert missing metadata, you can increase the percentage of the content that gets converted in the output.
- The cleanup options don't convert the color space. Instead, cleanup options work on the metadata that is attached to the video.
- You should only clean up the color space if you are sure that all the unmarked portions use the color space that you choose. If the cleanup results in marking content as being in a specific color space when it isn't, then the video color quality will be degraded in the output.

Topics

Scenario A – Pass through accurate metadata

The details of this scenario are the following:

- Intended handling in the output – Pass through the color space.
- Status of the input – The video content is any combination of color spaces—SDR, HDR, or both.
- Status of the input color space metadata – The metadata is correct.

Recommendation:

- **Color Space** field – Set to **FOLLOW**
- **Force Color** field – Elemental Live ignores this field.

During ingest, Elemental Live will retain (pass through) the metadata.

Scenario B – Convert accurately marked color space

The details of this scenario are the following:

- Intended handling in the output – Convert the color space and metadata.
- Status of the input color space – The video content is any combination of color spaces—SDR, HDR, or both.
- Status of the input color space metadata – The metadata is correct.

Recommendation:

- **Color Space** field – Set to **FOLLOW**.
- **Force Color** field – Elemental Live ignores this field.

During ingest, Elemental Live will retain (pass through) the metadata.

Scenario C – Remove metadata

The details of this scenario are the following:

- Intended handling in the output – Remove the color space metadata.
- Status of the input – The video content is any combination of color spaces—SDR, HDR, or both.
- Status of the input color space metadata – The metadata can be of any quality.

Recommendation:

- **Color Space** field – Set to **FOLLOW**.
- **Force Color** field – Elemental Live ignores this field.

During ingest, Elemental Live will retain (pass through) the metadata. You plan to remove the metadata, so you don't care about its quality.

Scenario D – Correct the metadata

The details of this scenario are the following:

- Intended handling in the output – Convert or pass through the color space.

- Status of the input – The video content is one color space. For example, the content is all REC_601.
- Status of the input color space metadata – Some of the metadata is missing, marked as *unknown*, or marked as a color space that Elemental Live doesn't support.

  In addition, some of the metadata is wrong. For example, it is marked as HDR10, but in fact, it is REC_601.

So in this scenario, the video content is all one color space, but the color space metadata doesn't correctly indicate that fact.

Recommendation:

- **Color Space** field – Set to the color space that applies to the video content.
- **Force Color** field – Set to **FORCE**.

During ingest, Elemental Live will create metadata of the specified color space for all missing, unmarked, and unknown metadata.

It will also force all existing metadata to match the specified color space. Therefore, all the content in the input will be consistently marked as belonging to one color space.

Scenario E – Correct the metadata in one color space

The details of this scenario are the following:

- Intended handling in the output – Convert or pass through the color space.
- Status of the input – The video content is any combination of color spaces—REC_601, REC_709, HDR, and HLG.
- Status of the input color space metadata – The metadata for the video content of one color space is a mixture of acceptable and unacceptable. The metadata for that content is missing, marked as *unknown*, or marked as a color space that Elemental Live doesn't support. But in fact, all that content should be marked as one specific color space, for example, as REC_601.

  The metadata for content for any other color space is correct. For example, the metadata for REC_709 content and HDR10 content is correct.

Recommendation:

- **Color Space** field – Set to the color space that has unacceptable metadata.
- **Force Color** field – Set to **FALLBACK**.

During ingest, Elemental Live will create metadata of the specific color space for all missing, unmarked, and unknown video content. It will retain existing metadata.

If you clean up the metadata in this way, Elemental Live might be able to handle the color space appropriately in the output. However, if the color map of the output is wrong in whole or in part, the video source was probably in a color space that Elemental Live can't handle.

Scenario F – Correct the metadata in multiple color spaces

The details of this scenario are the following:

- Intended handling in the output – Convert or pass through the color space.
- Status of the input – The video content is in *more than one* color space. For example, the content is a mix of REC_601, REC_709, and HDR10.
- Status of the input color space metadata – The metadata for one color space is missing, wrong, marked as *unknown*, or marked as a color space that Elemental Live doesn't support. For example, the color space is REC_601, but its corresponding metadata is unreliable.

  In addition, the metadata for one or more other color spaces is also missing, wrong, unknown, or not supported. For example, the color space of that content is HLG, but its corresponding metadata is unreliable.

Recommendation:

There is no way to clean up this content because you can only mark all the content as one type of color space. But in this scenario, the metadata is incorrect in different types of color space.

If you force the color space, some of it will be forced to be correct, but some of it will be forced to incorrect information. Inaccurate metadata will result in an inaccurate conversion (if you convert in the output), or in an inferior viewing experience (if you pass through in the output).

The best recommendation we can provide is to remove the metadata on the output side, as described in scenario C.

If you remove the metadata, Elemental Live might be able to handle the color space appropriately in the output. However, if the color map of the output is wrong in whole or in part, the video source was probably in a color space that Elemental Live can't handle.

Step 3: Set up each input

To set up each input in the event

> **Note**
>
> This section assumes that you are familiar with creating or editing an event.

1. On the **Event** page, in the **Input** section, open the **Advanced** section. More fields appear.
2. In the **Video selector** section, set the appropriate values for **Color Space** and **Force Color**.

   In the following table, each row shows a valid combination of the two fields and the result of that combination.

Color Space field	Force Color field	HDR Master Display Information field	Result
FOLLOW	This field is ignored.		Passthrough. Elemental Live doesn't change the color space metadata.
REC_601 or **REC_709** or **HLG**	**Force**		Cleanup. Elemental Live marks all the content as using the specified color space.
HDR10	**Force**	See /the section called "Tips for HDR master display information"	Cleanup. Elemental Live marks all the content as using HDR10, and sets

Color Space field	Force Color field	HDR Master Display Information field	Result
			the metadata to the specified values.
REC_601 or **REC_709** or **HLG**	**Fallback**		Cleanup. Elemental Live marks the content as using the specified color space only for portions that are marked as follows: • Unmarked • Marked as unknown • Marked with an unsupported color space

Color Space field	Force Color field	HDR Master Display Information field	Result
HDR10	**Fallback**	See /the section called “Tips for HDR master display information”	Cleanup. Elemental Live marks the content as using the specified color space only for portions that are marked as follows: • Unmarked • Marked as unknown • Marked with an unsupported color space Elemental Live marks the relevant portions as using HDR10, and sets the metadata to the specified values.

Tips for HDR master display information

The HDR Master Display Information fields appear in the color space fields if you set the **Color Space** field to HDR10. Take the appropriate action:

- Complete these fields only if your plan is to pass through this color space to the output, and only if the content provider has told you that the content currently doesn't include this metadata. For details about a field on the web interface, choose the question mark next to the field.

  If the content provider has told you that the content already contains the metadata, leave these fields blank.

Make sure to obtain values used in the color grading process for the input. You can't use the defaults or null values and expect to obtain valid color results. It's better to set the fields to null values, rather than to make up values.

- Don't complete these fields if your plan is to convert from this HDR10 color space to another color space.

Red, green, blue, white point x and y

Your content provider might provide numbers like this for X and Y points:

- G (x=0.265, y=0.690)
- B (x=0.150, y=0.060)
- R (x=0.680, y=0.320)

You must convert these numbers to numbers like this:

- G (13250, 34500)
- B (7500, 3000)
- R (34000, 16000)

To convert between the two formats, divide each number by 0.00002 as per the HEVC specification.

For example, 0.265 divided by 0.00002 is 13250.

Max luminance and min luminance

The maximum and minimum luminance are given in units of **0.0001 candelas per square meter**. Your content provider might provide this value in candelas per square meter instead. If so, then convert these numbers by multiplying by 10,000, then entering the result in the web interface.

For example, a value of 1000.0000 cd/m2 for max luminance would be converted to 10,000,000 and entered as that in the web interface.

Configuring color space handling in each output

After you have set up each input in the event, you must configure the outputs for the desired handling of color space. You can do the following:

- Convert the color space in the content to a different color space in the output. See the section called “Color space standards” for the supported conversions.
- Remove the color space metadata. Elemental Live doesn't touch the color space itself, it only removes the color space metadata.

  You might choose to remove the color space metadata in situations such as the following:
  - The pixel data and color space data in the input is incorrect, so that the downstream player can't use it to enhance the color.
  - The color space (and its metadata) changes frequently within the input, or between one input and another, and you know that there is a system downstream of AWS Elemental Live that can't handle changes in the metadata.

  Keep in mind that removing metadata doesn't necessarily make the color poorer. Removing it might only mean that the downstream player can't implement enhancements to make the color even richer.
- Pass the color space metadata and the color space through to the output.

You can set up each output with different color space handling. For example, you can create one output that passes through the original color space, and another that converts it.

> **Note**
>
> Elemental Live converts from one color space to another based on the metadata in the content. Elemental Live doesn't examine the video to try to determine whether it actually matches the color space identified in the metadata. Therefore, to successfully convert, the metadata must be as accurate as possible. To correct the metadata, see the section called “Configuring input”.

Topics

- Converting color space: Procedure B
- Removing color space metadata

Passing through color space

You can pass through any color space that Elemental Live supports, except for Dolby Vision 5.0. You can pass through both color spaces that Elemental Live supports, and color spaces that it doesn't support, so long as the output type supports the passed-through color space standard.

> **Note**
>
> This section assumes that you are familiar with creating or editing an event.

To set up each output

Follow this procedure in each output.

1. On the **Event** page, in the **Output groups** section, choose the output group, and choose the output that contains the video.
2. Open the **Advanced** section. More fields appear.
3. Leave **Insert Color Metadata** checked. You should never remove the color metadata if you are passing through the color space.
4. Scroll down to the **Preprocessors** section and turn on **Color Corrector**. More fields appear.
5. Set **Color Space Conversion** to **None**, which means you don't want to convert the color space.

The following table shows how Elemental Live handles each type of color space that it encounters. Each row in the table describes a different handling.

Color space metadata that Elemental Live encounters	How Elemental Live handles the color space
Content in any color space that Elemental Live supports	It doesn't touch the color space or brightness (the pixel values) in the output. It passes through any of the three sets of metadata that are present.

Color space metadata that Elemental Live encounters	How Elemental Live handles the color space
Content marked with unknown or an unsupported color space	It doesn't touch the color space or brightness (the pixel values) in the output. It leaves the content as marked with the unknown color space. It passes through any brightness metadata and display metadata.
Content with no color space metadata	It doesn't touch the color space or brightness (the pixel values) in the output. It leaves the content as unmarked (no color space metadata).

Converting color space: Procedure A

Follow this procedure to convert to one of these color spaces:

- 601
- 709
- SDR2020
- HLG

For information about the source color spaces that you can convert to one of these color spaces, see the section called “Support for conversion and passthrough”.

For information about the results of conversion, see the section called “The results of different types of conversions”.

> **Note**
>
> This section assumes that you are familiar with creating or editing an event.

To set up each output

Follow this procedure in each output.

1. On the **Event** page, in the **Output groups** section, choose the output group, and choose the output that contains the video.
2. Open the **Advanced** section. More fields appear.
3. Scroll down to the **Preprocessors** section and turn on **Color Corrector**. More fields appear.
4. Complete fields in the **Video** section as described in the following table

Field	Description
Video Codec	Choose any codec.
Advanced, then **Insert Color Metadata**	Leave this field checked. You should never remove the color metadata if you are converting the color space.
Video Range	Choose the correct option. For details, choose the icon above the field.
Preprocessors, then **Color Space Conversion**	Choose the correct conversion: **Force 601** **Force 709** **Force SDR2020** **Force HLG 2020**

Converting color space: Procedure B

Follow this procedure to convert to one of these color spaces:

- HDR10
- Dolby Vision

For information about the source color spaces that you can convert to one of these color spaces, see the section called "Support for conversion and passthrough".

For information about the results of conversion, see the section called "The results of different types of conversions".

To set up each output

> **Note**
>
> This section assumes that you are familiar with creating or editing an event.

Follow this procedure in each output.

1. On the **Event** page, in the **Output groups** section, choose the output group, and choose the output that contains the video.
2. Open the **Advanced** section. More fields appear.
3. Scroll down to the **Preprocessors** section and turn on **Color Corrector**. More fields appear.
4. Complete fields in the **Video** section as described in the following table.

Field	Description
Video Codec	If you are converting to HDR10, choose **MPEG-4 AVC (H.264)** or **HEVC (H.265)**. If you are converting to Dolby Vision, choose **HEVC (H.265)**.
Advanced, then **Insert Color Metadata**	Leave this field checked. You should never remove the color metadata if you are converting the color space.
Advanced, then **Profile** This field is towards the end of the **Advanced** section	Choose a profile that includes the term **Main10**.

Field	Description
Preprocessors, then **Video Range**	Choose the correct option, according to the information you obtained from the content provider: • If the video input is full range, choose **Passthrough**. • If the video input is video range, choose **Full Swing**.
Preprocessors, then **Color Space Conversion**	Choose the correct conversion: **Force HDR10** **Dolby Vision Profile 5** **Dolby Vision Profile 8.1**
Preprocessors, then **HDR Master Display Information**	These fields appear after you complete the **Color Space Conversion**. You can optionally complete the **HDR Master Display Information** fields. For information about master display information, see the section called "Tips for HDR master display information".

Tips for HDR master display information

The HDR Master Display Information fields appear if you are converting to HDR10 or Dolby Vision. Take the appropriate action:

- If you have previously converted similar content to HDR10 and a color grading specialist in your organization has given you metadata, then enter it here. The values to enter here depend on the downstream player, so there is no point to asking your content provider for values.

  For details about a field on the web interface, choose the question mark next to the field.

- If you don't have metadata to use, set all fields to null values. It's better to set the fields to null values, rather than to make up values or to use the default values.

Red, green, blue, white point x and y

Your color grader might provide numbers like this for X and Y points:

- G (x=0.265, y=0.690)
- B (x=0.150, y=0.060)
- R (x=0.680, y=0.320)

You must convert these numbers to numbers like this:

- G (13250, 34500)
- B (7500, 3000)
- R (34000, 16000)

To convert between the two formats, divide each number by 0.00002 as per the HEVC specification.

For example, 0.265 divided by 0.00002 is 13250.

Max luminance and min luminance

The maximum and minimum luminance are given in units of **0.0001 candelas per square meter**. Your color grader might provide this value in candelas per square meter instead. If so, then convert these numbers by multiplying by 10,000, then entering the result in the web interface.

For example, a value of 1000.0000 cd/m2 for max luminance would be converted to 10,000,000 and entered as that in the web interface.

Removing color space metadata

Follow this procedure in each output where you want to remove the color space metadata.

You might choose to remove the color space metadata in situations such as the following:

- The pixel data and color space data in the input is incorrect, so that the downstream player can't use it to enhance the color.

- The color space (and its metadata) changes frequently within the input, or between one input and another, and you know that there is a system downstream of AWS Elemental Live that can't handle changes in the metadata.

Keep in mind that removing metadata doesn't necessarily make the color poorer. Removing it might only mean that the downstream player can't implement enhancements to make the color even richer.

> **Note**
>
> This section assumes that you are familiar with creating or editing an event.

To set up each output

1. On the **Event** page, in the **Output groups** section, choose the output group, and choose the output that contains the video.
2. Open the **Advanced** section. More fields appear.
3. Set **Insert Color Metadata** to unchecked.
4. Scroll down to the **Preprocessors** section and turn on **Color Corrector**. More fields appear.
5. Set **Color Space Conversion** to **None**, which means you don't want to convert the color space.

The following table shows how Elemental Live handles each type of color space it encounters.

<table>
<tr><th>Color space metadata that Elemental Live encounters</th><th>How Elemental Live handles the color space</th></tr>
<tr><td>Content in any color space that Elemental Live supports</td><td rowspan="3">It doesn't touch the color space or brightness (the pixel values) in the output.
It removes all the metadata. The output won't contain any color space metadata, brightness metadata, or display metadata.</td></tr>
<tr><td>Content with no color space metadata</td></tr>
<tr><td>Content with unknown or unsupported color space metadata</td></tr>
</table>

The results of different types of conversions

Following is information about how Elemental Live handles the different types of conversion of the color space.

Topics

Converting one SDR color space to another

You can convert an SDR color space to another SDR color space. In this case, Elemental Live makes the following changes:

- It changes pixels to values that represent the same color as the original values. The video now fits in the larger color space.
- It changes the color space metadata to identify the new color space.
- It applies the same brightness function to the video, because all the SDR color spaces use the same function.

After the conversion, the video complies completely with the new color space.

Converting an SDR color space to HDR

You can convert an SDR color space to HDR10 or HLG color space. In this case, Elemental Live makes the following changes:

- It changes the pixel values, if necessary, to fit the colors into the different color space.
- It changes the color space metadata to identify the new color space.
- It applies the new brightness function to the video.
- If converting to HDR10, it calculates display metadata for the video.

After the conversion, the video fits in the new color spaces, but the color is not any richer than before the conversion. However, the bright parts of the video are brighter, and the dark parts are darker.

Converting one HDR color space to another

You can convert video between the HDR10 color space and the HLG color space, in either direction. In this case, Elemental Live makes the following changes:

- It changes the pixel values, if necessary, to fit the colors into the different color space.
- It changes the color space metadata to identify the new color space.
- It applies the new brightness function to the video.
- If converting to HDR10, it calculates display metadata for the video.

After the conversion, the video complies completely with the new color space. The color will be slightly different, but probably not more or less rich. The color will match the new brightness function.

Converting an HDR color space to SDR

You can convert HDR10 or HLG video to an SDR color space. In this case, Elemental Live makes the following changes:

- It changes the pixel values, if necessary, to fit the colors into the smaller color space.
- It changes the color space metadata to identify the new color space.
- It applies the new brightness function to the video.
- It removes any display metadata because the SDR color spaces don't include display metadata.

After the conversion, the video complies completely with the new color space. The color will be less rich. The color will match the new brightness function.

Converting HDR10 to Dolby Vision 5.0 or 8.1

You can convert video in the HDR10 color space to Dolby Vision 5.0 or 8.1.

If you convert the video to Dolby Vision 5.0, video players that are Dolby Vision-compliant will be able to play it. If you convert the video to Dolby Vision 8.1, video players that are Dolby Vision-compliant and video players that are HDR10-compliant will be able to play it.

When you convert a suitable video to Dolby Vision, Elemental Live makes the following changes:

- It doesn't change the pixel values, because HDR10 and Dolby Vision both use the same color space.
- It changes the color space metadata to identify the new color space.
- It applies the new brightness function to the video.
- It calculates the Dolby Vision display metadata for the video.

After the conversion, the video fits in the new color spaces, but the color is not any richer than before the conversion, because the color space hasn't changed. However, the bright parts of the video are brighter, and the dark parts are darker.

Converting other color spaces to Dolby Vision 5.0 or 8.1

You shouldn't convert non-HDR10 video to Dolby Vision. For example, you shouldn't convert SDR 601 to Dolby Vision. Converting a non-HDR10 video to Dolby Vision doesn't comply with the usage intended by Dolby Vision. After conversion of the color space, the color map of the video will be completely wrong.

The only color space that you should convert to Dolby Vision is HDR10.

Converting a mixed color space to one color space

The video in your input or inputs might contain a mix of color spaces. You can still set up to convert these color spaces to one color space.

In this case, Elemental Live makes the following changes to the pixel values:

- For color spaces where it supports conversion, Elemental Live changes the pixel values to values that are appropriate to the new color space. See the sections that describe the other conversions.
- It doesn't change the pixel values for video in unsupported color spaces, or video that has no color space metadata. See the section called “Handling of unsupported color spaces” for more information.
- It doesn't change the pixel values for Dolby Vision 8.1 video because Elemental Live doesn't read the color space metadata for Dolby Vision. On the input side, Elemental Live treats Dolby Vision 8.1 as an unknown color space.
- Keep in mind that Elemental Live can't ingest Dolby Vision 5.0, so the handling is irrelevant.

Elemental Live makes the following changes to the metadata:

- If it converts the color space, it changes the color space metadata to identify the new color space. It applies the new brightness function to the video.
- If it leaves the color space unchanged, it also leaves the metadata unchanged.

Location of HDR fields on the web interface

- Input > Video Selector > Color Space
- Input > Video Selector > Force Color
- Input > Video Selector > HDR Master Display Information group of fields
- Stream > Video > Video Codec
- Stream > Video > Insert Color Metadata (near the top of the tab)
- Stream > Video > Profile (near the bottom of the tab)
- Stream > Video > Preprocessors > Color Corrector > Color Space Conversion

Location of HDR fields in the XML

Input > Video Selector > Color Space	input/video_selector/color_space
Input > Video Selector > Force Color	input/video_selector/force_color
Input > Video Selector > HDR Master Display Information group of fields	input/video_selector/hdr10_metadata/blue_primary_x
	input/video_selector/hdr10_metadata/blue_primary_y
	input/video_selector/hdr10_metadata/green_primary_x
	input/video_selector/hdr10_metadata/green_primary_y
	input/video_selector/hdr10_metadata/max_content_light

	input/video_selector/hdr10_metadata/max_luminescence
	input/video_selector/hdr10_metadata/max_picture_avg_light
	input/video_selector/hdr10_metadata/min_luminescence
	input/video_selector/hdr10_metadata/red_primary_x
	input/video_selector/hdr10_metadata/red_primary_y
	input/video_selector/hdr10_metadata/white_point_x
	input/video_selector/hdr10_metadata/white_point_y
Stream > Video > Video Codec	stream_assembly/video_description/codec
Stream > Video > Insert Color Metadata	stream_assembly/video_description/insert_color_metadata
Stream > Video > Profile	stream_assembly/video_description/profile
Stream > Video > Preprocessors > Color Corrector > Color Space Correction > Color Space Conversion	stream_assembly/video_description/color_corrector/color_space_conversion
Stream > Video > Preprocessors > Color Corrector > Color Space Correction > HDR Master Display Information group of fields	stream_assembly/video_description/color_corrector/hdr10_metadata/blue_primary_x
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/blue_primary_y

	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/green_primary_x
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/green_primary_y
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/max_content_light
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/max_luminescence
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/max_picture_avg_light
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/min_luminescence
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/red_primary_x
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/red_primary_y
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/white_point_x
	stream_assembly/video_description/video_preprocessors/color_corrector/hdr10_metadata/white_point_y

Dolby Vision color space

The information for handling video that is in the Dolby Vision color space has changed. AWS Elemental Live no longer requires source video to contain the RPUs that are used to produce some of the color space metadata. Now, Elemental Live produces the metadata without relying on external RPUs.

In addition, the information for Dolby Vision has been integrated into the standard section on color space, the section called “Color space”.

Setting up QVBR and rate control mode

The Rate Control Mode fields in the Video section of each output on the Elemental Live web interface let you control the quality and bitrate of the video.

When encoding visually complex video (such as high-motion sports events with brightly dressed crowds in the background), there is always a trade-off between high video quality and low bitrate. Higher video quality requires higher bitrate. There is less trade-off with visually simple video such as cartoons.

Elemental Live offers several options that provide different balances of video quality versus bitrate.

Do the following to set the rate control mode and bitrate for the output:

- In **Event** > **Streams** > **Video** > **Video Code**, choose **H.264**
- In **Event** > **Streams** > **Video** > **Advanced** > **Rate Control Mode for Video**, for **Codec settings**, select the desired option. For information about choosing the best option, see the sections below.
- Complete other fields, as appropriate:
  - If you select **QVBR**, complete **Max bitrate**, **Quality level**, **Buffer size**, and **Buffer fill percentage**.
  - If you select **VBR**, complete **Bitrate (average bitrate)**, **Max bitrate**, **Buffer size**, and **Buffer fill percentage**.
  - If you select **CBR**, complete **Bitrate**, **Buffer size**, and **Buffer fill percentage**.

> **Note**
>
> We do not recommend use of either the CQ or ABR options. The Statmux option only applies if you have the Statmux option enabled on Elemental Live and only if you are creating a "statmux output".

Quality-defined variable bitrate mode (QVBR)

With quality-defined variable bitrate mode (QVBR), you specify a maximum bitrate and a quality level. Video quality will match the specified quality level except when it is constrained by the maximum bitrate. This constraint occurs when the video is very complex, so that it is not possible to reach the quality level without exceeding the maximum bitrate.

We recommend this mode if you or your viewers pay for bandwidth, for example, if you are delivering to a CDN or if your viewing users are on mobile networks.

When selecting QVBR, set the quality level and maximum bitrate for your most important viewing devices. Set the buffer size to twice the maximum bitrate, and set the initial buffer to 90%. Set the Buffer size to twice (2x) the Max bitrate and set the Buffer fill percentage to 90%. The buffer contributes to a smooth playout.

The following table lists the recommended values for this mode:

Viewing device	Quality level	Max bitrate
Primary screen	8 to 10	4,000,000 to 6,000,000
PC or tablet	7	1,500,000 to 3,000,000
Smartphone	6	1,000,000 to 1,500,000

With this mode, the bitrate can change with each frame (in order to obtain at least the specified quality), but it cannot exceed the maximum bitrate. Elemental Live does not attempt to maintain an average bitrate. It always hits the maximum bitrate if that is necessary to obtain the specified quality. On the other hand, if the quality can be obtained with lower bitrates, Elemental Live does not use a higher bitrate.

Variable bitrate mode (VBR)

With variable bitrate mode (VBR), you specify an average bitrate and a maximum bitrate. Video quality and bitrate vary, depending on the video complexity.

Select VBR instead of QVBR if you want to maintain a specific average bitrate over the duration of the channel. If bitrate does not need to be constrained, then consider using QVBR.

When selecting VBR, try to assess the expected complexity of the video, and set a suitable average bitrate. Set the maximum bitrate to accommodate expected spikes. Set the Buffer size to twice (2x) the Max bitrate and set the Buffer fill percentage to 90%. The buffer contributes to a smooth playout.

With this mode, the bitrate can change with each frame (in order to obtain the best quality) but it cannot exceed the specified maximum bitrate. Elemental Live also ensures that as the channel progresses, the stream meets the specified average bitrate. This mode is useful when you expect short spikes in the complexity of the video. Elemental Live aims for the average bitrate but spikes to the maximum bitrate for a short time when necessary.

Constant bitrate mode (CBR)

With constant bitrate mode (CBR), you specify a bitrate. Video quality varies, depending on the video complexity.

Select CBR only if you distribute your assets to devices that cannot handle variable bitrates. But if it is acceptable for the bitrate to occasionally differ from a specified rate, then consider using VBR or QVBR instead. Over the duration of the channel, you might obtain both a lower bitrate and better quality with VBR or QVBR.

When selecting CBR, set the bitrate to balance the video quality and the output bitrate. Set the Buffer size to twice (2x) the Max bitrate and set the Buffer fill percentage to 90%. The buffer contributes to a smooth playout.

With this mode, the output always matches the specified bitrate. Sometimes that bitrate results in higher-quality video, and sometimes it results in lower-quality video.

Converting the scan type

You can convert the scan type of the input to a different scan type: progressive, interlaced, hard telecine, or soft telecine. You can configure to leave the scan type as is or to convert from one incoming type (or a mix of incoming scan types) to another single type.

Topics

- Key fields for converting scan type
- Secondary fields for converting scan type
- Adaptive field frame controls

Key fields for converting scan type

Description

Configuring for scan type conversion involves setting fields in specific ways. The three key fields to convert the scan type of the input are **Configuration - Deinterlace Mode**, **Configuration - Interlace Mode**, and **Configuration - Telecine**. The following table describes how to set these three key fields to convert a given input to a given output.

Input	Output	Configuration - Deinterlace mode	Configuration - Interlace mode	Configuration - Telecine
Progressive	Progressive	Off	Progressive	None
Interlaced	Progressive	Deinterlace	Progressive	None
Interlaced	Progressive	Adaptive	Progressive	None
Hard telecine	Progressive	Inverse telecine	Progressive	None
Hard telecine	Progressive	Adaptive	Progressive	None
Soft telecine	Progressive	Off	Progressive	None
Mixed	Progressive	Adaptive	Progressive	None
Progressive	Hard telecine	Off	One of the other options	Hard telecine
Hard telecine	Hard telecine	Off	One of the other options	None
Soft telecine	Hard telecine	Off	One of the other options	Hard telecine

Input	Output	Configuration - Deinterlace mode	Configuration - Interlace mode	Configuration - Telecine
Mixed	Hard telecine	Off	One of the other options	Hard telecine
Interlaced	Interlaced	Off	One of the other options	None
Mixed	Interlaced	Off	One of the other options	None
Progressive	Soft telecine	Off	One of the other options	Soft telecine
Hard telecine	Soft telecine	Inverse telecine	One of the other options	Soft telecine
Hard telecine	Soft telecine	Adaptive	One of the other options	Soft telecine
Soft telecine	Soft telecine	Off	One of the other options	Soft telecine
Mixed	Soft telecine	Adaptive	One of the other options	Soft telecine

* Deinterlace Mode is an image processing control. Interlace Mode and Telecine are encoding controls.

Note: Converting the scan type to progressive

If content is not being converted to a higher framerate, the deinterlacer outputs one frame for every two fields in the source content (i.e., 1080i30 content is converted to 1080p30). If the framerate is being doubled (e.g. 29.97 fps to 59.94 fps, 29.97 to 60, or 25 to 50), the deinterlacer converts each field into a frame.

Deinterlace mode

This field applies an initial conversion for certain from/to conversions (as shown in the table above).

- **Deinterlace**: Applies a deinterlace algorithm to content. If Elemental Live detects that the source content is already progressive, no deinterlacing is applied.
- **Inverse Telecine**: Converts hard telecine 29.97i to progressive 23.976p.
- **Adaptive**: Analyzes source content to determine whether to apply the deinterlace or inverse telecine algorithm on source content.

Interlace mode

This field controls video field order and how the scan type is represented in the output bitstream.

- **Progressive**: Encodes output as "progressive."
- **Top Field First** or **Bottom Field First**: Forces field polarity (top or bottom first) to the specified value, reordering fields if the source has a different order and encodes the result as interlaced.
- **Follow (Default Top)** and **Follow (Default Bottom)**: Produces interlaced output, with the output having the same field polarity as the source. Therefore:
  - If the source is "interlaced", the output is interlaced with the same polarity as the source (it follows the source). The output could therefore be a mix of “top field first” and “bottom field first.”
  - If the source is "progressive", the output is interlaced with “top field first” or “bottom field first” polarity (depending on which Follow option you chose).

Telecine

This field appears for MPEG-4 AVC and MPEG-2 only if the Streams > Advanced > framerate field is set to 29.970.

- **Hard**: Produce 29.97i output from 23.976 input.
- **Soft**: Produce 23.976; the player converts this output to 29.97i.

Recommendations

- Converting to progressive output always improves output quality and should be enabled in any use case where progressive output is required or acceptable.

- Interlace coding is inherently less efficient than progressive coding so use interlace coding only for content that has already been interlaced.
- The choice of deinterlacing algorithm is very subjective. **Motion Adaptive Interpolation** is generally recommended as the sharper image quality tends to provide the best perceived quality across a broad set of users. Use the **Force Mode** option for deinterlacing only when the input is known to be interlaced content incorrectly flagged as "progressive."
- Use the **Force Mode** option for **Inverse Hard Telecine** when the input is known to consist entirely of Hard Telecine content.

Location of fields

Location of field on web interface	Location of tag in XML
Streams > Advanced > Telecine	stream_assembly/video_description/*codec*/telecine where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings** • **prores_settings**
Streams > Advanced > Interlace Mode	stream_assembly/video_description/*codec*/interlace_mode where *codec* is one of the following: • **vc1_settings** • **h264_settings** • **mpeg2_settings** • **h265_settings** • **prores_settings**

Location of field on web interface	Location of tag in XML
Streams > Advanced >Preprocessors > Deinterlacer > Deinterlace Mode	stream_assembly/video_description/video_preprocessors/deinterlacer/deinterlace_mode
Streams > Advanced >Preprocessors > Deinterlacer > Deinterlace Algorithm	stream_assembly/video_description/video_preprocessors/deinterlacer/algorithm
Streams > Advanced >Preprocessors > Deinterlacer > Force Mode	stream_assembly/video_description/video_preprocessors/deinterlacer/force

Secondary fields for converting scan type

Description

The scan type of the input can be converted to a different scan type: progressive, interlaced, hard telecine, or soft telecine. You can configure to leave the scan type as is or to convert from one incoming type (or a mix of incoming scan types) to another single type. Configuring for scan type conversion involves setting fields in specific ways.

Force deinterlace

The field applies only when the Deinterlacer Preprocessor is turned on. It deals with issues of badly tagged frames.

- When the **Deinterlace Mode** is Adaptive, set **Force Mode** to Off.
- When the **Deinterlace Mode** is Deinterlace, use **Force Mode** as follows:
  - **Off**: The processor does not convert frames that are tagged in metadata as "progressive." It only converts those that are tagged as some other type.
  - **On**: The processor converts every frame to "progressive" – even those that are already tagged as "progressive." Turn **Force Mode** on only if the input frames are incorrectly marked as "progressive." Do not turn otherwise; processing frames that are already tagged as "progressive" degrade video quality.
- When the **Deinterlace Mode** is Inverse Telecine, use **Force Mode** as follows:
  - Off: The processor monitors presence/absence of the hard telecine field repeat cadence and applies only to hard telecine to progressive conversion on frames that have a distinct cadence.

  - On: The processor still monitors for hard telecine cadence and adapts to cadence, but all frames are converted from hard telecine to progressive using the last detected cadence.

Deinterlace algorithm

The field applies only when the Deinterlacer Preprocessor is turned on and when the **Deinterlace Mode** is set to **Deinterlace** or **Adaptive**. Set it to the desired value:

- Motion adaptive interpolation: Provides for better spatial quality (i.e. produces sharper images).
- Motion adaptive blend: Provides for better temporal quality (i.e. produces smoother motion).
- Low latency interpolation: Performs interpolated line-doubling to allow for lower latency applications.

Location of fields

Location of field on web interface	Location of tag in XML
Streams > Advanced >Preprocessors > Deinterlacer > Deinterlace Algorithm	stream_assembly/video_description/video_preprocessors/deinterlacer/algorithm
Streams > Advanced >Preprocessors > Deinterlacer > Force Mode	stream_assembly/video_description/video_preprocessors/deinterlacer/force

Adaptive field frame controls

Description

The following are the settings and internal algorithms tied to the scan type:

-
- **Picture Adaptive Field Frame (PAFF)**: This setting is automatically enabled on GPU-enabled versions of Elemental Live and automatically disabled on CPU-only versions.
- **Macroblock Adaptive Field Frame (MBAFF)**: This setting is automatically enabled on CPU-only versions of Elemental Live and automatically disabled on GPU-enabled versions.
- **Force Field Pictures**: This field appears only if the codec is H.264 and only affects GPU-enabled versions of Elemental Live.

- **Enabled**: All outputs are forced to use PAFF field picture encoding.
- **Disabled**: Elemental Live switches between PAFF and MBAFF, depending on the content.

Recommendations

- **Force Field Pictures** results in a significant reduction in quality so it should only be used if required for compatibility with specific decoders or playback devices.

Location of fields

Location of field on web interface	Location of tag in XML
Streams – Video > Advanced > Force Field Pictures	stream_assembly/video_description/*codec*/force_field_pictures where *codec* is: **h264_settings**

Setting up for ultra-low latency

If you are sending output to an HTTP server or to AWS Elemental MediaStore, you can produce ultra-low latency outputs if you create a DASH output and enable chunked encoding and chunked transfer.

With chunked encoding and transfer Elemental Live delivers more quickly, without any decrease in quality. When you enable chunked encoding and transfer, Elemental Live divides the video into fragments, with several fragments in each segment. Elemental Live delivers the fragments before the entire segment is complete. Delivery of smaller chunks more frequently can speed up handling by the downstream system, assuming that the downstream system can handle the frequency.

From the Elemental Live side, this feature works with any downstream system that can accept DASH. But you should always discuss your requirements with the downstream system administrator, to make sure that they are prepared to accept the output.

Setup

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To produce an ultra-low latency DASH output, create a DASH output group in the usual way, with these additions:

1. Enable **Chunked Encoding**.
2. Change **Fragment Length** and **Segment Length** if necessary. For information about these fields, choose the ? icon that appears when you hover over the field name.
3. In **HTTP Push Dialect**, choose the appropriate option. The **Chunked Transfer** field appears.
4. Enable **Chunked Transfer**.

Controlling video quality

You can tune video quality in each output in an Elemental Live event. You tune the video quality by setting parameters that are in the output section of the event: in the relevant output group, choose an **Output**, choose a **Stream**, choose the **Video** tab, then open the **Advanced** section.

The following list identifies the parameters that affect video quality. The list also specifies the section in this guide that describes the parameter.

- **Adaptive Quantization (AQ)** the section called "Quantization controls"
- **B-Frames** the section called "Group of pictures (GOP) configuration"
- **Bitrate** the section called "QVBR and rate control mode"
- **Buffer Size** the section called "QVBR and rate control mode"
- **Initial Buffer Fill %** the section called "QVBR and rate control mode"
- **CABAC** the section called "Miscellaneous video tuning parameters"
- **Closed GOP Cadence** the section called "Group of pictures (GOP) configuration"
- **Density vs Quality** the section called "Miscellaneous video tuning parameters"

- **Flicker AQ** the section called “Quantization controls”
- **Framerate** the section called “Frame rate conversion”
- **Framing AQ** the section called “Quantization controls”
- **GOP Mode** the section called “Group of pictures (GOP) configuration”
- **GOP Reference B-Frame** the section called “Group of pictures (GOP) configuration”
- **GOP Size** the section called “Group of pictures (GOP) configuration”
- **Interpolated** the section called “Frame rate conversion”
- **Lookahead** the section called “Miscellaneous video tuning parameters”
- **Max Bitrate** the section called “QVBR and rate control mode”
- **Min I-interval** the section called “Group of pictures (GOP) configuration”
- **Noise Reducer** the section called “Noise reduction”
- **Profile** the section called “Miscellaneous video tuning parameters”
- **Rate Control Mode** the section called “QVBR and rate control mode”
- **Reference Frames** the section called “Group of pictures (GOP) configuration”
- **Scene Change Detect** the section called “Group of pictures (GOP) configuration”
- **Slices** the section called “Miscellaneous video tuning parameters”
- **Softness** the section called “Quantization controls”
- **Spatial AQ** the section called “Quantization controls”
- **Temporal AQ** the section called “Quantization controls”

Topics

Frame rate conversion

Description

Framerate conversion is typically used when producing content for devices that use different standards (for example, NTSC vs. PAL), or different content playback scenarios (for example, film at 24 fps vs. television at 29.97 fps). You can set the frame rate for the video output to a value that suits your workflow requirements. You can choose a standard frame rate, such as 29.97. Or you can define a custom rate.

When you set the frame rate, consider the following related parameters:

- The scan type for the output – interlaced or progressive. The choice of frame rate might be constrained by the scan type conversion you apply. For more information, see the section called “Scan type”.
- Interpolation. The **Interpolated** parameter has values to favor sharpness or smoothness:
  - If **Interpolated** is disabled, the encoder drops or repeats frames, as needed. This results in sharp individual frames.
  - If **Interpolated** is enabled, a weighted average is applied between frames when new frames need to be added as part of the frame rate conversion. This results in smoother motion. For example, when converting from a 24 fps input to 29.97 fps output, the encoder uses an algorithm to average the 4th and 5th frames of the source content. this produces the additional frame that's needed in the output.

  Recommendation: Enable the **Interpolated** parameter if input and output frame rates are close. For example, 24 fps inputs to 25 fps outputs.

Location of parameters

This table shows where the parameters mentioned in this section are located. The first column shows the location on the web interface. The second column shows the location in the event XML.

Location of parameter on web interface	Location of tag in XML
Stream – Video > **Advanced** > **Framerate**	stream_assembly/video_description/*codec*/framerate_numerator

Location of parameter on web interface	Location of tag in XML
	stream_assembly/video_description/*codec*/framerate_denominator where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings** • **prores_settings**
Stream – Video > **Advanced** > **Interpolated**	stream_assembly/video_description/*codec*/interpolate_frc where *codec* is: • **h264_settings** • **mpeg2_settings** • **h265_settings** • **prores_settings**

Group of pictures (GOP) configuration

Description

GOP parameters define the basic pattern of the video stream in terms of how the encoder uses I-, P-, and B- frames.

The parameters that control the GOP structure are the following:

- **GOP Mode**: Choose **Fixed**.

  Recommendation: Always choose **Fixed**. The **Follow** mode is obsolete and is not recommended.
- **GOP Size**: Defines the interval between I-frames.

  Recommendation: When using MPEG-2, the recommended GOP size is up to 30. A size of 15 is also common.

For H.264, the recommendation is to make the GOP size as large as possible while still meeting other encoding requirements. For example, for adaptive bitrate delivery in which a segment size of 6 seconds is used for 29.97 fps outputs, the largest GOP size should be 180 frames.

- **B-Frames**: Defines the maximum run of B-frames. Note that the encoder might sometimes decide to use a smaller run of B- frames within the GOP. It usually does this to produce higher video quality.

  Recommendation: When using H.264 or H.265, enable **GOP Reference B-Frame** to obtain the best quality and set B-frames to a value from 3 to 5 (3 is recommended).

  For other codecs, there is no quality benefit to setting the B-frames to more than 2. For high-motion content, use 0 or 1 to produce the best quality.
- **Closed GOP Cadence**: Defines the number of GOPs across which P- or B-frames are allowed to predict for encoding purposes.

  Recommendation: Set the parameter to **1** for segmented outputs such as adaptive bitrate content in HLS or DASH outputs,

  Set the parameter to **0** for non-segmented outputs, to allow for an open GOP.
- **GOP Reference B-Frame** (H.264 and H.265 only): Enables the use of reference B-frames for GOP structures that have B- frames greater than 1.
- **Min I-interval**: Specifies a minimum number of frames between I-frames for GOP cadence and I-frames for scene change detection. I-frames require more bits than P- or B-frames, so encoding two in quick succession can hurt quality, particularly with small buffer sizes. The encoder enforces the minimum I-interval by shifting the GOP cadence.

  Recommendation: For segmented outputs that require a fixed cadence of I-frames, set this parameter to 0, in order to disable it.
- **Reference Frames**: Applies only to H.265. This parameter defines the number of frames that can be used to define B- and P- frames.

  Recommendation: We recommend that you set this parameter to 3.
- **Scene Change Detect**: Enables an algorithm that determines when a scene change occurs. The encoder inserts an I-frame at each scene change.

Recommendation: Always enable this detection for the best video quality. The only scenario where you might disable this detection is to accommodate the rare case that a set-top box or playback device can't accommodate an additional I-frame within the normal GOP pattern.

Some service providers, encoding vendors, or manufacturers might state that scene change detection should be disabled during encoding of content. Typically, they make this recommendation because their systems require a consistent GOP structure. They assume that the encoder resets the GOP structure when an I-frame is inserted for a scene change, which disrupts the GOP structure.

However, the encoder doesn't disrupt the GOP structure in this way. Instead, the encoder inserts an additional I-frame on a scene change. With this approach, the adaptive bitrate outputs are always GOP-aligned. This alignment has the side effect of allowing for compatibility with these third-party systems, even when scene change detection is enabled.

Location of parameters

This table shows where the parameters mentioned in this section are located. The first column shows the location on the web interface. The second column shows the location in the event XML.

Location of parameter on web interface	Location of tag in XML
Streams – Video > Advanced > GOP Size	stream_assembly/video_description/*codec*/gop_size where codec is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > B Frames	stream_assembly/video_description/*codec*/ gop_num_b_frames where *codec* is one of the following: • **h264_settings**

Location of parameter on web interface	Location of tag in XML
	• **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > Closed GOP Cadence	stream_assembly/video_description/*codec*/ gop_closed_cadence where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > Reference Frames	stream_assembly/video_description/h265_settings/num_ref_frames
Streams – Video > Advanced > Scene Change Detect	stream_assembly/video_description/*codec*/ transition_detection where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > Min I-interval	stream_assembly/video_description/*codec*/ min_i_interval where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings**

Location of parameter on web interface	Location of tag in XML
Streams – Video > Advanced > GOP Reference B-Frame	stream_assembly/video_description/*codec*/ gop_b_reference where *codec* is one of the following: • **h264_settings** • **h265_settings**

Miscellaneous video tuning parameters

Description

This section describes individual encoding parameters that provide let you tune the video quality.

- **CABAC**: Applies only to H.264, and only when the **Profile** parameter is set to **Main** or **High**. This parameter enables arithmetic coding and can compress the same data with approximately 15% fewer bits. As a result, encoding is more efficient. There is no impact on video quality.

  Recommendation: Always enable this parameter unless the intended decoder or playback device doesn't support it.
- **Density vs Quality**: Applies only to H.264 and H.265. This parameter sets the balance between density on the appliance and video quality.
  - A value above **0** favors density over quality. This means that as you add more events to the appliance, the point will be reached where the encoder automatically starts to lower the video quality in any events that has this parameter set above 0.
  - A value below **0** favors quality over density.

  Note that this parameter controls density and quality, even though the parameter name in the XML is **svq** (speed vs quality).

  Recommendation: On newer appliance models (L8xx and later), we recommend that you leave the default. On older appliance models, a different value might change the balance. However, never use **-2** or **-3** with H.265.

- **Lookahead**: Lookahead indicates that the encoder should analyze a few frames into the future of the currently encoded frame, to allow the encoder to take future frame data into account during rate control logic. For example, if future frames are more complex, the encoder can allocate fewer bits to encode the current frame. In this way, the unused bits can be used to encode those future frames.

  The tradeoff of a higher lookahead is that processing and latency increase slightly, to allow the encoder to analyze those future frames.

  Recommendation: Set to **Medium** unless latency is critical.

- **Profile**: Applies only to H.264 and H.265. For H.264, the profile affects video quality. For H.265, the profile sets various characteristics of the video.

  Recommendation: With H.264, use **High Profile** to achieve higher video quality.

- **Slices**: Applies only to H.264 and H.265. This parameter improves speed of encoding. Using a higher number of slices can improve speed optimization. However, this results in slightly lower video quality.

  Recommendation:

  Set to **Auto** so that the encoder uses a value appropriate to the image height (resolution). The following table specifies which value the encoder will assign when you choose **Auto**, based on the image height.

Height (pixels)	Recommendation for H.264	Recommendation for H.265
Less than 720	1	1
Greater than or equal to 720	2	4
Greater than or equal to 1080	4	4
Greater than or equal to 2160	8	8

Location of parameters

This table shows where the parameters mentioned in this section are located. The first column shows the location on the web interface. The second column shows the location in the event XML.

Location of parameter on web interface	Location of tag in XML
Streams – Video > Advanced > Lookahead	stream_assembly/video_description/*codec*/look_ahead_rate_control where *codec* is one of the following: • `h264_settings` • `mpeg2_settings` • `h265_settings`
Streams – Video > Advanced > Profile	stream_assembly/video_description/h264_settings>/profile
Streams – Video > Advanced > Level	stream_assembly/video_description/h264_settings>/level
Streams – Video > Advanced > CABAC	stream_assembly/video_description/h264_settings>/cabac
Streams – Video > Advanced > Slices	stream_assembly/video_description/*codec*/slices where *codec* is one of the following: • `h264_settings` • `h265_settings`
Streams – Video > Advanced > Density vs Quality	stream_assembly/video_description/*codec*/svq where *codec* is one of the following: • `h264_settings`

Location of parameter on web interface	Location of tag in XML
	• **`h265_settings`**

Noise reduction

Description

You can apply the noise reducer preprocessor to improve output video quality.

- The **Filter** parameter has four options:
  - **Mean / Gaussian / Lanczos**: All of these algorithms allow for varying blur strengths. **Mean** is the strongest filter; it operates on a smaller group of pixels. **Lanczos** is the mildest filter; it operates on a larger group of pixels.
  - **Sharpen**: This algorithm sharpens the edges instead of softening them.
  - **Conserve**: This algorithm limits the pixel values to within the minimum and maximum values of the neighboring pixel values. It is designed to reduce speckle noise. This filter does not seem to be very valuable with video.
  - **Bilateral**: This algorithm tends to preserve strong edges; it's the best compromise between noise reduction and visual quality.
- **Strength**: The strength parameter of the filtering has its greatest effect at **3**.

Recommendations

In most cases, the noise reducer is not required. The reducer can help output quality if the content is being heavily compressed. However, it changes the visual quality of the video.

The recommended algorithm is bilateral, but the other algorithms might produce better results in certain use cases. We recommend that you run tests on a video sample, to determine the best option for your application.

Location of parameters

This table shows where the parameters mentioned in this section are located. The first column shows the location on the web interface. The second column shows the location in the event XML.

Location of parameter on web interface	Location of tag in XML
Streams > Advanced >Preprocessors > Noise Reducer > Filter	stream_assembly/video_description/video_preprocessors/ noise_reducer/filter
Streams > Advanced >Preprocessors > Noise Reducer > Strength	stream_assembly/video_description/video_preprocessors/ noise_reducer/strength

Quantization controls

Description

This guide shows how quantization can improve video quality. Quantization is a lossy data compression technique. The encoder includes several quantization controls.

- **Adaptive Quantization** (**AQ**):

  AQ can reduce *smear* in high-motion and sports content. It can also improve the video quality on areas that are smooth surfaces.

  Recommendation: We recommend that you set this parameter to **Auto**.

- **Spatial AQ**:

  Spatial AQ can improve video quality in video blocks where distortion would be noticeable. Spatial AQ allocates more bits to those high-risk areas, such as smooth textured blocks. It allocates slightly fewer bits to blocks where distortion won't cause noticeable visual degradation, such as complex textured blocks.

  Keep in mind that spatial AQ isn't content aware. The blocks that will receive fewer bits might belong to an object that is drawing the viewer's attention at that moment. Removing bits at that specific scene might not be ideal.

  This parameter applies only when **Adaptive Quantization** is set to **Low** or higher.

Recommendation: Choose a low spatial AQ for homogenous content, such as gaming and cartoons. Choose high or higher spatial AQ for content with a wider variety of textures.

- **Temporal AQ**:

  It applies fewer bits to areas within the frame that are complex in nature and that are affected by motion. For example, text tickers on newscasts and scoreboards on sports matches. The text characters are highly complex (sharp edges) and they scroll across the screen. Temporal AQ attempts to improve the readability by detecting motion of such complex entities, thus removing possible trailing artifacts.

  This parameter applies appears only when **Adaptive Quantization** is set to **Low** or higher.

  Recommendation: Temporal AQ should almost always be enabled. But keep in mind that temporal AQ might remove bits from some relevant areas of the frame. For example, if enabling temporal AQ on sports matches would make the athletes faces and movements less clear, it might be better to turn it off. On the other hand, for newscasts, you should definitely enable temporal AQ. The parameter helps improve video quality because it removes bits from the news presenters, who are moving less and therefore present less complexity. It adds bits to text tickers that present more comlexity.
- **Flicker AQ**: This parameter reduces flicker in the video. The parameter might prevent the abrupt change in detail that gives the effect of a flicker in the video.

  This parameter applies only to H.264 and H.265, and only when **Adaptive Quantization** is set to **Low** or higher.

  Recommendation: We recommend that you enable this parameter only when I-frame pulsing is noticeable. Otherwise, let the encoder run its macroblock optimizations freely.
- **Framing AQ**: Compresses the edges of the video slightly more than the center. The effect assigns more bits to the middle of the image, where the action and the viewer's attention are typically focused. When more bits are assigned, the video quality increases.

  This parameter appears only when the output codec is MPEG-2, and when **Adaptive Quantization** is set to **Low** or higher.

  Recommendation: For low bitrate encodes (for example, MPEG-2 1080i at 10 Mbps), we recommend that you enable this parameter. The visual effect of framing quantization is intentionally subtle.

- **Softness**: A higher softness value compresses high spatial frequencies more, which reduces bitrate allocation at the cost of image sharpness.

  This parameter appears only when the output codec is H.264 or MPEG-2.

  Recommendation: For low bitrate encodes (for example, MPEG-2 1080i at 10 Mbps), try values **24** and **32**. Then test whether the parameter makes any difference to your content.

Location of parameters

This table shows where the parameters mentioned in this section are located. The first column shows the location on the web interface. The second column shows the location in the event XML.

Location of parameter on web interface	Location of tag in XML
Streams – Video > Advanced > Adaptive Quantization	stream_assembly/video_description/*codec*/adaptive_quantization where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > Spatial AQ	stream_assembly/video_description/*codec*/spatial_aq where *codec* is one of the following: • **h264_settings** • **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > Temporal AQ	stream_assembly/video_description/*codec*/temporal_aq where *codec* is one of the following: • **h264_settings**

Location of parameter on web interface	Location of tag in XML
	• **mpeg2_settings** • **h265_settings**
Streams – Video > Advanced > Flicker AQ	stream_assembly/video_description/*codec*/flicker_aq where *codec* is one of the following: • **h264_settings** • **h265_settings**
Streams – Video > Advanced > Framing AQ	stream_assembly/video_description/mpeg2_settings/framing_aq
Streams – Video > Advanced > Softness	stream_assembly/video_description/*codec*/softness where *codec* is one of the following: • **h264_settings** • **mpeg2_settings**

Rate control mode

The rate control mode of the output video affects video quality. For more information, see [the section called "QVBR and rate control mode"](#).

Working with Audio

This chapter describes how to set up the audio features of AWS Elemental Live.

Topics

Converting to Dolby Digital Plus with Atmos

The following handling is supported:

- Encoding an audio output as Dolby Digital Plus with Atmos. The audio source must be a source that contains up to 16 channels.
- Passthrough of a source audio that is already Dolby Digital Plus with Atmos.

Elemental Live doesn't support transcoding (re-encoding) an audio source that is Dolby Digital Plus with Atmos.

Conversion to Dolby Digital Plus with Atmos requires the Advanced Audio Pack. For information about purchasing this package, see the section called "Purchasing a license". Note that passthrough of Dolby Digital Plus with Atmos doesn't require a license.

Supported sources

The source must have these characteristics:

- To convert to Dolby Digital Plus with Atmos, the audio input can be any source that has these characteristics:
  - Up to 16 channels in the following order:

    ```
    L R C LFE Ls Rs Lb Rb Tfl Tfr Tsl Tsr Tbl Tbr Lw Rw
    ```

  - If the source has fewer than 16 channels, Elemental Live extracts all the channels and then pads the output by inserting silence in the higher-numbered channels. For example, if the source has two channels, Elemental Live puts those channels in L and R, then inserts silence in the remaining channels.

- If the source doesn't have the channels in the specified order, the results might be wrong on the downstream player. For example, the sound of rain falling might come out of the left speaker instead of a ceiling speaker.
- A sampling rate of 48000 hz.

- To pass through Dolby Digital Plus with Atmos, the audio can be any coding mode and any sampling rate that Dolby Digital Plus supports.

For information about the types of inputs that support Dolby Digital Plus and Dolby Digital Plus with Atmos, see the section called “Reference: Supported live inputs”.

Supported outputs

Audio encoding

The Elemental Live implementation of Dolby Digital Plus with Atmos supports the following coding modes in the output:

- 5.1.4 coding mode
- 7.1.4 coding mode
- 9.1.6 coding mode

Within each coding mode, the speaker channels are arranged as shown in the following table.

Coding mode	Channel arrangement
5.1.4	`L R C LFE Ls Rs Tfl Tfr Tbl Tbr`
7.1.4	`L R C LFE Ls Rs Lb Rb Tfl Tfr Tbl Tbr`
9.1.6	`L R C LFE Ls Rs Lb Rb Tfl Tfr Tsl Tsr Tbl Tbr Lw Rw`

The abbreviations are the standard Dolby abbreviations: Left, Right, Center, LFE, Left surround, Right surround, Left rear surround, Right rear surround, Left back, Right back, Top front left, Top front right, Top side left, Top side right, Top back left, Top back right, Left wide, and Right wide.

Supported output groups

For information about the types of outputs that support Dolby Digital Plus with Atmos and Dolby Digital Plus with Atmos (as passthrough), see the section called “Reference: Supported outputs”.

Setting up the event

Follow this procedure to produce Dolby Digital Plus with Atmos in one or more outputs.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To set up the input

Follow this procedure if the source audio is Dolby Digital Plus, to convert the audio to Dolby Digital Plus with Atmos.

1. In the event in Elemental Live, go to the input that contains the Dolby Digital Plus audio that you want to transcode or pass through.
2. Choose **Advanced**, then choose **Add Audio Selector**.
3. Complete the fields to extract the Dolby Digital Plus audio.

To set up the output if the source audio is Dolby Digital Plus

1. In the event, go to the output group where you want to add the audio. Or create a new group.
2. Create the output where you want to add the audio encode.
3. In the **Streams** settings section for the output, choose the **Audio** section**.** Complete the fields as follows.

Field	Description
Audio Source	Choose the audio selector that you set up in the input.
Audio Codec	Choose Dolby Digital Plus with Atmos.
Coding mode	Choose the coding mode you want. For more information, see the section called "Supported outputs".
Bitrate	Choose a value that is applicable to the coding mode. For information, choose the **?** icon above the field.
Dialnorm	Complete these fields to specify values for the metadata. For information about a field, click the question mark icon above the field in Elemental Live.
DRC Line Mode Profile	
DRC RF Mode Profile	
Surround Trim	
Height Trim	

4. Complete the fields in the **Advanced** section as desired. Ignore **ARIB Dynamic Audio**, it doesn't apply.

To pass through Dolby Digital Plus with Atmos from the input to the output

Follow this procedure if the source audio is already Dolby Digital Plus with Atmos.

1. In the event, go to the output group where you want to add the audio. Or create a new group.
2. Create the output where you want to add the audio encode.
3. In the **Streams** settings section for the output, choose the **Audio** section.
4. Set these fields:
   - **Audio Source**: Set to the audio selector that you set up in the input.

- **Audio Codec**: Set to Dolby Digital Plus Passthrough.

With this setup, all audio sources in all the inputs will be passed through, both Dolby Digital Plus with Atmos and other audio.

> **Important**
>
> Don't set **Audio Codec** to **Dolby Digital Plus with Atmos**. That isn't the correct value for passing through. If you choose this option, the output might have silent audio.

Sample HLS manifest

If you include Dolby Digital Plus with Atmos in an HLS output group, the audio line in the HLS manifest looks like this example:

```
#EXTM3U
#EXT-X-VERSION:4
#EXT-X-INDEPENDENT-SEGMENTS
#EXT-X-STREAM-INF:BANDWIDTH=2208800,AVERAGE-
BANDWIDTH=2142800,CODECS="avc1.64001f,ec-3",RESOLUTION=1280x720,FRAME-
RATE=30.000,AUDIO="program_audio_0"
index_video.m3u8
#EXT-X-
MEDIA:TYPE=AUDIO,LANGUAGE="eng",NAME="English",AUTOSELECT=YES,DEFAULT=YES,CHANNELS="12/
JOC",GROUP-ID="program_audio_0",URI="index_audio.m3u8"
```

The `Channels` attribute in the last line is significant for Dolby Digital Plus with Atmos:

- 12/JOC indicates that the coding mode is 5.1.4 or 7.1.4 and the codec is Dolby Digital with Atmos.
- 16/JOC indicates that the coding mode is 9.1.6 and the codec is Dolby Digital with Atmos.

Working with Dolby metadata

Audio encoded with a Dolby codec always includes Dolby metadata, as per the ATSC A/52 2012 standard. This Dolby metadata is used by AWS Elemental Live in two ways when the stream is encoded with Dolby codec:

- It is used to manipulate the audio just before encoding the output.
- It is included in the metadata for the output stream.

This document describes how to set up an Elemental Live profile or event to use Dolby metadata in these ways.

Dolby metadata is supported in the output only when the audio codec for the output is Dolby Digital (also known as AC3) or Dolby Digital Plus (also known as Enhanced AC3).

Topics

Categories of metadata: Delivered and encoder control

There are two categories of parameters in the Dolby metadata, characterized by how Elemental Live uses it:

- Delivered: Elemental Live does not read these parameters, so they have no effect on the audio produced by Elemental Live. Instead, they are included as metadata in the output in order to **deliver** them to the downstream decoder.

  "Delivered" metadata is also called *Consumer* metadata because it is intended to be used by the end consumer's home decoder.
- Encoder Control: Elemental Live uses these parameters to manipulate the audio just before encoding the stream and producing the output. They provide a mechanism for Elemental Live to control the transcoding performed by Elemental Live. These parameters are never included in the output metadata.

“Encoder Control” metadata is also called *Professional* metadata because it is intended to be used by a professional device – in our case Elemental Live. It is never intended for the end consumer's home decoder.

Source of Elemental Live metadata

The metadata that Elemental Live emits can come from one of two sources:

- Metadata that is already in the source. Only audio sources that use a Dolby codec can include this metadata. Different Dolby codecs include different categories of metadata as shown in this table.

Codec	Categories present
Dolby Digital or Dolby Digital Plus	Delivered only
Dolby E	Delivered and Encoder Control

- Metadata that is specified by completing metadata fields in the profile or event. You can specify this metadata in any audio whose output codec is a Dolby codec. In other words, you can add it when the audio source is not a Dolby codec as long as the output audio uses a Dolby codec.

  Both categories of metadata can be specified when specifying this source.

You specify the source when setting up the profile or event.

Impact of the metadata on the output audio

Regardless of the source of the metadata, it affects the audio (either by manipulating encoder control or by being included in the output metadata) but only if the output codec is Dolby Digital or Dolby Digital Plus.

Combinations of input and output codec

The possible input and output codec combinations (in which at least one codec is a Dolby codec) are as follows. All these combinations support including metadata in the output.

Input codec	Output codec
Dolby Digital or Dolby Digital Plus	Dolby Digital or Dolby Digital Plus
Dolby Digital	Dolby Digital Passthrough (so Dolby Digital audio is passed through; it is not transcoded)
Dolby Digital Plus	Dolby Digital Passthrough (so Dolby Digital Plus audio is passed through; it is not transcoded)
Mix of Dolby Digital Plus and another codec	Dolby Digital Plus (with the Automatic Passthrough field checked)
Dolby E	Dolby Digital
Dolby E	Dolby Digital Plus
Dolby E	Dolby E (passthrough)
A non-Dolby codec	Dolby Digital or Dolby Digital Plus

The sample rate when encoding with a Dolby codec is always 48000.

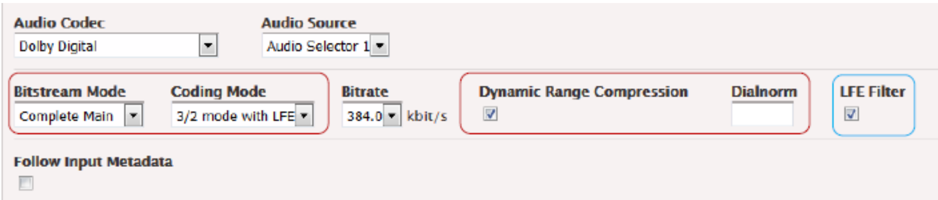
Setting up the profile or event using the web interface

This section describes how to set up the project or event using the web interface. To set up using the REST API, see the section called "Output with Dolby Digital" and the section called "Output with Dolby Digital Plus" to map the fields to their XML tags according to the following steps.

1. In the Output > Stream section, click the Audio tab to display the fields for audio.
2. Choose one of these Audio Sources as appropriate and complete the Audio Codec field: Dolby Digital, Dolby Digital Plus or Dolby Digital Pass Through. The fields for metadata appear.
3. Complete the remaining fields for the audio source that you selected. See the following to determine how to achieve the desired effect.

Dolby Digital codec

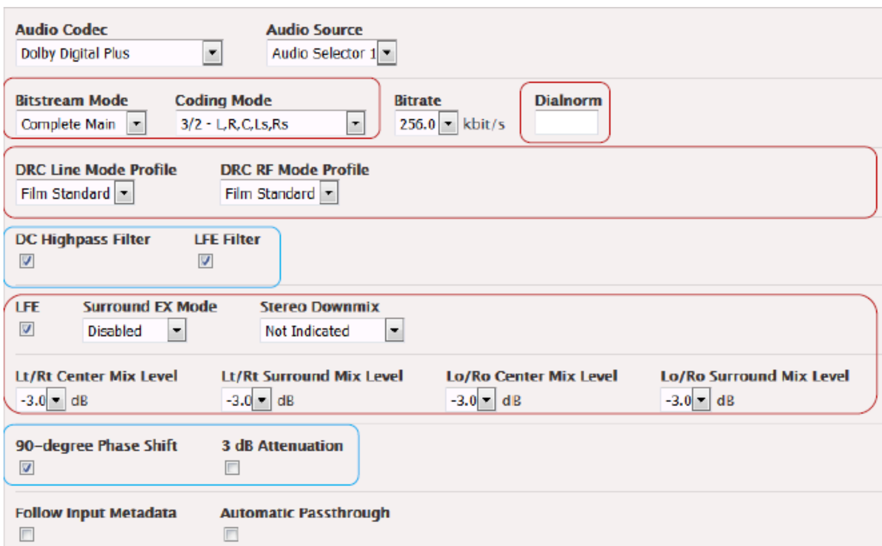
For Dolby Digital, encoder control fields are circled in blue and delivery fields are circled in red. Note that the LFE Filter field appears only when the Coding Mode is 3/2 mode.



Dolby Digital Plus codec

Encoder Control fields are circled in blue. Delivery fields are circled in red. Note that the Automatic Pass-through field does not relate to metadata.

Note that the Surround Mode field appears only when Coding Mode is 2/0.



Dolby Digital Passthrough

There are no fields for metadata.

Use the metadata in the audio source – case 1

Input codec	Output codec	Handling of audio
Dolby Digital or Dolby Digital Plus	Dolby Digital or Dolby Digital Plus	You are transcoding the audio.

- **Metadata Parameters** fields: Complete only Elemental Live Control fields, as required for your workflow.
- **Follow Input Metadata** field: Check this field after completing the Metadata Parameter fields.

Result for Metadata

Elemental Live Control parameters from the profile are applied during transcoding (given that the input does not include these parameters). If a given parameter is not exposed in the profile, a default value is always applied; see the section called "Output with Dolby Digital" and the section called "Output with Dolby Digital Plus".

The Delivery parameters from the input metadata are included in the output.

Use the metadata in the audio source – case 2

Input codec	Output codec	Handling of audio
Dolby Digital or Dolby Digital Plus	Dolby Digital or Dolby Digital Plus (Passthrough)	You are passing through the audio.

- **Metadata Parameters** fields: Not applicable.
- **Follow Input Metadata** field: No Encoder Control parameters are applied (because no transcoding occurs).

Result for Metadata

The Delivery parameters from the input metadata will be included in the output.

Use the metadata in the audio source – case 3

Input codec	Output codec	Handling of audio
Mix of Dolby Digital Plus and another codec	Dolby Digital Plus	You are passing through the Dolby Digital Plus audio and transcoding the remainder (Automatic Passthrough field is checked).

- **Metadata Parameters** field: Complete all the parameters.
- **Follow Input Metadata** field: Check this field after completing the metadata fields.

Result for Metadata

- Elemental Live Control parameters from the profile will be applied when transcoding the non-Dolby Digital Plus audio.
- No Encoder Control parameters will be applied when passing through the Dolby Digital Plus audio.
- The Delivery parameters from the profile will be used when transcoding the non-Dolby Digital Plus audio.
- The Delivery parameters from the audio source will be used for the passed-through Dolby Digital audio.

Use the metadata in the audio source – case 4

Input codec	Output codec	Handling of audio
Dolby E	Dolby Digital or Dolby Digital Plus	You are transcoding the audio.

- **Metadata Parameters** fields: Ignore.
- **Follow Input Metadata** field: Check.

Result for Metadata

- Elemental Live Control parameters from the input metadata are applied during transcoding.
- The Delivery parameters from the input metadata are included in the output.

Override the metadata with new values – case 5

Input codec	Output codec	Desired effect
Any codec	Dolby Digital or Dolby Digital Plus	To override the metadata in the audio source.

- **Metadata Parameters** field: Complete as desired.
- **Follow Input Metadata** field: Leave unchecked.

Result for metadata

The values from the profile are used.

- With all parameters except Dialnorm, the values from the profile are used. If a given parameter is not exposed in the profile, a default value is always applied; see the section called "Output with Dolby Digital" and the section called "Output with Dolby Digital Plus".
- With Dialnorm, the value from the profile is used. If the profile has no value and the source is a Dolby file, the value from the input metadata is used. If the profile has no value and the source is *not* a Dolby file, a default value is used; see the section called "Output with Dolby Digital" and the section called "Output with Dolby Digital Plus".

Elemental Live Control parameters are applied during transcoding. The Delivery parameters are included in the output.

Output with the Dolby Digital codec

N m p s	C	Fi	A ta (a st se \a d ip \a 3, s)	D
D Lo	D	D	d	Not set norm
C M	D	C M	co d	2/0 ng_mo
LF C	D	C M C M se to "3 w LF m LF is er A o o		Disabled ode d. s

N m p s	C	Fi	A ta (a st se \a d ip \a 3, s)	D
		m LI is d		d.
B M	D	B M	b _	Complete Main
Li M C o	D	N u cc		Film Std.
R M C o	D	N u cc		Film Std.
R O a P n	D	N u cc		

N m p s	C	Fi	A ta (a st se \a d ip \a 3, s)	D
C D L	D	N u co		-3dB
S D L	D	N u co		Not indicated
D S M	D	N u co		Disabled
A P n Ir o	D	N u co		0 (does not exist)
M L	D	N u co		Not set
R T	D	N u co		Not set

N m p s	C	Fi	A ta (a st se \a d ip \a 3, s)	D
C B	D	N u: co		0
O B	D	N u: co		0
P S D	D	N u: co		Not indicated
Lt Rt C D Lo	D	N u: co		-3.0 dB
Lt Rt S D Lo	D	N u: co		-3.0 dB

N m p s	C	Fi	A ta (a st se \a d ip \a 3, s)	D
L R C D L	D	N u: cc		-3.0 dB
L R S D L	D	N u: cc		-3.0 dB
D S E M	D	N u: cc		Disabled
A D C Ty	D	N u: cc		0 (standard)
D Fi	E	N u: cc		Enabled

N m p s	C	Fi	A ta (a st se \a d ip \a 3, s)	D
Ll Lo Fi	E	W C M is 3, Ll Fi cl ap at th fa ri	lf r	Disabled ox s
S 3 d A o	E	N us co		Enabled
S P S	E	N us co		Disabled

Output with Dolby Digital Plus (EC2, EAC3) codec

N m p s	C	Fi	A ta (s ss \ a cr \ a n	D
D L	D	D	d	Not set norm
C M	D	C M	cc d	3/2 – L,R,C,LS, Rs ng_mo
LI C	D	W C M is 7 o 3, tl LI cl a tl li b tl cl	lf	Enabled ox s ox.

N m p s	C	Fi	A ta (s ss \ a cr \ a n	D
B M	D	B M	b _	Complete Main
Li M C o	D	D Li M P	d	Film Std.
R M C o	D	D R M P	d	Film Std.
R O a P n	D	N u: co	-	
C D L	D	N u: co	-	-3dB
S D L	D	S D	st w	Not indicated

N m p s	C	Fi	A ta (s ss \ a cr \ a n	D
D S M	D	N u co	-	Disabled
A P n Ir o	D	N u co	-	0 (does not exist)
M L	D	N u co	-	Not set
R Ty	D	N u co	-	Note set
C B	D	N u co	-	0
O B	D	N u co	-	0

N m p s	C	Fi	A ta (s ss \ au cr \ ad n	D
P S D	D	N us co	-	Not indicated
Lt R C D Lo	D	Lt R C M Lo	lt te ev	-3.0 dB
Lt R S D Lo	D	Lt R S M Lo	lt ro _l	-3.0 dB
Lo R C D Lo	D	Lo R C M Lo	lo te ev	-3.0 dB

N m p s	C	Fi	A ta (s ss \ au cr \ ac n	D
Lo R S D Lo	D	Lo R S M Lo	lo ro _l	-3.0 dB
D S E: M	D	S E: M	su ex	Disabled
A D C Ty	D	N us co		0 (standard)
D Fi	E	C H Fi	d	Enabled
LI Lo Fi	E	LI Fi	lf r	Enabled

N m p s	C	Fi	A ta (s ss \ au cr \ ac n	D
S 3 d A o	E	3 d A o	at _3	Disabled / nable db ati
S P S	E	9 d P S	p ft re	Enabled / se_shi 0_deg
S M	D	W C M is 2, th S M cl ap	su m	Not indicated / bund_ led nd ox s.

Setting up HLS rendition groups

In AWS Elemental Live, you can set up an HLS output group to support an audio rendition group.

In setting up an HLS output group to support an audio rendition group, each HLS output you create contains a “set” consisting of one video stream and several audio streams. All the audio streams in the set are associated with that one video stream. The HLS output group can contain more than one of these outputs. For example, one set consisting of high bitrate video and audio in four languages and another set consisting of low bitrate video and audio in the same four languages.

With this setup, the manifest that is created provides options for video. The logic of the manifest allows the player to select one of those video options and then to select audio that is valid for that video option.

For example:

1. The client player reads the manifest and selects the desired video, such as a high bitrate video.
2. The client player then selects an audio group from among the groups associated with that video, such as the Dolby Digital group instead of the AAC group.
3. The client player then selects an audio from that group, such as Spanish.

   Typically, the player makes its audio selection based on rules on the player side, such as selecting the language that corresponds to the operating system language, or based on rules defined in the manifest, such as when the manifest identifies one audio as the default.

Standards compliance

This implementation of audio rendition groups is compliant with the “HTTP Live Streaming draft-pantos-http-live-streaming-18” section 4.3.4.1.1.

Note that Elemental Live does not support rendition groups for video. They do support rendition groups for captions since Elemental Live automatically creates one captions rendition group to hold all caption stream assemblies in a given output.

Topics

How video is associated with audio rendition groups

The different “sets” of media are created as follows:

- Create one “video-only” stream assembly containing only one video stream.
- Create two or more “audio-only” stream assemblies, each containing only one audio stream.
- Assign “audio group IDs” to the audio-only streams. To group several audio streams into one rendition group, assign the same audio group ID to the relevant audio streams. To group other audio streams into another rendition group, define a different audio group ID and assign it to the relevant audio streams.
- Associate each video-only stream with its corresponding audio rendition group by assigning the desired audio group ID to that stream.

For example:

- To group stream 3, 4 and 5 to one audio rendition group, set the audio group ID for each of these streams to “audio 1” or some other name of your choosing.
- To group streams 6, 7 and 8 to another audio rendition group, set the ID for each of these streams to “audio 2” or some other name.
- To associate video 1 with the first rendition group, set the “audio rendition sets ID” of that video to “audio 1”.
- To associate video 2 with the other group, set the audio rendition sets ID to “audio 2.”

Rules for rendition groups

Rules exist for associating both audio and video streams in their respective rendition groups. These are described following.

- A given audio stream can belong to only one audio rendition group.
- Any given video stream can be associated with more than one rendition group. (For example, “video high” can be associated with both “Dolby audio streams” and “AAC audio streams.” You do not need to create two video streams.)

  With this setup, all the rendition groups associated with the same video stream must contain the same audio streams.(For example, “Dolby audio streams” and “AAC audio streams” must contain the same audio streams (perhaps English, French and Spanish)).

- Any given audio rendition group can be associated with more than one video stream.(For example, “Dolby audio streams” rendition group can be associated with “video high” and “video low.” You do not need to create two rendition groups, one for each video.)
- Any video stream can be associated with more than one output group. (For example, “video high” can appear in two different HLS output groups).

You can use a combination of these rules. For more information, see Examples of HLS rendition groups.

Examples of HLS rendition groups

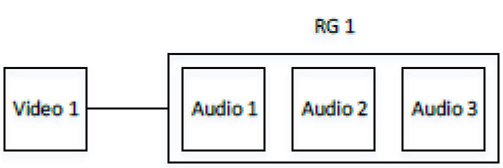
Example 1

The outputs in an HLS output group consist of:

- One video stream.

This video is associated with an audio rendition group that contains:

- One English stream.
- One French stream.
- One Spanish stream.



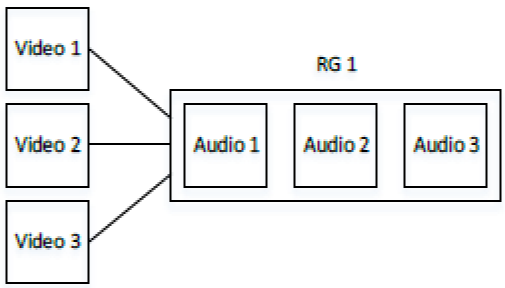
Example 2

The outputs in an HLS output group consist of:

- One “video high” stream.
- One “video medium” stream.
- One “video low” stream.

Each of these videos is associated with the same audio rendition group that contains:

- One English stream.
- One French stream.
- One Spanish stream.



Example 3

The outputs in an HLS output group consist of:

- One video stream.

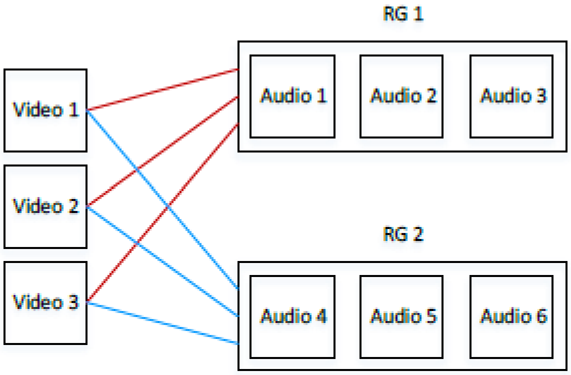
This video is associated with two audio rendition groups.

The first audio rendition group contains:

- One English stream in AAC codec.
- One French stream in AAC codec.
- One Spanish stream in AAC codec.

The second audio rendition group contains:

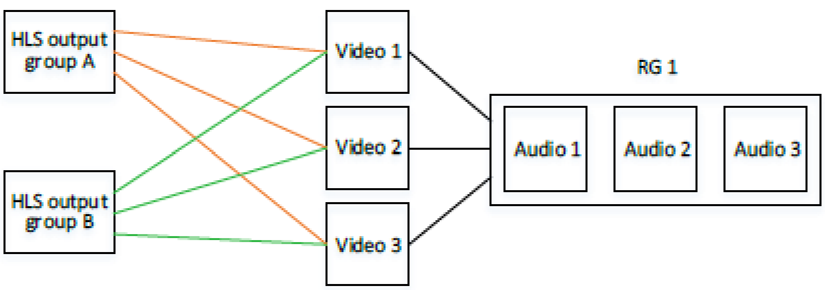
- One English stream in Dolby Digital.
- One French stream in Dolby Digital.
- One Spanish stream in Dolby Digital.



Example 4

There are two output groups, one that pushes to a WebDAV server and the other that delivers to an Akamai server.

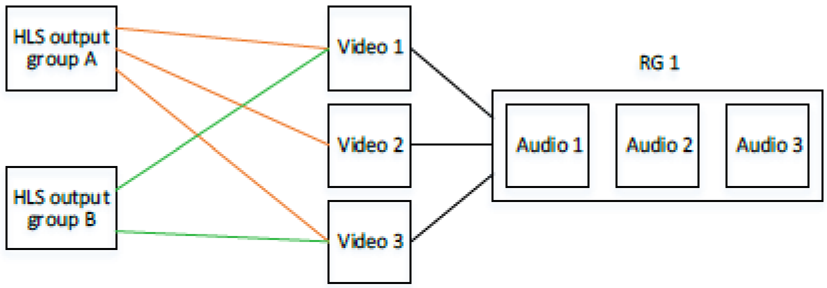
Each output group is identical in terms of its video and rendition groups. For example, each output group produces the video and rendition group from Example 2. You do not need to encode the streams twice; do it only once for each output group. So long as the two output groups are in the same event, each can be associated with the same streams.



Example 5

There are two output groups, one that pushes to a WebDAV server and the other that delivers to an Akamai server.

Each output group is similar in terms of its video and rendition groups. For example, the first output group produces the video and rendition group from Example 2. The second output group produces the only “video high” and “video low” but it is associated with the same audio rendition group as the first output group.



Creating HLS rendition groups

The key to creating rendition groups is that each output you create must contain only one stream. Therefore, for each video to include in the output group, you create a stream assembly that contains only one video (no audio or captions). For each audio to include in a rendition group, you create a stream assembly that contains only one audio (no video or captions).

This means that when rendition groups are present in the HLS output group, an output is identical to a stream. (Usually an output contains a mix of several streams and several stream types.)

Topics

- Getting ready to create HLS rendition groups
- Creating HLS rendition groups (web interface)
- Creating HLS rendition groups (REST API)

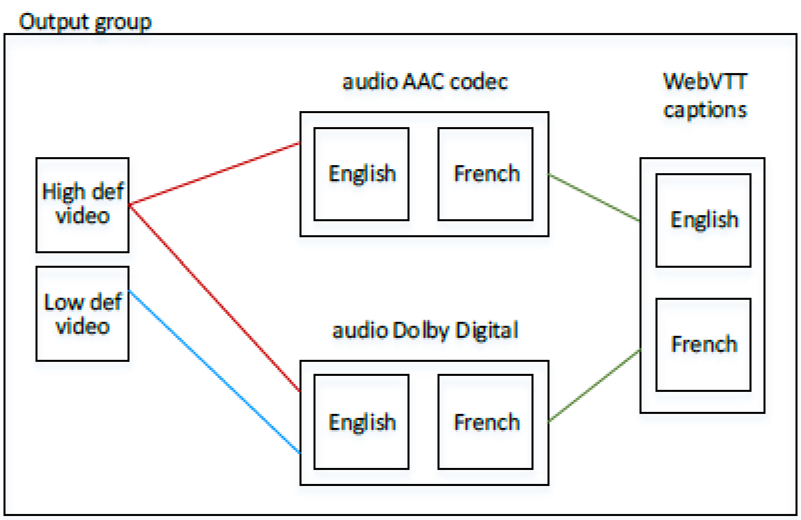
Getting ready to create HLS rendition groups

Step 1. Create a mapping

Identify the video, audio, audio rendition groups and captions you require. Review the the section called "Rules for rendition groups" to ensure you design an output that is valid. For example:

- Video "high definition."
- Video "low definition."
- A rendition group named "AAC group" for AAC audio.
- A rendition group named "Dolby group" for Dolby Digital audio.
- Audio English AAC in "AAC group" rendition group.
- Audio English Dolby Digital in "Dolby group" rendition group.

- Audio French AAC in “AAC group” rendition group.
- Audio French Dolby Digital in “Dolby group” rendition group.
- Video “high definition” to be associated with both rendition groups.
- Video “low definition” to be associated with the “Dolby group” rendition group.
- Captions in English and French in WebVTT format, to be associated with both rendition groups.



Step 2. Determine defaults and auto-selection behavior

For each audio rendition group, decide which audio will be the default and how auto-selection works for the non-defaults. Setting up this information is useful if:

- The user has specified an audio preference on the client player but that preference is not available, or
- If the user has not specified an audio preference.

(Obviously, if the user has specified a preference and that preference is available, the client player will select that preference.)

Determine defaults and auto-selection behavior

Set the Audio Track Type field. The options for this field for each audio stream follow.

Value	Client player behavior	Representation in HLS manifest
Alternate Audio, Auto Select, Default	The client player should *select* this stream. Only one stream in the rendition group should be set as the default; otherwise, the client player may behave unexpectedly.	EXT-X-MEDIA with DEFAULT=YES, AUTOSELEC T=YES
Alternate Audio, Auto Select, Not Default	The client player *may* select this stream. Any number of streams in the rendition group can be set this way.	EXT-X-MEDIA with DEFAULT=NO, AUTOSELEC T=YES
Alternate Audio, not Auto Select	The client player *should never* select this stream. Any number of streams in the rendition group can be set this way.	EXT-X-MEDIA with DEFAULT=NO, AUTOSELEC T=NO
Audio-Only Variant Stream	The client can play back this audio-only stream instead of video in low-bandwidth scenarios.	EXT-X-STREAM-INF

1. Set the **Alternate Audio Track Selection** as follows:

Desired result	How to set
There is a default. The player can auto-sele ct any of the other audios.	• Set only one audio stream to “Alternate Audio, Auto-Select, Not Default, Default.” • Set every other audio stream to “Alternate Audio, Auto-Select, Not Default.”

Desired result	How to set
There is a default. The player cannot auto-select any of the other audios.	• Set only one audio stream to "Alternate Audio, Auto-Select, Not Default, Default." • Set every other audio stream to "Alternate Audio, not Auto-Select."
There is a default. There are specific audios that the player can auto-select.	• Set only one audio stream to "Alternate Audio, Auto-Select, Not Default, Default." • Set some of the other audio streams to "Alternate Audio, Auto-Select, Not Default." • Set some of the other audio streams to "Alternate Audio, not Auto Select."
There is no default. The player can auto-sele ct any audio it chooses.	• Set every audio stream to "Alternate Audio, Auto-Select, Not Default."
There is no default. The player cannot auto-select any audio.	• Set every audio stream to "Alternate Audio, not Auto-Select."
There is no default. There are specific audios that the player can auto-select.	• Set some audio streams to "Alternate Audio, Auto-Select, Not Default." • Set some audio streams to "Alternate Audio, not Auto-Select."

2. In addition, if you have an audio that is intended as the audio to play when the bandwidth is so low that the video cannot be delivered, then set that audio to "**Audio-Only Variant Stream**."

Creating HLS rendition groups (web interface)

Topics

- Step 1. Create video-only outputs
- Step 3. Create audio-only outputs
- Step 4. Caption-only streams
- Step 5. Verify Outputs for the HLS Rendition Group

- Summary of the steps to create an HLS rendition group
- Example of the Event Output: Creating Caption-Only Streams for an HLS Rendition Group

Step 1. Create video-only outputs

You must create “video-only” outputs. Follow these steps for each video-only output you need:

1. In the Elemental Live web interface, display the **Apple HLS** output group tab.
2. Click **Add Output** in order to create an output.
3. Select or create a stream to be associated with that output. For example, Stream 1.
4. In that stream, delete the **Default Audio** tab. This stream now contains only a video stream.
5. Go back to the Output section associated with this stream and click **Advanced**. The **Audio Rendition Sets** field now appears, with a default value of “audio_program.” This field shows in an output only when the associated stream contains only one video stream.
6. Change the **Audio Rendition Sets** field to specify the rendition group or groups to associate with this video:
   - To associate the video with one rendition group, enter the name of the rendition group.
   - To associate the video with several rendition groups, enter a comma-separated list of the rendition group names. Do not put a space after each comma.

Step 3. Create audio-only outputs

Follow these steps for each audio-only output you need:

1. Click **Add Output** in order to create an output.
2. Select or create a stream to be associated with that output. For example, Stream 3.
3. In that stream, delete the default **Video** and **Captions** tabs.This stream is now an audio stream.
4. Complete the following fields in the **Advanced** section:
   - **Stream Name**: The wording for the NAME parameter in the manifest, as described in Audio information for an HLS output group with audio rendition group event . This is the audio description that the client player user interface displays. If the description is a language, it should be in that language. For example, “Deutsch ”, not “German”.
   - **Language Code**: Optional; complete only if the audio is a language. The wording that is to appear in the LANGUAGE parameter in the manifest should be the language code as per RFC

5646, as described in Audio information for an HLS output group with audio rendition group event . This is the language code that the client player reads.

You can also leave this field blank and check **Follow Input Language Code**; the language of the audio (specified in **Audio Source**, a bit higher up on the screen) is detected.

5. Go back to the Output section associated with the first audio stream and click **Advanced**.

   The **Audio Group ID** field and **Audio Track Type** now show. These fields show in an output only when the associated stream contains only one audio stream. The **Audio Group ID** field shows the default value “audio_program.”

   (Note that the **Audio Only Image** field also appears in an audio-only stream. This field has nothing to do with audio rendition groups; it is used to assign an image in a “regular stream that has no video.”)

   - Set the **Audio Group ID** field to specify the audio rendition group that this audio will belong to. For example, “AAC group” or “Dolby group.”
   - Set up the **Alternate Audio Track Selection** field (**Audio Track Type** field), as described in Step 2. Determine defaults and auto-selection behavior.

Step 4. Caption-only streams

If your output includes captions, you must create “captions-only” outputs. Follow these steps for each captions-only output you need:

1. Click **Add Output** in order to create an output.
2. Select or create a stream to be associated with that output.For example, Stream 4.
3. In that stream, delete the default **Video** tab and the default **Audio** tab. This stream now contains only a **captions** stream.

For more information about setting up captions, see Working with captions.

Step 5. Verify Outputs for the HLS Rendition Group

- Finally, check all your outputs for the HLS output groups and make sure you do not have an output that contains both audio and video. Including such an output may produce a manifest that the client player cannot interpret.

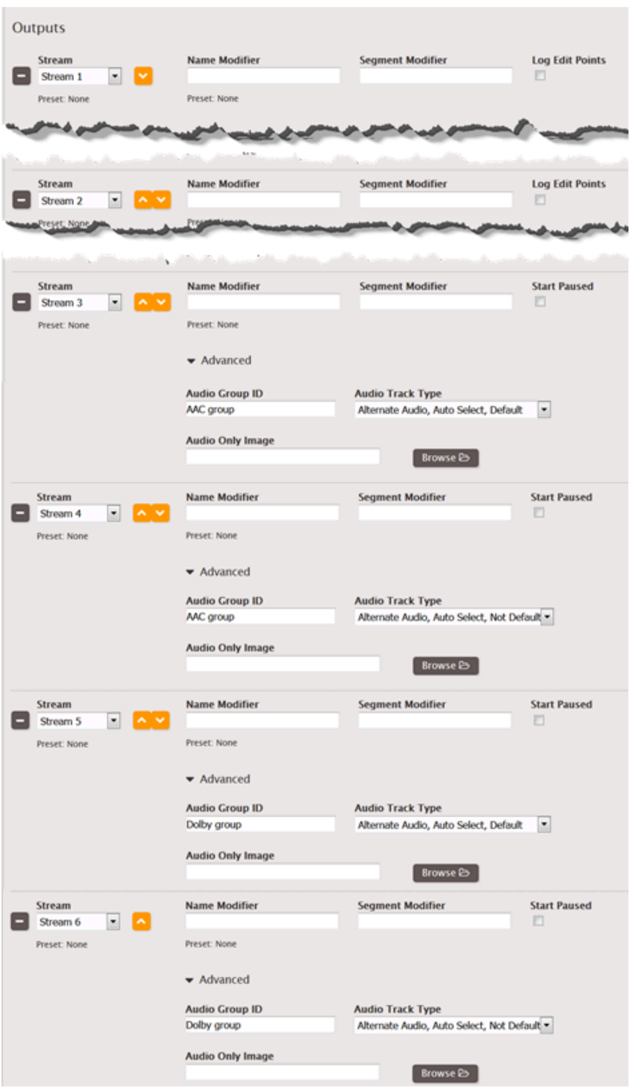
Summary of the steps to create an HLS rendition group

After these steps, you have:

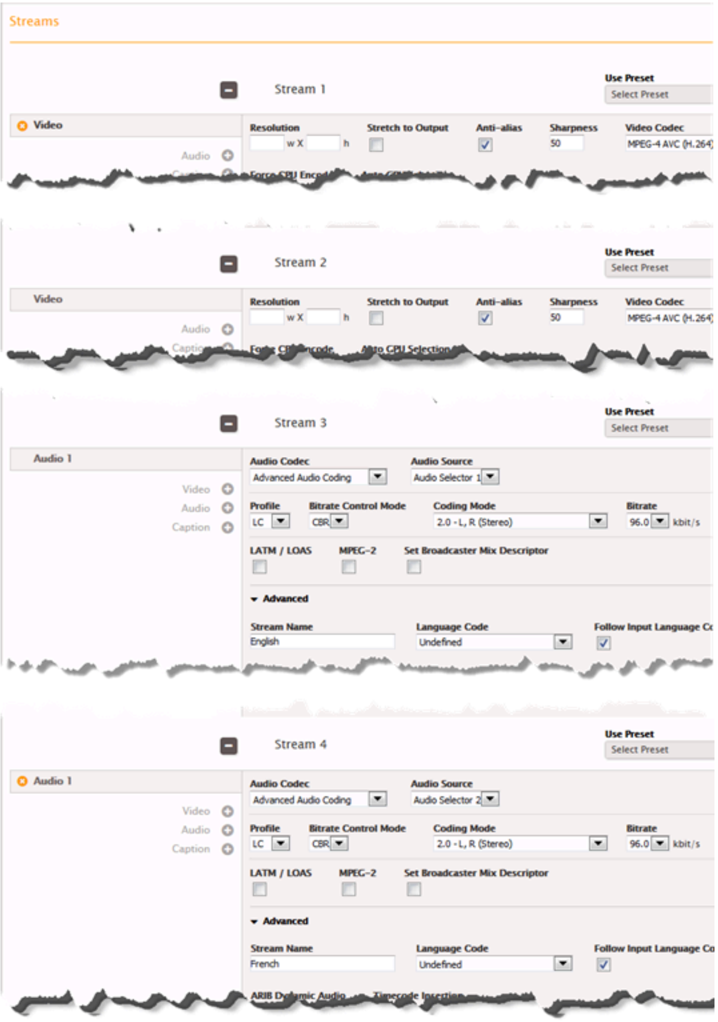
- One or more video-only outputs. Each output is associated through its **Audio Rendition Sets** field to one or more audio rendition groups.
- Two or more audio-only outputs. Each output belongs to an audio rendition group based on the value in the **Audio Group ID** field.
- Optionally, one or more captions-only outputs.

Example of the Event Output: Creating Caption-Only Streams for an HLS Rendition Group

Here is the Output section.



Here is the Streams section.



Creating HLS rendition groups (REST API)

The following information assumes that you have read the section called "Creating HLS rendition groups (web interface)" and are therefore familiar with the construction and association of an output containing video and rendition groups.

Via the REST API, create or modify the event to include the elements and tags in the XML body as described in the following sections.

Topics

- Creating streams for HLS rendition groups using the REST API
- Creating output groups for HLS rendition groups using the REST API
- Creating outputs for HLS output groups using the REST API

- Sample XML body for an HLS output group with audio rendition group event

Creating streams for HLS rendition groups using the REST API

- Create as many stream_assembly elements as you require, one for each unique video stream, one for each unique audio stream, and one for each caption stream.
- Each stream_assembly element must contain only of these:
  - One video_description element (plus an optional preset_id tag and name tag), or
  - One audio_description element (plus an optional preset_id tag and name tag), or
  - One caption_description element (plus an optional preset_id tag and name tag).

Creating output groups for HLS rendition groups using the REST API

- Create as many HLS output groups as desired by creating one output group that has the value "apple_live_group_settings" in its type tag and that contains one apple_live_group_settings element. Set other tags as desired.

Creating outputs for HLS output groups using the REST API

- Within each HLS output group, create as many output elements as required, one for each video stream (plus captions), one for each audio stream, and one for each captions stream.
- Each **video** output element must contain:
  - container: m3u8
  - extension: m3u8
  - stream_assembly_name: The name of the one stream_assembly to associate with this output.This value matches the value of the name tag in the corresponding stream_assembly_name element.
  - apple_live_settings element that contains:
    - audio_rendition_sets tag: A comma-separated list of the names of the audio rendition groups to associate with this video output to create a set. This value matches the value of the audio_group_id tag in each of the associated audio outputs. For example, "audio_1" in the audio_rendition_sets of this video output matches the "audio_1" in the audio_group_id tag of the associated audio output.
    - Other tags as you require.

- Each **audio** output element must contain:
  - container: m3u8
  - extension: m3u8
  - stream_assembly_name: The name of the one stream_assembly to associate with this output.This value matches the value of the name tag in the corresponding stream_assembly_name element.
  - apple_live_settings element that contains:
    - audio_group_id: The name of the audio rendition group this audio output belongs to. Specifying a value here creates the rendition group and puts this audio output into that rendition group.
    - audio_track_type: Either “alternate_audio_auto_select_default” or “alternate_audio_auto_select” or “alternate_audio_not_auto_select” or “audio_only_variant_stream”. See the section called “Step 2. Determine defaults and auto-selection behavior”for information.
    - Other tags as you require.
- Each **captions** output element must contain:
  - container: m3u8
  - extension: m3u8
  - stream_assembly_name: The name of the one stream_assembly to associate with this output.This value matches the value of the name tag in the corresponding stream_assembly_name element.
  - apple_live_settings element that contains the usual tags as required.

Sample XML body for an HLS output group with audio rendition group event

This example shows the XML body for an event that contains an HLS output group that includes audio rendition groups.

Following is the `<input>` element. There are no special rendition group requirements that affect this element.

```
<live_event>
<name>Multi Audio - one video</name>
<input>
.
.
```

```
.
</input>
.
.
.
```

Following is the `<stream_assembly>` element for one video. This stream_assembly has the name tag set to “stream_assembly_0” (assigned by default).

```
<stream_assembly>
    <name>stream_assembly_0</name>
    .
    .
    .
    <video_description>
    .
    .
    .
    <h264_settings>
    .
    .
    .
    </h264_settings>
    <codec>h.264</codec>
    .
    .
    .
    </video_description>
    <caption_description>
      <language_code>eng</language_code>
      <language_description>English</language_description>
    .
    .
    .
    </caption_description>
</stream_assembly>
```

Following is the `<stream_assembly>` for the first audio. This stream_assembly has the name tag set to “stream_assembly_1” (assigned by default).

```
<stream_assembly>
    <name>stream_assembly_1</name>
```

```
    <audio_description>
    <follow_input_language_code>false</follow_input_language_code>
    <language_code>eng</language_code>
    <stream_name>English</stream_name>
    .
    .
    .
    <aac_settings>
    .
    .
    .
    .
    </aac_settings>
    .
    .
    .
    <codec>aac</codec>
    <audio_source_name>Audio Selector 1</audio_source_name>
    </audio_description>
</stream_assembly>
```

Following are the `<stream_assembly>` elements for three more audios: stream_assembly_2, stream_assembly_3, and stream_assembly_4.

```
<stream_assembly>
    <name>stream_assembly_2</name>
    <audio_description>
    <follow_input_language_code>false</follow_input_language_code>
    <language_code>eng</language_code>
    <stream_name>English</stream_name>
    .
    .
    .
    <aac_settings>
    .
    .
    .
    </aac_settings>
    .
    .
    .
    <codec>aac</codec>
```

```
        <audio_source_name>Audio Selector 1</audio_source_name>
        </audio_description>
</stream_assembly>

<stream_assembly>
        <name>stream_assembly_3</name>
        <audio_description>
        .
        .
        .
        </audio_description>
</stream_assembly>

<stream_assembly>
        <name>stream_assembly_4</name>
        <audio_description>
        .
        .
        .
        </audio_description>
</stream_assembly>
```

Following is the `<stream_assembly>` for the first caption. This stream_assembly has the name tag set to “stream_assembly_5” (assigned by default).

```
<stream_assembly>
        <name>stream_assembly_5</name>
        <caption_description>
            <destination_type>WebVTT</destination_type>
            <language_code>eng</language_code>
            <language_description>English</language_description>
            <order>1</order>
            <caption_source_name>Caption Selector 1</caption_source_name>
        </caption_description>
</stream_assembly>
```

Following are the `<stream_assembly>` elements for three more captions: stream_assembly_6, stream_assembly_7, and stream_assembly_8.

```
<stream_assembly>
        <name>stream_assembly_6</name>
        <caption_description>
```

```
    <destination_type>WebVTT</destination_type>
    <language_code>eng</language_code>
    <language_description>English</language_description>
    <order>1</order>
    <caption_source_name>Caption Selector 1</caption_source_name>
    </caption_description>
</stream_assembly>
<stream_assembly>
    <name>stream_assembly_7</name>
    .
    .
    .
</stream_assembly>
<stream_assembly>
    <name>stream_assembly_8</name>
    .
    .
    .
</stream_assembly>
```

Following is the `<output_group>` of type apple_live_group_settings.

```
<output_group>
<apple_live_group_settings>
.
.
.
</apple_live_group_settings>
<type>apple_live_group_settings</type>
```

Following is the `<output>` (nested in the HLS output_group element) that is associated with stream_assembly_0 and is therefore a video output. This video is associated with the rendition groups “Audio_aac_hi” and “Audio_aac_lo.”

```
    <output>
        <extension>m3u8</extension>
        .
        .
        .
        <apple_live_settings>
        <audio_rendition_sets>Audio_aac_hi,Audio_aac_lo</audio_rendition_sets>
```

```
        .
        .
        .
        </apple_live_settings>
        <m3u8_settings>
        .
        .
        .
        </m3u8_settings>
        <stream_assembly_name>stream_assembly_0</stream_assembly_name>
        <container>m3u8</container>
    </output>
```

Following is the `<output>` (nested in the HLS output_group element) that is associated with stream_assembly_1 and is therefore an audio output. This audio is part of the rendition group "Audio_aac_hi."

```
    <output>
        <extension>m3u8</extension>
        .
        .
        .
        <apple_live_settings>
        <alternate_audio_track_selection>default_audio</
alternate_audio_track_selection>
        <audio_group_id>Audio_aac_hi</audio_group_id>
        .
        .
        .
        </apple_live_settings>
        <m3u8_settings>
        .
        .
        .
        </m3u8_settings>
        <stream_assembly_name>stream_assembly_1</stream_assembly_name>
        <container>m3u8</container>
    </output>
```

More outputs follow, one for each audio stream assembly. Each is part of a rendition group.

```
    <output>
```

```
.
.
.
</output>
<output>
.
.
.
</output>
<output>
.
.
.
</output>
```

Following is the <output> (nested in the HLS output_group element) that is associated with stream_assembly_5 and is therefore a caption output.

```
<output>
    <extension>m3u8</extension>
    .
    .
    .
    .
    <stream_assembly_name>stream_assembly_5</stream_assembly_name>
    <container>m3u8</container>
</output>
```

More outputs follow, one for each caption stream assembly and one for each caption assembly.

```
<output>
      .
      .
      .
</output>
<output>
      .
      .
      .
</output>
<output>
      .
      .
```

```
          .
    </output>
          .
          .
          .
</output_group>
</live_event>
```

Sample HLS output group with audio rendition group event manifest

Video information for an HLS output group with audio rendition group event

- There are two video streams, as indicated by the presence of two EXT-STREAM-INF lines.
  - The first video stream has a low bandwidth. As indicated by the AUDIO parameter, it is associated with “audio1.”
  - The second video stream has a higher bandwidth. As indicated by the AUDIO parameter, it is associated with “audio2.”

Audio information for an HLS output group with audio rendition group event

- There are four audio streams as indicated by the presence of four EXT-X-MEDIA lines with TYPE=AUDIO.
- There are two audio rendition groups, as indicated by the values for the GROUP-ID in each line. The first two lines belong to audio1, the second two to audio2.
- In each audio stream, the values for the various parameters come from these fields in the web interface:
  - TYPE: Always Audio.
  - GROUP-ID: from Audio Group ID field in Output > Advanced.
  - LANGUAGE: from the Language Code field in Stream > Advanced.
  - NAME: from the Stream Name field in Stream > Advanced.
  - AUTOSELECT: from the Audio Track Type in Output > Advanced.
  - DEFAULT: from the Alternate Audio Track Selection field in Output > Advanced.
  - URI: from the combined Destination field (in Output Group) and Name Modifier field (in Stream).

Captions information for an HLS output group with audio rendition group event

- There are two captions streams, as indicated by the presence of two EXT-X-MEDIA lines with TYPE=SUBTITLES.

```
#EXTM3U
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="audio1",LANGUAGE="eng",NAME="English",AUTOSELECT=YES,
 \
DEFAULT=YES,URI="eng1/prog_index.m3u8"
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-
ID="audio1",LANGUAGE="fre",NAME="français",AUTOSELECT=YES, \
DEFAULT=NO,URI="fr1/prog_index.m3u8"
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="audio2",LANGUAGE="eng",NAME="English",AUTOSELECT=YES,
 \
DEFAULT=YES,URI="eng2/prog_index.m3u8"
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="audio2",LANGUAGE="fr",NAME="français",AUTOSELECT=YES,
 \
DEFAULT=NO,URI="fr2/prog_index.m3u8"

#EXT-X-MEDIA:TYPE=SUBTITLES,GROUP-ID="subs",LANGUAGE="eng",NAME="English",
DEFAULT=YES,AUTOSELECT=YES,FORCED=NO,URI="1c1.m3u8"
#EXT-X-MEDIA:TYPE=SUBTITLES,GROUP-ID="subs",LANGUAGE="fra",NAME="French",
DEFAULT=YES,AUTOSELECT=YES,FORCED=NO,URI="1c2.m3u8"

#EXT-X-STREAM-INF:PROGRAM-
ID=1,BANDWIDTH=195023,CODECS="avc1.42e00a,mp4a.40.2",AUDIO="audio1"
lo/prog_index.m3u8,SUBTITLES="subs",URI="1c2.m3u8"
#EXT-X-STREAM-INF:PROGRAM-
ID=1,BANDWIDTH=591680,CODECS="avc1.42e01e,mp4a.40.2",AUDIO="audio2"
hi/prog_index.m3u8,URI="1c2.m3u8"
```

Elemental Live Features

This chapter describes the features you can implement in AWS Elemental Live.

Topics

Working with captions

You can set up Elemental Live to extract captions when it ingests the source and to include those captions in the output in either the same or a different format. You can include several captions in the output. For example, you can include captions for several languages. You can take a source captions asset and convert it to one format in one output and to another format in a different output.

You perform the setup for captions in your Elemental Live event. The information in this section assumes that you are familiar with the general steps for creating an event.

By default, Elemental Live does not ingest any captions (not even captions that are embedded in the video). You must explicitly identify the captions to ingest and the captions to output.

The information in this section assumes that you are familiar with the general steps for creating an event.

Topics

Supported features

This section provides information on the various features of captions that Elemental Live supports.

Topics

Supported formats

For information about the captions that Elemental Live can ingest, and the captions that it can produce, see the section called “Reference: Supported captions”.

Definitions of captions categories

Elemental Live groups captions formats into several categories. You set up captions differently, depending on the category.

You don't need to understand how the categories work. But you must know what category your input captions and your output captions belong to, so that you can follow the correct procedure.

The categories are the following:

- **Embedded**. The captions are carried inside the video encode. For example, 608 embedded captions.
- **Captions Object**. The captions are in their own "captions encode." They are not part of the video encode. But they are in the same output as their corresponding video and audio encodes. For example, DVB-Sub captions are object-style captions.
- **Sidecar**. The captions are each in their own output, separate from the output that contains the video and audio. The event can contain "captions-only" outputs, for example, one for each language. For example, TTML are sidecar captions.

For more information the category for each captions format, see the section called " Step 4: Match formats to categories".

Support for multiple languages

If the source includes captions in multiple languages, you can include multiple languages in the output as follows:

- **Embedded Passthrough**. For any of the embedded source formats, if you specify embedded as the output format, then all languages that are in the input are included in the output. You can't remove any of the languages.
- **Embedded In, Other Out**. For any of the embedded source formats, if you are doing "embedded in, other out," you can specify which languages to extract from the input.
- **Teletext Passthrough**. For teletext sources, if you specify teletext as the output, then all languages (pages) are included in the output. You can't strip out any languages. In fact, the entire teletext content is included in the output; you can't strip out any of the pages. Furthermore, teletext passthrough is supported only in TS outputs.
- **Teletext In, Other Out**. For teletext source, if you are doing "teletext in, other out," you can specify which languages (teletext pages) to extract and which languages to include in an output.
- **Any Other Combination**. For all other sources, you always specify the language to extract from the input and the language to include in an output, regardless of the source format and output format.

Support for conversion using OCR optical character recognition)

You can convert source captions that are in DVB-Sub or SCTE-27 format into output captions that are in WebVTT format. AWS Elemental Live uses OCR (optical character recognition) to perform this conversion.

Enabling the OCR feature

If you want to use OCR conversion, you must enable the feature when you install or upgrade Elemental Live. For more information, see these guides:

- AWS Elemental Live Installation Guide
- AWS Elemental Live Upgrade Guide

You can determine if the feature has already been enabled. In the event, go to the output section and start to set up a captions encode. Choose WebVTT as the output format. If the feature is enabled, a list of languages (language libraries) appears.

> **Note**
>
> When you enable OCR, you must make sure that you don't use the `--skip-all` option with the command to install, upgrade or configure. If you use that skip option, you won't see the prompts to enable OCR conversion.

Using OCR conversion

For more information about setting up to convert captions using OCR, see the section called "Sidecar captions or SMPTE-TT captions in MS Smooth".

For a list of languages supported with OCR conversion, see the section called "Reference: OCR languages".

Support for font styles in output captions

Depending on the scenario, there are three possibilities for the font style for output captions:

- You can specify the style you want for fonts, including color, outline, and background color.
- The font styles in the input are passed through.
- The font styles are controlled by the downstream player.

Font style options

Source captions	Output captions	Options for font style
ARIB	ARIB	None. The font styles in the input are automatically passed through in the output.
SCTE-27	SCTE-27	None. The font styles in the input are automatically passed through in the output.
DVB-Sub	DVB-Sub	None. The font styles in the input are automatically passed through in the output.
Teletext	Teletext	None. The font styles in the input are automatically passed through in the output.
Teletext	DVB-Sub	None. The font styles in the input are automatically passed through in the output.
Any supported captions format	Burn-in	You can specify font styles in the output. If you don't specify styles, the Elemental Live defaults are used.
Any supported captions format	DVB-Sub	You can specify font styles in the output. If you don't specify styles, the Elemental Live defaults are used.
An Embedded Combinati on (Embedded, Embedded +SCTE-20, SCTE-20+E mbedded)	CCF-TT or TTML	The font information in the source can be copied to the output, or you can let the downstream player determine the font style.

Source captions	Output captions	Options for font style
Teletext or SMPTE-TT or TTML or CCF-TT	CCF-TT or TTML	The font information in the source can be copied to the output, or you can let the downstream player determine the font style.
Any Other	Any Other	No control: the font style is always determined by the downstream player.

Captions in events with multiple inputs

If your event includes multiple inputs, these rules apply to Elemental Live handling of captions:

- The captions formats in one input can be different from the captions formats in another input. For example, 608 embedded captions might be in one input and teletext might be in another.
- There is no requirement for all the inputs to have captions that are capable of producing the specified captions in any given output.

  If the captions from an input cannot produce the specified captions in one of the outputs, the captions will be omitted for the course of that input. The event will not fail. When the event switches to a different input, the captions will be included again if the captions from that input can produce the specified captions in that output.

Captions and input switching setups

Your input might include a backup input that is only switched to if the first input fails. (This feature is called Input Switching or Input Switching with “Hot Hot” backup.) Or your input might include multiple inputs, each with its own backup input – several “input pairs.” The same rules apply as for multiple inputs: the captions in all the inputs must be identical for captions to work smoothly in the output.

Typical scenarios

Topics

- Use case: One input format to one output and not converted
- Use case: One input format converted to one different format in one output
- Use case: One input format converted to different formats, one format for each output
- Use case: One captions output shared by multiple video encodes

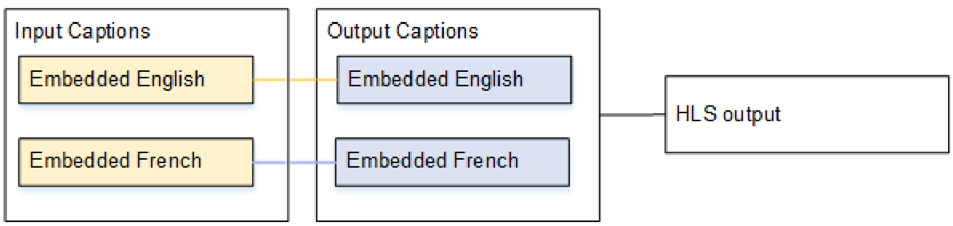
The following sections describe some typical scenarios for setting up captions in the event.

These four scenarios demonstrate a range of use cases.

Use case: One input format to one output and not converted

In this case, the input is set up with one format of captions and two or more languages (in the graphic below, you see both English and French.). Assume that you want to maintain the format in the output, that you want to produce only one type of output, and that you want to include all languages in that output.

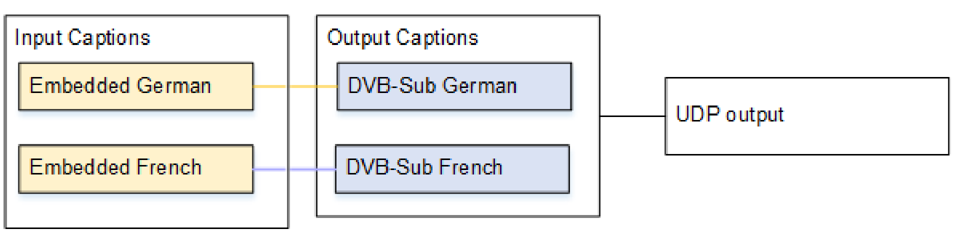
For example, the input has embedded captions in English and French. You want to produce HLS output that includes embedded captions in both English and French.



Use case: One input format converted to one different format in one output

The input is set up with one format of captions and two or more languages. You want to convert the captions to a different format in the output. You want to produce only one type of output and include all the languages in that output.

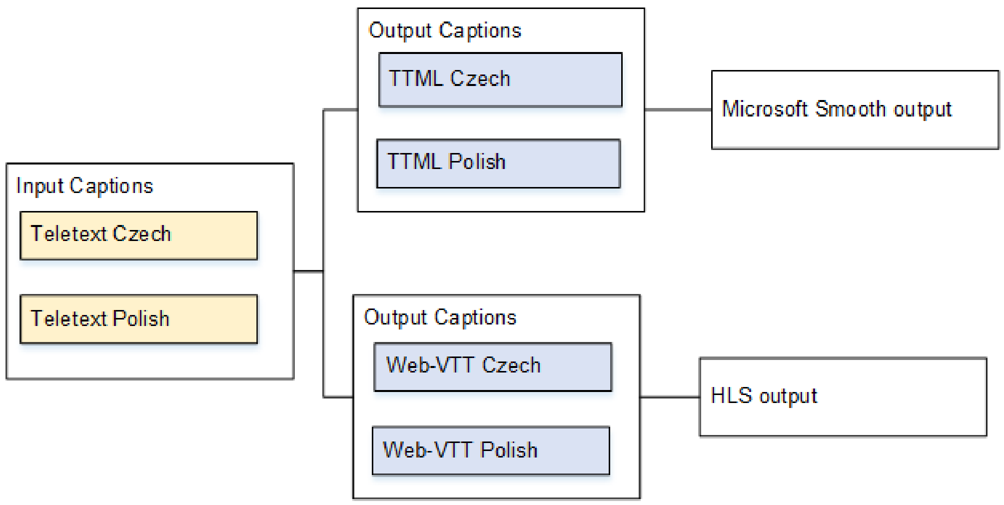
For example, the input has embedded captions in German and French. You want to convert the captions to DVB-Sub and include these captions in both languages in a UDP output.



Use case: One input format converted to different formats, one format for each output

The input is set up with one captions format and two or more languages. Assume that you want to produce several different types of output, and that in each output you want to convert the captions to a different format, and include all the languages.

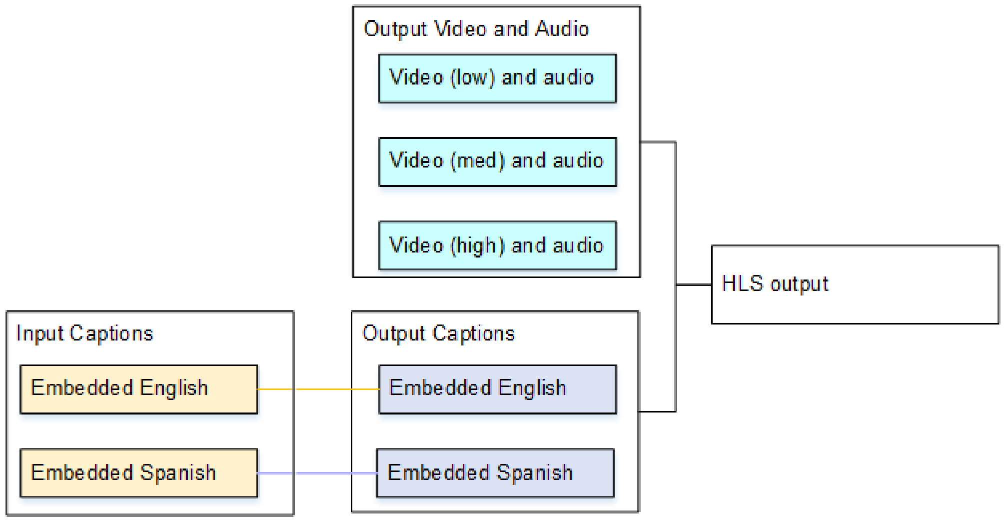
For example, the input has teletext captions in Czech and Polish. You want to produce an MS Smooth output and an HLS output. In the MS Smooth output, you want to convert both captions to TTML. In the HLS output, you want to convert both captions to WebVTT.



Use case: One captions output shared by multiple video encodes

This use case deals with captions in an ABR workflow. In this example, one captions output is shared by several video encodes.

For example, assume that there are three video/audio media combinations: one for low-resolution video, one for medium, and one for high. Assume that there is one output captions asset (English and Spanish embedded) that you want to associate with all three video/audio media combinations.



Setting up for captions

When you create an event, you must specify the format of the input captions. On the output side, you must specify the desired formats of the captions for each output. When you save the event, Elemental Live validates your choices in terms of whether the specified input format can produce the specified output format, and whether that output format is supported in the specified output type.

Topics

Step 1: Identify the source captions that you want

You must identify the captions that you want to use and assign each to a captions selector. If you don't create any captions selectors, you will not be able to include captions in the output. All the captions will be removed from the media.

To identify the captions you want

1. Identify which captions are in the input (the provider of the input should provide you with this information) and identify which captions are available to you as external files. Identify the captions formats and, for each format, the languages.
2. Identify which of those formats and languages that you want to use.
3. Determine how many captions selectors to create in the input in the event, using the following guidance:
   - For embedded passthrough, create a single captions selector for all languages. All languages are passed through; there is no other option. For details, see the section called “Information for embedded ”.
   - For embedded-to-other-format, create one captions selector for each language.
   - For teletext passthrough, create a single captions selector for all languages (in fact, one captions selector for the entire content). All languages (teletext pages) are passed through; there is no other option. For details, see the section called “Information for Teletext”.
   - For teletext-to-other-format, create one captions selector for each language.
   - In all other cases, create one captions selector for each language and format combination.
4. You end up with a list of captions selectors to create. For example:
   - Captions Selector 1: teletext captions in Czech
   - Captions Selector 2: teletext captions in Polish

   You are not required to use all the languages that are available. You can ignore those you are not interested in.

Step 2: Create captions selectors

After you have created a list of captions selectors, you can create the captions selectors in the event.

To create the captions selectors

1. In the event, in the **Input** section, choose **Advanced**.
2. Choose **Add Caption Selector**.
3. For **Source**, choose the format of the source captions.

To identify SMPTE-TT as the source captions, choose **TTML**. When Elemental Live ingests the captions, it automatically detects that they are SMPTE-TT.

4. For most formats, more fields appear. For details about a field, choose the Info link next to the field. In addition, see extra information about DVB-Sub or SCTE-27, on Embedded, on SCC, on SMI, SRT, STL, TTML, on teletext, or on Null.
5. Create more captions selector, as required.

Information for DVB-Sub or SCTE-27

This section provides information specific to DVB-Sub or SCTE-27 input captions. It describes the fields that appear when you choose **DVB-Sub** or **SCTE-27** in the **Source** field in the **Caption Selector** section of the event. For more context, see the steps earlier in this section.

DVB-Sub and SCTE-27 formats are supported only in TS inputs. You must specify the location of the captions.

Complete the **PID** and **Language code** fields in one of the ways described in the following table. Each row in the table describes a valid way to complete these two fields.

PID	Language code	Result
Specified	Blank	Extracts the captions from the specified PID.
Blank	Specified	Extracts the captions from the first PID that Elemental Live encounters that matches the specified language. This might or might not be the PID with the lowest number.
Specified	Specified	Extracts the captions from the specified PID. Elemental Live ignores the language code, therefore we recommend you leave it blank.

PID	Language code	Result
Blank	Blank	Valid only if the source is DVB-Sub and the output is DVB-Sub. With this combinati on of PID and Language, all input DVB-Sub PIDs will be included in the output. Not valid for SCTE-27.

Information for embedded

This section provides information specific to embedded input captions. It describes the fields that appear when you choose **Embedded** in the **Source** field in the **Caption Selector** section of the event. For more context, see the steps earlier in this section.

Read this section if the input captions you have are any of the following: embedded (EIA-608 or CEA-708), embedded+SCTE-20, SCTE-20+embedded, or SCTE-20.

> **Note**
>
> For captions in VBI data: If you are extracting embedded captions from the input and using embedded captions in the output, and if the input includes VBI data and you want to include all that data in the output, then do not follow this procedure. Instead, see Passing through VBI data.

Determining the number of captions selectors needed

To determine the number of captions selectors you need to created in the event, follow these rules:

- **Embedded Passthrough** – Create only one captions selector. With this scenario, all languages are automatically extracted and are automatically included in the output.
- **Embedded In, Other Out** – If you are setting up embedded-to-other, create one captions selector for each language that you want to include in the output, to a maximum of four selectors.

- **A combination of Embedded passthrough and Embedded conversion** – If you are setting up embedded passthrough in some outputs and embedded-to-other in other outputs, create one captions selector for each language that you want to include in the output, to a maximum of four selectors. Do not worry about a selector for the embedded passthrough output. Elemental Live will extract all the languages for that output, even though no selector exists to explicitly specify this action.

Completing the fields in the captions selector group

- **Source**:
  - Choose embedded if the source captions are embedded (EIA-608 or CEA-708), embedded +SCTE-20, or SCTE-20+embedded.
  - Choose SCTE-20 if the source captions are SCTE-20 alone.

Completing the fields in the CC channel number

- **CC Channel number**: This field specifies the language to extract. Complete as follows:
  - If you are setting up embedded passthrough only (you are creating only one captions selector for the input embedded captions), this field is ignored, so keep the default.
  - If you are setting up embedded-to-another-format, (you are creating several captions selectors, one for each language), enter the number of the CC instance (from the input) that holds the desired language. For example, if this captions selector is intended to hold the French captions and the French captions are in event 2, enter 2 in this field.

- **Force 608 to 708 Upconvert**: The embedded source captions can be EIA-608 captions, CEA-708 captions, or both EIA-608 and CEA-708. You can specify how you want these captions to be handled when Elemental Live is ingesting content. The following table describes the behavior for various scenarios.

EIA-608 in Source	CEA-708 in Source	Convert Field	Result
Yes	No	Checked	CEA-708 data is created based on the EIA-608 data. EIA-608 data is added as 608-compatibility bits in the CEA-708 data.
Yes	No	Unchecked	Original EIA-608 is preserved.
No	Yes	Checked	Original CEA-708 is preserved.
No	Yes	Unchecked	Original CEA-708 is preserved.
Yes	Yes	Checked	CEA-708 data is discarded. New CEA-708 data is created based on the EIA-608 data, and EIA-608 data is added as 608-compatibility bits in the CEA-708 data. The new CEA-708 data will not include any CEA-708 formatting features. Not recommended.

EIA-608 in Source	CEA-708 in Source	Convert Field	Result
Yes	Yes	Unchecked	Original EIA-608 is preserved and original CEA-708 is preserved.

- **Use SCTE-20 if Embedded Unavailable**: This field appears only if you set the **Source** to **Embedded**. If the source captions combine embedded (EIA-608 or CEA-708) and SCTE-20, you might want to set this field to **Auto**. Elemental Live will give preference to the 608/708 embedded captions but will switch to use the SCTE-20 captions when necessary. If you set this field to Off, Elemental Live will never use the SCTE-20 captions.

Information for SCC

This section provides information specific to SCC input captions. It describes the fields that appear when you choose **SCC** in the **Source** field in the **Caption Selector** section of the event. For more context, see this procedure.

SCC source captions are supplied in a captions file that is external to the video input. You must specify this file.

- **External Caption File**: Specify the location of the file.
- **Time Delta**: Complete this field to adjust the timestamp in the caption file. With the SCC files, the situation sometimes arises where the timestamp in the file for the first captions does not work with the video. The start time for the video/audio always 00:00:00. The start time of the captions may not be 00:00:00 – it may be some completely different, arbitrary time, such as 20:00:15. Assume that, in the video, the first words are spoken at 00:06:15. But given that the start time for the captions file is 20:00:15, then the time for the first caption will be marked as 20:06:30. This time will usually never work with the video. The solution is to adjust the times in the captions file. In this example, subtract 20 hours and 15 seconds (72015 seconds) from the captions file.

  Enter a value in this field to push the captions earlier or later:

  - Enter a positive number to add to the times in the caption file. For example, enter **15** to add 15 seconds to all the times in the caption file.
  - Enter a negative number to subtract from the times in the caption file. For example, enter **-5** to remove 5 seconds from all the times in the caption file.

The format of the times in the captions does not have to match the value in the **Timecode Config** field (in the Input) of the video. The number you enter in this field will simply delay the captions or make the captions play earlier, regardless of the formats.

When using SCC, the video must absolutely have a value in the **Timecode Config** field. Otherwise the captions will not be inserted.

- **Force 608 to 708 Upconvert**: SCC source captions are EIA-608 format and are contained in an external file. The options for converting the caption are the following:
  - Check: To convert the captions to CEA-708 format.
  - Unchecked: To leave the captions unconverted.

Information for SMI, SMPTE-TT, SRT, STL, TTML

This section provides information specific to SMI, SMPTE-TT, SRT, STL, and TTML input captions. describes the fields that appear when you choose **SCC** in the **Source** field in the **Caption Selector** section of the **Create New Live Event** screen. For more context, see the section called “Step 1: Identify source captions”.

With these formats, the source captions are supplied in a captions file that is external to the video input. You must specify this file.

- **External Caption File**: Specify the location of this file.
- **Time Delta**: Complete this field to adjust the timestamp in the caption file. With the SCC files, the situation sometimes arises where the timestamp in the file for the first captions does not work with the video. With these types of captions, the start time for both the video/audio always 00:00:00. Assume that, in the video, the first words are spoken at 00:06:15. But in the captions file, this time is marked as 00:06:18, and every other caption is also off by 3 seconds. The solution is to adjust the times in the captions file. In this example, subtract 3 seconds from the captions file.

  Enter a value in this field to push the captions earlier or later.
  - Enter a positive number to add to the times in the caption file. For example, enter **2** to add 2 seconds to all the times in the caption file.
  - Enter a negative number to subtract from the times in the caption file. For example, enter **-3** to remove 3 seconds from all the times in the caption file.

Information for Teletext

This section provides information specific to Teletext input captions. It describes the fields that appear when you choose **SCC** in the **Source** field in the **Caption Selector** section of the event. For more context, see the section called "Step 1: Identify source captions".

Teletext is a form of data that can contain several types of information, not just captions. Teletext can be present in SDI input, in MXF input, and in TS input, in which case it might be referred to as "DVB teletext.".

You can set up to handle teletext in one of the following ways:

- If you want to extract the entire teletext input, you must set up teletext passthrough. The entire teletext can never be converted to another format. Teletext passthrough is supported only in a TS output.
- You can extract individual captions pages (the captions in a specific language) and convert them to another captions format.
- You cannot extract individual captions pages (the captions in a specific language) and keep them in teletext.

Determining the number of captions selectors needed

- If you are setting up teletext passthrough captions, create only one captions selector, even if you want to include multiple languages in the output. With this scenario, all languages are automatically extracted and are automatically included in the output.
- If you are setting up teletext-to-other, create one captions selector for each language that you want to include in the output. For example, one selector to extract English teletext, and one selector to extract Swedish teletext.
- If you are setting up teletext passthrough in some outputs and teletext-to-other in other outputs, create individual selectors for the teletext-to-other, one for each language being converted. Do not worry about a selector for the teletext passthrough output. Elemental Live will extract all the data in the teletext, even though there is not a selector to explicitly specify this action.

Completing the fields in the Captions Selector Group

- **Source**: Choose **Teletext**.

- **Page**: This field specifies the page of the desired language. Complete as follows:
  - If you are setting up teletext passthrough captions (you are creating only one captions selector for the input captions), leave blank: the value is ignored.
  - If you are converting teletext to another format (you are creating several captions selectors, one for each language), specify the page for the desired language. If you leave this field blank, you get a validation error when you save the event.

Information for null

The list of sources in the **Sources** field for the caption selector includes the option **Null**. This source is not intended for stripping out captions. Instead, it is used for 608 XDS data.

Step 3: Plan captions for the outputs

If you followed the instructions in the section called "Step 1: Identify source captions", you should have a list of the captions formats and languages that will be available for inclusion in the outputs.

You must now plan the captions information for the outputs.

To plan captions for the outputs

1. Identify the types of output media that you plan to create in the event. For example, MS Smooth and HLS.
2. Identify the streams (the combinations of video and audio) that you plan to create for each output media.
3. Map each output to the stream it uses. For example:

   - HLS (Output 1) uses video/audio Stream 1.
   - DASH (Output 2) also uses video/audio Stream 1. (Or it might need its own stream if the video requirements are different.)
4. For each output media, identify which input captions will be converted to which output formats. For example, you might convert teletext captions to TTML for the MS Smooth output media, and those same teletext captions to WebVTT for the HLS output media.

   The output formats that are possible depend on the input formats and the type of output media. See Reference: Supported captions to determine which output captions are possible given the input format.
5. Identify the languages for each output format:

- In general, count each language separately.
- Exception: For embedded passthrough, count all languages as one.
- Exception: For teletext passthrough, count all languages as one.

The Result

You end up with a list of outputs, and the captions formats and languages for each output. For example:

- MS Smooth output with TTML captions in Czech
- MS Smooth output with TTML captions in Polish
- HLS output with WebVTT captions in Czech
- HLS output with WebVTT captions in Polish.

Planning for Output in Multiple Formats

You can include captions from two or more different formats in an output. For example, you can include both embedded captions and WebVTT captions in an HLS output, to give the downstream system more choices about which captions to use. The only rules for multiple formats are the following:

- The output container must support all the formats. See Reference: Supported captions.
- The font styles in all the captions that are associated with an output must match. This means that the end result must be identical, not that you must use the same option to get that result. For example, all captions that are associated with the output must be white for the first language and blue for the second language.

Managing this style matching can be a little tricky. For information about the font style options, see Support for font styles in output captions.

Step 4: Match formats to categories

There are different procedures to follow to create captions encodes in the output. The correct procedure depends on the "category" that the output captions belong to. There are five categories of captions, described in the following table.

On the list of outputs that you have created, make a note of the category that each captions option belongs to.

Format of output captions	Category of this format
Ancillary+Embedded	The captions in ancillary format are in the ancillary data in the stream. The embedded captions are embedded in the video. To choose Ancillary+Embedded, choose **Embedded** as the **Destination Type** (in the procedure). Elemental Live will automatically produce both Ancillary and embedded.
ARIB	Object
Burn-in	Burn-in
DVB-Sub	Object
Embedded	Embedded
Embedded+SCTE-20	Embedded
RTMP CaptionInfo	Object
RTMP CuePoint	Object
SCC	Sidecar
SCTE-20+Embedded	Embedded
SCTE-27	Object
SMI	Sidecar
SMPTE-TT	Sidecar when in Archive
SMPTE-TT	Stream when in MS Smooth
SRT	Sidecar

Format of output captions	Category of this format
teletext	Object
TTML wrapped in ID3 data	Wrapped in ID3 data
TTML	Sidecar
WebVTT	Sidecar

For example, your list of outputs might now look like this:

- MS Smooth output with TTML captions (sidecar) in Czech.
- MS Smooth output with TTML captions (sidecar) in Polish.
- HLS output with WebVTT captions (sidecar) in Czech.
- HLS output with WebVTT captions (sidecar) in Polish.

Captions embedded in video

The captions are carried inside the video encode, which is itself in an output in the output group. Only one captions asset ever exists within that video encode. That single asset might contain captions for several languages.

Captions object

The captions are in their own "captions encode" in an output in the output group. They are not part of the video encode. However, they are in the same output as their corresponding video and audio encodes. There might be several captions encodes in the output, for example, for different languages.

Sidecar

The captions are each in their own output in the output group, separate from the output that contains the video and audio. Each captions output contains only one captions asset (file), meaning that it is a "captions-only" output. The output group might contain several "captions-only" outputs, for example, one for each language in the output group.

TTML captions wrapped in ID3 data

The captions are converted to TTML and included in ID3 data. (The other way to produce TTML output is as a sidecar.)

SMPTE-TT in MS Smooth

The captions are handled as a separate stream in MS Smooth.

Note that SMPTE-TT captions for other package types are handled as sidecars. However, for both sidecar handling and stream handling, the procedure for setting up SMPTE-TT captions in the output is identical. Elemental Live will package the SMPTE-TT captions correctly for the package type.

Burn-in

Here, the captions are converted into text and then overlaid on the picture directly in the video encode. Strictly speaking, once the overlay occurs, these are not really captions because they are indistinguishable from the video.

Step 5: Create captions encodes

Go through the list of outputs you created and set up the captions in each output group, one by one.

Follow the procedure that applies to the format category of the captions output.

Topics

- All captions except sidecar or SMPTE-TT in MS Smooth
- Sidecar captions or SMPTE-TT captions in MS Smooth
- TTML captions wrapped in ID3 data
- Setting up for 608 XDS data

All captions except sidecar or SMPTE-TT in MS Smooth

Follow this procedure if the format of the captions asset that you want to add belongs to the category of embedded, burn-in, or object. You will set up the captions and video and audio in the same output.

To create captions (*not* sidecar or SMPTE-TT)

1. On the web interface, on the **Event** screen, click the appropriate output group.
2. If you have already set up this output group with video and audio, find the outputs where you want to add the captions. Or if you have not set up with video and audio, create a new output in this output group; you can set up the captions now and you can set up the video and audio later.
3. Go to the output, then go to the stream that is associated with that output. For example, go to Stream 1.
4. Click the + beside **Caption** to add a Caption section.
5. Complete the fields that appear for the selected format. For details about a field, choose the Info link beside the field.

Field	Applicability	Description
Caption Source	All formats	Select the Caption Selector that you created when you specified the input captions.
Destination Type	All formats	Select the caption type.
Pass Style Information	If Destination Type is CCF-TT or TTML. And if the source caption type is an Embedded combination (Embedded , Embedded+SCTE-20, SCTE-20+Embedded), or Teletext, or TTML, or SMPTE-TT, or CCF-TT.	The choices are: • Check this box if you want the style (font, position and so on) of the input captions to be copied. • Leave unchecked if you want a simplified caption style. Some client players work best with a simplifie d caption style. (For other combinations of source caption types and

Field	Applicability	Description
		output caption type, the output is always simplified.)
Font style fields	Destination type is Burn-in or DVB-Sub	For tips about font styles in DVB-Sub or burn-in, see Font Styles for Burn-in or DVB-Sub Output.
Language	All captions except not for embedded-to-embedded or teletext-to-teletext.	Complete if desired. This information may be useful to or required by a downstream system. For embedded-to-embedded or teletext-to-teletext, leave as Undefined.
Description	All captions except not for embedded-to-embedded.	This field is auto-completed after you specify the language.
Use ID3 as Caption Content	Destination type is TTML And if this stream is associated with an MS Smooth output.	Leave unchecked. This field applies only when wrapping TTML captions in ID3; see the section called "TTML captions wrapped in ID3 data".

6. If the output format is embedded and the output group is HLS, you can include captions language information in the manifest. You perform this setup in the output settings (separate from the captions encode). See the section called "Set up the HLS Manifest (embedded captions)".
7. If the output format is ARIB or DVB-Sub or SCTE-27, you must perform some extra setup in the output settings (separate from the captions encode). See PIDS for ARIB output or PIDs for DVB-Sub output or PIDs for teletext output.
8. You now have a captions encode that is fully defined.

9. Repeat these steps to create captions, as applicable.
10. Go to the output group and output that this stream belongs to. Set the Stream field in that output to match the stream you created.
11. When you are ready, save the event.

    If the “Caption Stream Incompatible” message appears, see the section called “"Caption Stream Incompatible" message”.

Font styles for Burn-in or DVB-Sub Captions

When you set up the captions encode as described in the section called “All captions except sidecar or SMPTE-TT in MS Smooth”, you can specify the appearance of the captions if the output captions are Burn-in or DVB-Sub. In the following table, the first column shows the field name, the third column specifies how to complete the field, and the third column specifies whether the description applies to Burn-in or DVB-Sub.

Name	Description	Applicability
Font	Click **Browse** to find a font file to use. The file must be on a server mounted to the node and must have the extension TTF or TTE. Do not specify a font file if the caption source is embedded or teletext.	Both
Font Size	Specify **auto** or enter a number. When set to auto, font_size will scale depending on the size of the output. Giving a positive integer will specify the exact font size in points.	Both

Name	Description	Applicability
Font Resolution	Font resolution in DPI (dots per inch). Range: 96 to 600. Default is 96 dpi.	Both
Text Justify	For conversions from STL to Burn-in: This field is ignored; the justification specified in the input STL file is always used. For all other conversions to Burn-in: • Centered: If X Position and Y Position are both empty, positions the captions at the bottom center of the video frame. If X Position and Y Position are specified, the captions are offset and then centered across the video frame. • Left: If X Position and Y Position are both empty, positions captions at the bottom left of the video frame. If X Position and Y Position are specified, the captions are offset and then left-alig ned.	Burn-in

Name	Description	Applicability
Text Justify	• Centered: If X Position and Y Position are both empty, positions the captions at the bottom center of the video frame. If X Position and Y Position are specified, the captions are offset and then centered across the video frame. • Left: If X Position and Y Position are both empty, positions captions at the bottom left of the video frame. If X Position and Y Position are specified, the captions are offset and then left-alig ned.	DVB-Sub

Name	Description	Applicability
X Position	For conversions from STL to Burn-in: This field is ignored; the position specified in the input STL file is always used. For all other conversions to Burn-in: • Offset for the left edge of the caption relative to the horizontal axis of the video frame, in pixels. 0 is the left edge of the video frame. 10 pixels means offset 10 pixels to the right. • Empty means 0 offset.	Burn-in
X Position	Offset for the left edge of the caption relative to the horizontal axis of the video frame, in pixels. 0 is the left edge of the video frame. 10 pixels means offset 10 pixels to the right. Empty means 0 offset.	DVB-Sub

Name	Description	Applicability
Y Position	For conversions from STL to Burn-in: This field is ignored; the position specified in the input STL file is always used. For all other conversions to Burn-in: • Offset of the top edge of the caption relative to the vertical axis of the video frame, in pixels. 0 is the top edge of the video frame. 10 pixels means offset 10 pixels from the top. • Empty means position the captions towards the bottom of the output.	Burn-in
Y Position	Offset of the top edge of the caption relative to the vertical axis of the video frame, in pixels. 0 is the top edge of the video frame. 10 pixels means offset 10 pixels from the top. Empty means position the captions towards the bottom of the output.	DVB-Sub

Name	Description	Applicability
Fixed Grid	Applies only for conversions from Teletext. • Checked (default): Font is mono-spaced: each character takes up the space horizontal space. • Unchecked: Font is proportionally spaced. (Note that for conversions from STL to Burn-in, the font is always mono-spaced; this information is never in the STL and the value in the field is ignored.)	Burn-in
Fixed Grid	Applies only for conversions from Teletext. • Checked (default): Font is mono-spaced: each character takes up the space horizontal space. • Unchecked: Font is proportionally spaced. (Note that for conversions from STL to DVB-Sub, the font is always mono-spaced; this information is never in the STL and the value in the field is ignored.)	DVB-Sub

Name	Description	Applicability
Font Color	For conversions from STL: The font color is taken from the STL file but the value in Font Color is used to override as follows: • If the STL file contains some white text of some "unspecified color" text, then setting the Font Color field changes the white (or unspecified) text. Text in other colors can't be changed. For example, if the file contains white and blue text, the white text is changed. • If the file contains specified text and yellow text, the unspecified text is changed. For all other conversions to Burn-in: Select the desired color.	Burn-in
Font Color	Select the desired color.	DVB-Sub
Font Opacity	The opacity for the font color. Range 0 (transparent) to 255 (opaque).	Both
Background Color	The color for the background rectangle.	Both

Name	Description	Applicability
Background Opacity	The opacity for the backgroun d rectangle. Range 0 (transpar ent) to 255 (opaque).	Both
Outline Size	The size for the font outline, in pixels. Range 0 (no outline) to 10.	Both
Outline Color	The color for the font outline.	Both
Shadow Color	The color for the shadow cast by the captions.	Both
Shadow Opacity	The opacity of the shadow, in pixels. Range 0 (transpar ent) to 255 (opaque). Empty means 0.	Both
Shadow X Offset	The horizontal offset of the shadow, in pixels. A value of -2 results in a shadow offset 2 pixels to the left. A value of 2 results in a shadow offset 2 pixels to the right.	Both
Shadow Y Offset	The vertical offset of the shadow, in pixels. A value of -2 results in a shadow offset 2 pixels above the text. A value of 2 results in a shadow offset 2 pixels above the text.	Both

Font Styles When You Use the Same Source in Several Outputs

If you are using the same caption source in several Stream sections (in other words, you are selecting the same Caption Selector in the Caption Source field in several Stream sections), then you must set up the font style information identically in each Stream section. If you do not, you will get an error when you save the event.

For example, stream 1 may use Caption Selector 1 with the Destination Type set to Burn-in. And stream 2 may also use Caption Selector 1 with the Destination Type set to Burn-in. You set the font information once in stream 1 and again in stream 2. You must make sure to set up all the font information identically in both streams.

The same rule applies if the output captions are all DVB-Sub.

Complete the PIDs for ARIB

This section applies when you set up the captions encode as described in the section called "Step 1: Identify source captions", if the output group is UDP/TS and the output captions format is ARIB. It describes how to complete the PIDs for the output that contains these captions.

To complete the PIDs (ARIB)

1. In the Output section, open the PID Control section.
2. Complete the ARIB Captions field and the ARIB Captions PID field as follows:

ARIB Captions PID Control	ARIB Captions PID	Result
Unchecked	Ignore.	A PID will automatically be assigned during encoding; this value could be any number.
Checked	Type a decimal or hexadecim al.	This PID will be used for the captions.
	Leave the default (507)	The PID for captions will be 507.
	Delete the default	A PID will automatically be assigned during encoding; this value could be any number.

Complete the PIDs for DVB-Sub

This section applies when you set up the captions encode as described in the section called “Step 1: Identify source captions”, if the output group is UDP/TS and the output captions format is DVB-Sub. It describes how to complete the PIDs for the output that contains these captions.

To complete the PIDs (DVB-Sub)

1. In the Output section, open the PID Control section.
2. In the **DVB Subtitle PIDs** field, enter the PID for the DVB-Sub caption in the stream for this output. Or leave the default.

Complete the PIDs for Teletext

This section applies when you set up the captions encode as described in the section called “Step 1: Identify source captions”, if the output group is UDP/TS and the output captions format is teletext. It describes how to complete the PIDs for the output that contains these captions.

To complete the PIDs (teletext)

1. In the Output section, open the PID Control section.
2. In the **DVB Teletext PID** field, enter the PID for the Teletext caption in the stream for this output. Or leave the default.

Set up the HLS Manifest (embedded captions)

This section applies when you set up the captions encode as described in the section called “Step 1: Identify source captions”, if the output group is HLS and the output captions format is embedded. It describes how to include captions language information in the manifest.

To specify language information in the manifest

1. In the HLS output group, go to the output. Click **Advanced**.
2. Complete **Caption Languages** as desired:
   - Omit: To omit any CLOSED-CAPTION lines in the manifest.
   - None: To include one CLOSED-CAPTION=None line in the manifest.
   - Insert: To insert one or more lines in the manifest.

3. If you chose **Insert**, more fields appear. Complete on more sets of fields.
    - You should complete as many fields as there are languages in this output.
    - The order in which you enter the languages must match the order of the captions in the source. For example, if the captions are in the order English, then French, then Spanish, then Portuguese, then set up CC1 as English, CC2 as French, and so on. If you do not order them correctly, the captions will be tagged with the wrong languages.

"Caption Stream Incompatible" message

When you save the event, this validation message might appear:

Stream Caption Destination Type Is Incompatible With XX Output...

Typically, this error will occur because of the following scenario:

- You have two outputs – perhaps HLS and DASH – that will have the same audio and video descriptions, which means you want them to share the same stream:
- You set up the HLS output group and add an Output and Stream 1. You add embedded captions.
- You then set up the DASH output group and add an Output and associate that output with the existing Stream 1.
- The problem is that DASH cannot contain embedded captions. Therefore, when you save the event, you will get the validation message.

The solution to this problem is:

- When you set up the DASH output, instead of associating it with the existing Stream 1, create a new stream (Stream 2)
- In Stream 2, set up the video and audio to be identical to the video and audio in Stream 1.
- For the DASH output, add the captions in the appropriate way.

The result: Assuming that you have set up the video and audio in both streams to be identical, the encoder will notice that they are identical and will in fact encode the video only once and the audio only once. So there will be no extra video encoding load from creating separate streams.

Sidecar captions or SMPTE-TT captions in MS Smooth

Follow this procedure if the format of the captions asset that you want to add is a sidecar, as identified in the section called “ Step 4: Match formats to categories”, or if the format is SMPTE-TT for an MS Smooth output.

When you follow this procedure, you set up each captions asset in its own output within the output group. When the event runs, the captions will be set up as sidecars in the output package, except for SMPT-TT captions in MS Smooth, which will be set up as streams in the output package.

To create captions (sidecar and SMPTE-TT)

1. On the web interface, on the **Event** screen, click the output group. (You should have already created this output group).
2. In the output group, choose **Add Output**. A new output appears, and by default this output has one stream. Make note of this stream. For example, **Stream 2**.
3. In the **Streams** section (for example, in **Stream 2**), hover over **Video** and choose the **x** icon. Hover over **Audio** and choose the **x** icon. The stream is now empty.
4. Beside **Captions**, choose the **+** icon. The stream now contains one captions encode and no video or audio encodes.
5. Complete the fields as shown in the table after this procedure.
6. Repeat these steps to create more sidecar captions in this or another output group, as applicable.
7. When you are ready, save the event.

   If the “Caption Stream Incompatible” message appears, see the section called “"Caption Stream Incompatible" message”.

Field	Applicability	Description
Caption Source	All	Select the Caption Selector you created earlier.
Destination Type	All	Select the caption type. This type must be valid for your output type as per the relevant Supported Captions

Field	Applicability	Description
		table. See Reference: Supported captions.

Field	Applicability	Description
Optical Character Recogniti on Language	The Destination Type is WebVTT	Complete this field only if the source captions in the chosen Caption Selector are DVB-Sub or SCTE-27. Specify the language of the captions in the source. This captions conversion uses OCR (optical character recognition) technolog y. You must identify the language of the captions to ensure that Elemental Live chooses the correct OCR library for the conversio n. The library speeds up conversion because it allows checking character strings against a dictionary, instead of recognizing words letter by letter. If you choose a language that doesn't match the language of the captions, conversion accuracy will be poor. Elemental Live ignores this field if you aren't convertin g DVB-Sub or SCTE-27 captions to WebVTT. For a list of languages supported with OCR conversion, see the section

Field	Applicability	Description
		called "Reference: OCR languages".
Framerate	The Destination Type is SCC.	Complete this field to ensure that the captions and the video are synchronized in the output. Specify a framerate that matches the framerate of the associated video. • If the video framerate is 23.97 or 24, choose the corresponding option. • If the video framerate is 29.97, choose 29.97 dropframe only if the video has the Video Insertion and Drop Frame Timecode both.

Field	Applicability	Description
Pass-style	The Destination Type is TTML	Complete this field only if: • The source caption type is TTML, or SMPTE-TT, or CCF-TT. • And the output is an Archive output. Complete as follows: • Check this box if you want the style (font, position and so on) of the input captions to be copied. • Leave unchecked if you want a simplified caption style. Some client players work best with a simplifie d caption style. (For other combinations of source caption types and output caption type, the output is always simplified.)
Font style fields	The Destination Type is Burn-in	See the table in the section called "Font styles".
Language	All	Complete if desired. This information may be useful to or required by a downstream system.

Field	Applicability	Description
Description	All	Complete if desired. This information may be useful to or required by a downstream system.

"Caption Stream Incompatible" message

When you save the event, this validation message might appear:

Stream Caption Destination Type Is Incompatible With XX Output...

Typically, this error will occur because of the following scenario:

- You have two outputs – perhaps HLS and RTMP – that will have the same audio and video descriptions, which means you want them to share the same stream:
- You set up the HLS output group and add an Output and Stream 1. You add embedded captions.
- You then set up the RTMP output group and add an Output and associate that output with the existing Stream 1.
- The problem is that RTMP cannot contain embedded captions. Therefore, when you save the event, you will get the validation message.

The solution to this problem is:

- When you set up the RTMP output, instead of associating it with the existing Stream 1, create a new stream (Stream 2)
- In Stream 2, set up the video and audio to be identical to the video and audio in Stream 1.
- For the RTMP output, add the captions in the appropriate way.

The result: Assuming that you have set up the video and audio in both streams to be identical, the encoder will notice that they are identical and will in fact encode the video only once and the audio only once. So there will be no extra video encoding load from creating separate streams.

TTML captions wrapped in ID3 data

Follow this procedure to produce an output that includes TTML captions wrapped in ID3 data. This format is supported only in an MSS output. Unlike unwrapped TTML captions (which you create as described in the section called “Sidecar captions or SMPTE-TT captions in MS Smooth”), these captions are included as an ID3 object in the same stream as the video.

To produce TTML captions wrapped in ID3 data

1. On the web interface, on the Event screen, click the appropriate output group.
2. In the output group, go to the output where you want to add captions.
3. Identify the stream that is associated with that output. In this example, there are two outputs; the first is associated with stream 1, the second is associated with stream 2.
4. Go to that Stream section. For example, go to Stream 1.
5. Click the + beside Caption to add a Caption section.
6. Complete the fields as shown in the table that follows this procedure.
7. Repeat these steps to add more captions for this output. For example, to add captions in another language.
8. Go to the MSS output group and output that this stream belongs to. Set the Stream field in that output to match the stream you created. For example:
9. When you are ready, save the event.

   If the “Caption Stream Incompatible” message appears, see the section called “"Caption Stream Incompatible" message”.

Field	Description
Caption Source	Select the Caption Selector you created when specifying the input captions.
Destination Type	Select the caption type. This type must be valid for your output type as per the relevant Supported Captions table.
Pass Style Information	Applicable only if the source caption type is an Embedded combination (Embedded,

Field	Description
	Embedded+SCTE-20, SCTE-20+Embedded), or Teletext, or TTML, or SMPTE-TT, or CCF-TT. The choices are: • Check this box if you want the style (font, position and so on) of the input captions to be copied. • Leave unchecked if you want a simplified caption style. Some client players work best with a simplified caption style. (For other source caption types, the output is always simplified.)
Language	Complete if desired. This information may be useful to or required by a downstream system.
Description	This field is automatically completed after you specify the language.
Use ID3 as Caption Content	Check this field, to insert the TTML captions into ID3 data.

Setting up for 608 XDS data

If your source content includes 608 XDS data, you can set up the event to include it or strip it from the output.

The Extended Data Services (XDS or EDS) standard is part of EIA-608 and allows for the delivery of ancillary data.

> **Note**
>
> You set up handling of this source data for the entire event, so you set up to either include it in every output and stream, or you set up to exclude it from every output and stream.

To configure handling of this data

1. In the Input section of the event, click **Advanced**.
2. Click the **Add Caption Selector** button.
3. Set the source to **Null**.

   You only need to create one Caption Selector for 608 XDS data, regardless of the number of outputs you are creating.
4. If you also want to extract regular captions, create more Caption Selectors according to the regular procedure.
5. In the **Global Processors** section, turn on **608 Extended Data Services** and complete the fields as desired.

> **Note**
>
> No setup is required in the captions section of the output or the streams.

Examples of implementing use cases

The following examples describe how to implement the use cases from the section called "Typical scenarios".

Topics

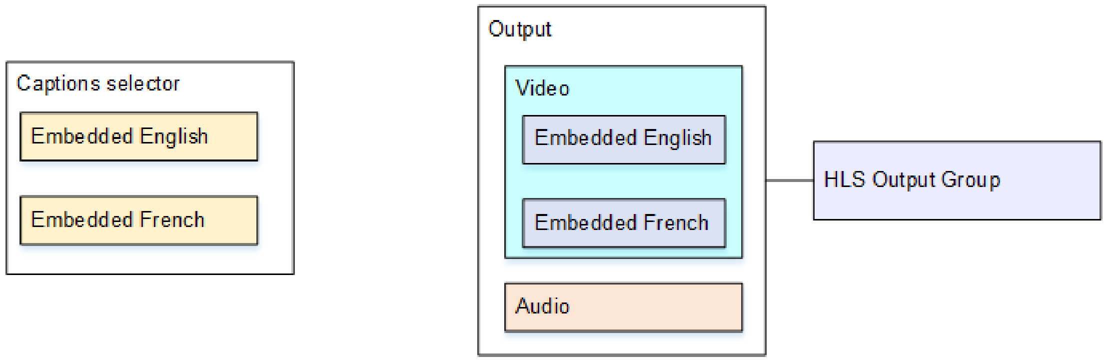
- Use case 1: one input format to one output
- Use case 2: One input format converted to one different output format
- Use case 3: One input format converted to different formats, one format for each output
- Use case 4: One captions output shared by multiple video encodes

Use case 1: one input format to one output

This example shows how to implement the first use case from the typical scenarios. The input is set up with one format of captions and two or more languages. Assume that you want to maintain the format in the output and that you want to produce only one type of output and include all the languages in that output.

For example, the input has embedded captions in English and French. You want to produce an HLS output that includes embedded captions in both English and French, plus one video and one audio.

This example illustrates two important features of an embedded passthrough workflow. First, you do not create separate captions selectors; all of the languages are all automatically included. Second, if you are outputting to HLS, you have an opportunity to specify the languages and the order in which they appear.



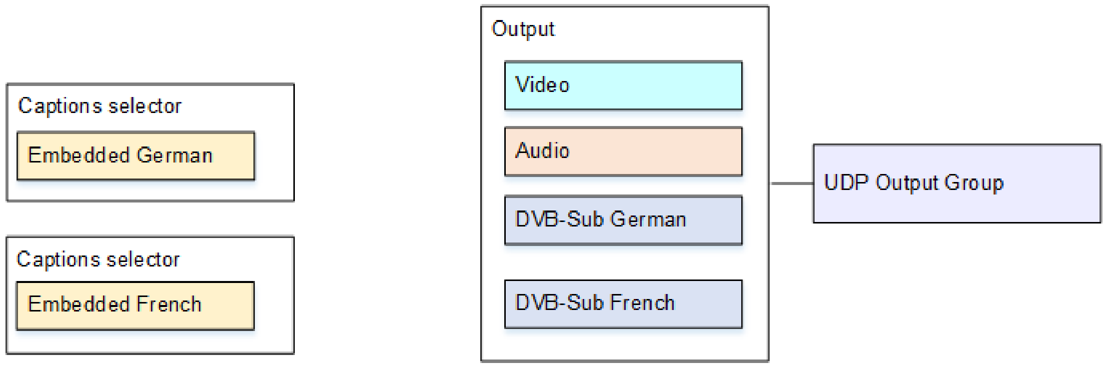
Event setup

To maintain the input format on output

1. On the web interface, on the **Event** screen, for **Input settings**, choose **Add captions selector** to create one captions selector. Set **Selector settings** to **Embedded source**.
2. In the **Output Groups** section, create an HLS output group.
3. Create one output and set up the video and audio.
4. In that same output, create one captions asset with the following:
   - **Captions selector name**: Captions selector 1.
   - **Captions settings**: One of the Embedded formats.
   - **Language code** and **Language description**: Leave blank; with embedded captions, all the languages are included.
5. In the HLS output group, in **Captions**, in **Captions language setting**, choose **Insert**.
6. For **HLS settings**, in **Captions language mappings**, choose **Add captions language mappings** twice (once for each language).
7. Complete the first group of mapping fields with **1**, **ENG**, and **`English`** and the second group with **2**, **FRE**, and **`French`**.
8. Finish setting up the event and save it.

Use case 2: One input format converted to one different output format

This example shows how to implement the second use case from the typical scenarios. The input includes two captions languages, and the single output will convert those captions. For example, the input has embedded captions in German and French. You want to produce a UDP output with both captions converted to DVB-Sub, plus one video and one audio.



Event setup

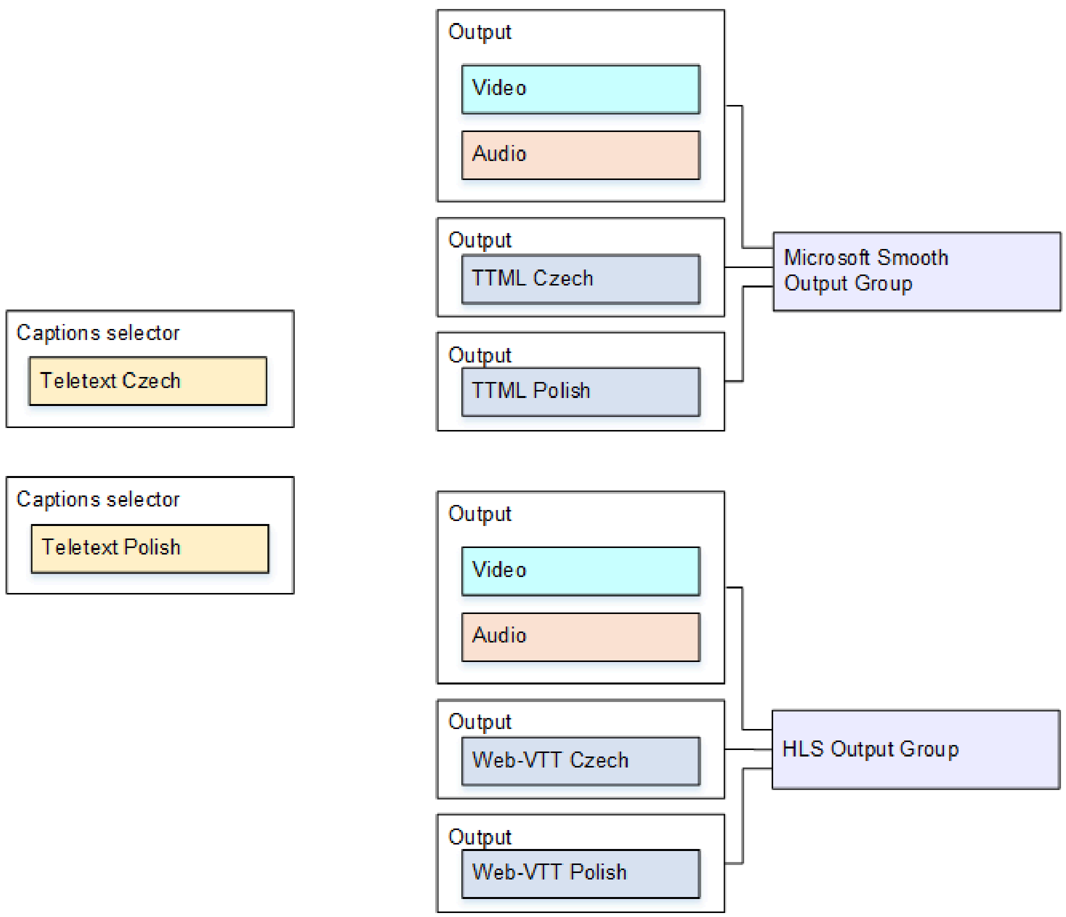
To convert the input format to another on output

1. On the web interface, on the **Event** screen, for **Input Settings**, choose **Add captions selector** twice, to create Captions selector 1 (for German) and Captions selector 2 (for French). In both cases, set **Selector settings** to **Embedded source**.
2. Create a UDP output group.
3. Create one output and set up the video and audio.
4. In this output, choose **Add captions** to create a captions encode.
   - **Captions selector name**: Captions selector 1.
   - **Captions settings**: DVB-Sub.
   - **Language code** and **Language description**: German.
   - Other fields: Keep the defaults or complete as desired.
5. Choose **Add captions** again to create another captions encode. Set up this encode for the French captions. Make sure that you set up the font fields for German and French in exactly the same way.
6. Finish setting up the event and save it.

Use case 3: One input format converted to different formats, one format for each output

This example shows how to implement the third use case from the typical scenarios. The input is set up with one format of captions and two or more languages. You want to produce several different types of output. In each output, you want to convert the captions to a different format but include all the languages.

For example, the input has teletext captions in Czech and Polish. Assume that you want to produce an MS Smooth output and an HLS output. Assume that in the MS Smooth output, you want to include one video and one audio and you want to convert the captions to TTML. In the HLS output, you want to include one video and one audio and you want to convert the captions to WebVTT.



Event setup

To convert the input format to a different format for each output

1. On the web interface, on the **Event** screen, for **Input Settings**, choose **Add captions selector** twice, to create the following captions selectors:

- Captions Selector 1 for teletext Czech. Specify the page that holds the Czech captions.
- Captions Selector 2 for teletext Polish. Specify the page that holds the Polish captions.

   Although you are including this captions in two different outputs (MS Smooth and HLS), you need to extract them from the input only once, so you need to create only one captions selector for each language.

2. Create a MS Smooth output group and configure it as follows:
   - Create one output and set up the video and audio.
   - Create a second output that contains one captions encode and no video or audio encodes and with the following settings:
     - **Captions selector name**: Captions Selector 1.
     - **Captions settings**: TTML.
     - **Language code** and **Language description**: Czech.
     - **Style control**: Set as desired.
   - Create a third output that contains one captions encode and no video or audio encodes, with the following settings:
     - **Captions selector name**: Captions Selector 2.
     - **Captions settings**: TTML.
     - **Language code** and **Language description**: Polish.
     - Other fields: same as the second output (the Czech captions).
3. Create an HLS output group and configure it as follows:
   - Create one output and set up the video and audio.
   - Create a second output that contains one captions encode and no video or audio encodes and with the following settings:
     - **Captions selector name**: Captions Selector 1.
     - **Captions settings**: WebVTT.
     - **Language code** and **Language description**: Czech.
     - Other fields: Set as desired.
   - Create a third captions output that contains one captions encode and no video or audio encodes and with the following settings:

- **Captions selector name**: Captions Selector 2.
- **Captions settings**: WebVTT
- **Language code** and **Language description**: Polish.
- Other fields: same as the second output (the Czech captions).

4. Finish setting up the event and save it.

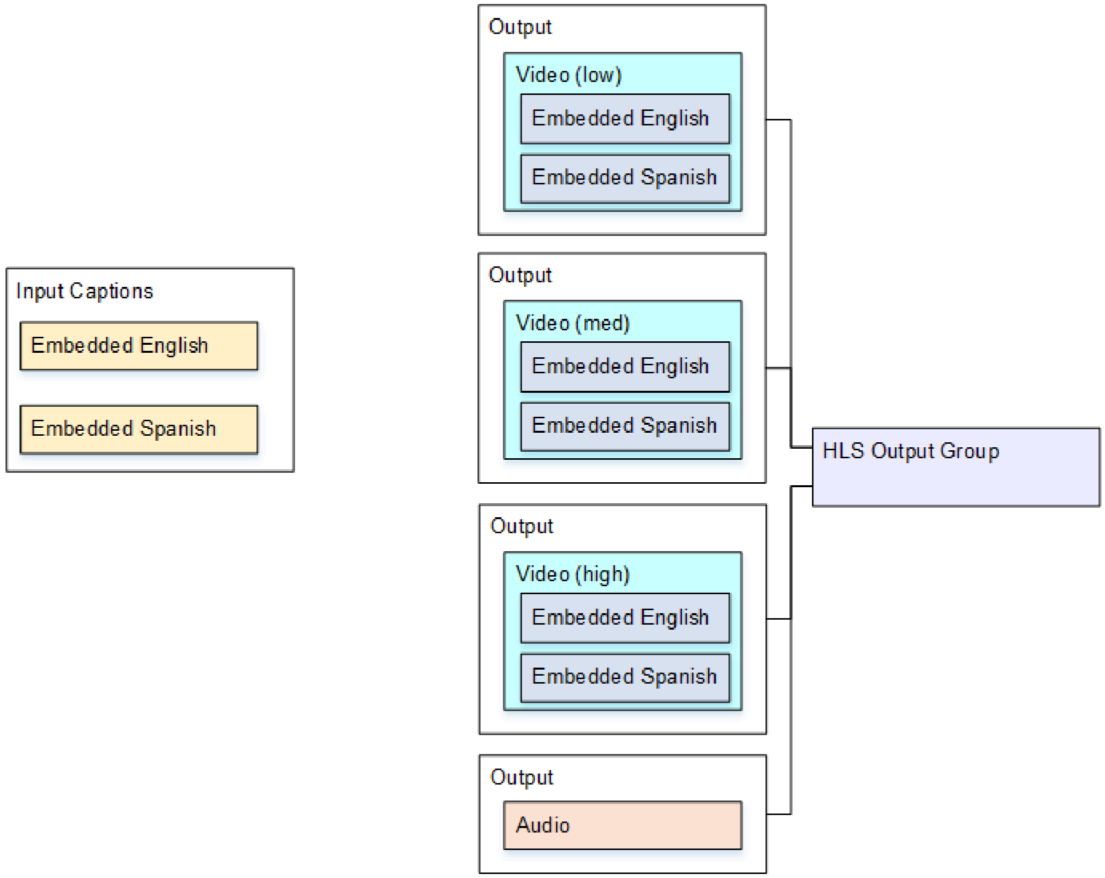
Use case 4: One captions output shared by multiple video encodes

This example shows how to set up captions in an ABR workflow. The first setup shows how to set up an ABR workflow when the captions are in the same output as the video, meaning that the captions are either embedded or captions style.

The second setup shows how to set up an ABR workflow when the captions belong to the sidecar category, in which case each captions encode is in its own output.

Event setup with embedded or object-style captions

This example shows how to implement the fourth use case from the typical scenarios. For example, you want to produce an HLS output with three video encodes (one for low-resolution video, one for medium, one for high) and one audio. You also want to include embedded captions (in English and Spanish) and associate them with all three video encodes.



To use one caption's output in multiple video encodes

1. On the web interface, on the **Event** screen, for **Input Settings**, choose **Add captions selector** to create one captions selector. Set **Selector settings** to **Embedded source**.
2. Create an HLS output group.
3. Create one output and set up the video and audio for low-resolution video.
4. In that same output, create one captions asset with the following:
   - **Captions selector name**: Captions Selector 1.
   - **Captions settings**: One of the Embedded formats.
   - **Language code** and **Language description**: Leave blank; with embedded passthrough captions, all the languages are included.
5. Create a second output and set up the video and audio for medium-resolution video.
6. In that same output, create one captions asset with the following:
   - **Captions selector name**: Captions Selector 1.

- **Captions settings**: One of the Embedded formats.
- **Language code** and **Language description**: Leave blank; with embedded captions, all the languages are included.

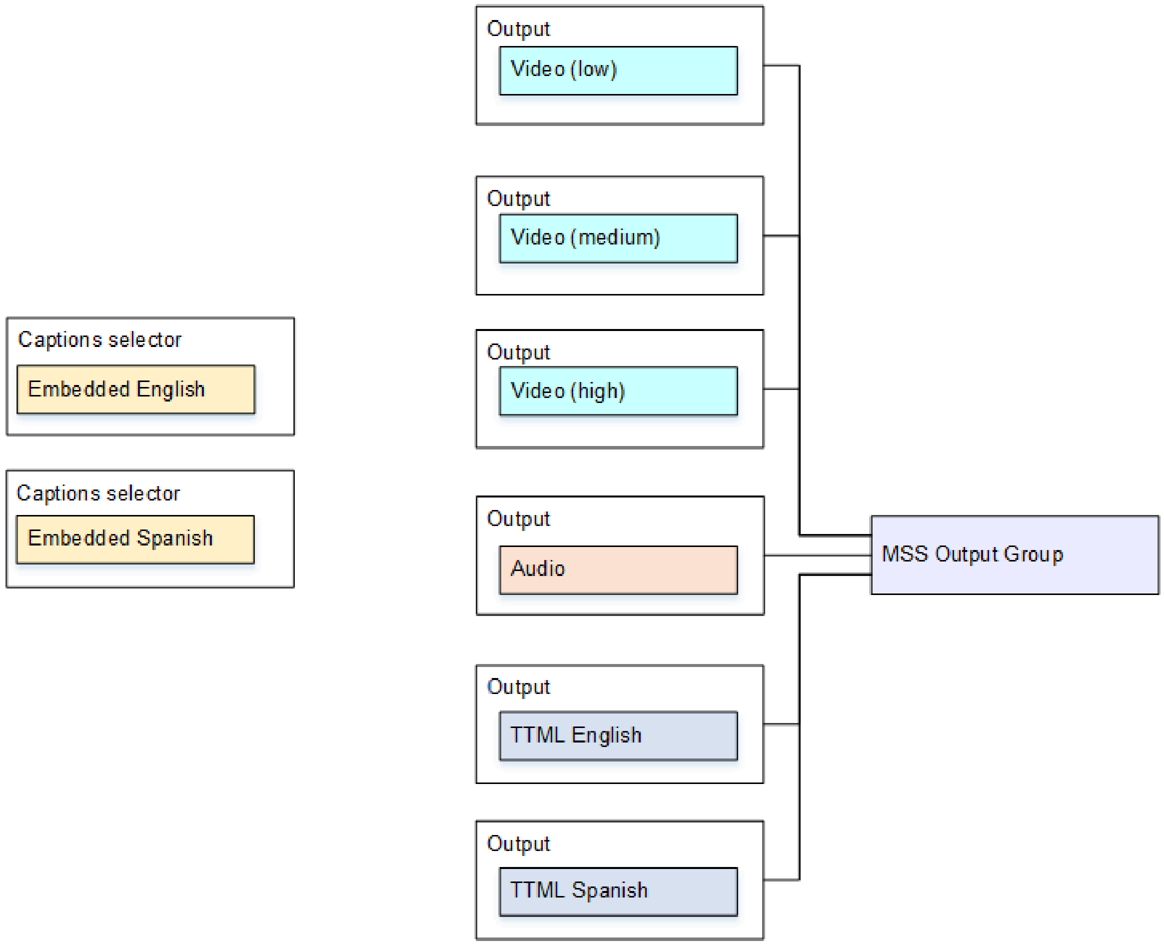
7. Create a third output and set up the video and audio for high-resolution video.
8. In that same output, create one captions asset with the following:

   - **Captions selector name**: Captions Selector 1.
   - **Captions settings**: One of the Embedded formats.
   - **Language code** and **Language description**: Leave blank; with embedded captions, all the languages are included.

9. Finish setting up the event and save it.

Event setup with sidecar captions

This example shows an ABR workflow where the captions are in sidecars. For example, you want to produce an MS Smooth output with three video encodes (one for low-resolution video, one for medium, one for high) and one audio. These encodes are in an MS Smooth output. You want to ingest embedded captions (in English and Spanish) and convert them to TTML captions, one for English and one for Spanish.



To set up sidecar captions

1. On the web interface, on the **Event** screen, for **Input Settings**, choose **Add captions selector** twice, to create the following captions selectors:
   - Captions selector 1: for Embedded English.
   - Captions Selector 2: for Embedded Spanish.
2. Create an MS Smooth output group.
3. Create one output that contains one video encode and set it up for low-resolution video.
4. Create a second output that contains one video encode and set it up for medium-resolution video.
5. Create a third output that contains one video encode and set it up for high-resolution video.
6. Create a fourth output that contains one audio encode and no video encode.
7. Create a fifth output that contains one captions encode and no video or audio encodes and with the following settings for the captions encode:

- **Captions selector name**: Captions Selector 1.
- **Captions settings**: TTML.
- **Language code** and **Language description**: English.

8. Create a sixth output that contains one captions encode and no video or audio encodes, and with the following settings for the captions encode:

   - **Captions selector name**: Captions Selector 2.
   - **Captions settings**: TTML.
   - **Language code** and **Language description**: Spanish.

9. Finish setting up the event and save it.

Passing through VBI data

Elemental Live supports passthrough of VBI data. You can pass through this data if the following statements are true:

- The input includes VBI data.
- You want to include all that data in the output. This data might include embedded captions.

To pass through VBI data

1. Create an output for the asset that is to include VBI data.
2. In the **Outputs** section, choose the **Settings** link for the output that contains the video asset.
3. Go to the **Stream** section. Display the **Video** fields. Click **Advanced**. More fields appear.
4. Check the **VBI Passthrough** field.

> **Important**
>
> Do not create a **Captions** object in this output.

All the VBI data (including embedded captions) from the input will be included in the output.

Reference: Languages supported with OCR captions

Following is a list of world languages that are supported in source captions when Elemental Live uses OCR translation to convert the source captions to the output format.

Afrikaans

Amharic

Arabic

Assamese

Azerbaijani

Azerbaijani - Cyrillic

Belarusian

Bengali

Tibetan

Bosnian

Bulgarian

Catalan; Valencian

Cebuano

Czech

Chinese - Simplified

Chinese - Simplified Vertical

Chinese - Traditional

Chinese - Traditional Vertical

Cherokee

Welsh

Danish

German

Dzongkha

Greek, Modern (1453-)

English

English, Middle (1100-1500)

Esperanto

Estonian

Basque

Persian

Finnish

French

German Fraktur

French, Middle (ca. 1400-1600)

Irish

Galician

Greek, Ancient (-1453)

Gujarati

Haitian; Haitian Creole

Hebrew

Hindi

Croatian

Hungarian

Inuktitut

Indonesian

Icelandic

Italian

Italian - Old

Javanese

Japanese

Japanese Vertical

Kannada

Georgian

Georgian - Old

Kazakh

Central Khmer

Kirghiz; Kyrgyz

Korean

Korean Vertical

Lao

Latin

Latvian

Lithuanian

Malayalam

Marathi

Macedonian

Maltese

Malay

Burmese

Nepali

Dutch; Flemish

Norwegian

Oriya

Panjabi; Punjabi

Polish

Portuguese

Pushto; Pashto

Romanian; Moldavian; Moldovan

Russian

Sanskrit

Sinhala; Sinhalese

Slovak

Slovenian

Spanish; Castilian

Spanish; Castilian - Old

Albanian

Serbian

Serbian - Latin

Swahili

Swedish

Syriac

Tamil

Telugu

Tajik

Thai

Tigrinya

Turkish

Uighur; Uyghur

Ukrainian

Urdu

Uzbek

Uzbek - Cyrillic

Vietnamese

Dynamic input switching

The dynamic input switching feature lets you use REST API commands to switch inputs in an AWS Elemental Live event. This method of input switching has two key features:

- The switching control (the REST API) is built into Elemental Live. You don't need an external server to control switching.
- You can modify the *playlist* of inputs without stopping the event. This ability to dynamically change the playlist is useful when you can't identify all of the inputs that you want to use in advance (before starting the event).

You can also switch inputs by using virtual input switching with an ESAM server. For more information, see the section called “Virtual input switching”.

> **Note**
>
> Use only one mechanism to control input switching. Don't give control of the input switching logic to both an ESAM server and a REST-based application. We strongly recommend against implementing a combination of these input switching features.

How dynamic input switching works

The dynamic input switching feature works as follows. When you create the event, you set up only the first input that you plan to process.

After you start the event, use the Elemental Live REST API to add more inputs to the event. You can create a playlist consisting of any number of live inputs, or file inputs, or both. This gives you a playlist consisting of the first input (which is currently being processed) and the inputs that you added using the REST API.

An input switch can occur in one of these ways:

- REST API-triggered switch. You switch to a different input by entering a REST API command to activate that input. This input can be any input in the playlist. You can set it up so that an input becomes active immediately or at a specific time.
- Automatic switch. If Elemental Live gets to the end of a file input that is not set up to loop, then Elemental Live switches to the next input in the list of inputs in the event.

  This method works well when you have a series of file inputs such as ad content. In this case, you let Elemental Live go through the inputs one by one. Before the series ends, enter a REST API command to activate a different input (perhaps a live input). This is to break out of the event order.

You can modify the playlist to add and remove inputs. If your event runs on a 24/7 basis, you will typically remove old inputs and add a fresh set of inputs on a regular basis.

Typical use cases

Use case 1

An event is designed to process live feed from a specific source. The input does not include ad markers.

You create a dynamic playlist in the event to process the live input. Then, at a given time, process one or more file inputs (the ad content). After that, switch back to the live input, either at a given time or when the last ad content has been processed. The dynamic playlist consists of the single live input “interrupted” periodically by ad content.

Use case 2

An event is designed to process live feed from a specific source. Periodically, the live feed should be replaced by file content (perhaps a movie). Then the same live feed should be resumed.

Use case 3

An event is designed to process live feed from a specific source. However, you may need to cut away from the live feed to process a file input. This cutaway file may be planned: For example, you may have a “broadcasting will resume shortly” message already prepared). Alternatively, the cutaway file may be something that is created on the spot to convey an unanticipated public announcement.

You create an event to process the live input. You then either create a dynamic playlist ahead of time to switch to a special file input, if necessary. Or at the last minute you create the entire dynamic playlist and its input, and then switch to it immediately.

Use case 4

An event is designed to process live feed from a specific source. Another event is designed to process live feed from a different source.

Before the dynamic playlist feature was introduced, you could only switch to the other live feed by stopping one event and starting the other. Now, you can merge these two events into one event, with each live feed in the dynamic playlist. You can set up the dynamic playlist to switch from one feed to another either at a scheduled time, or when the operator interrupts.

Use case 5

An event is designed to process file inputs, one after the other, without the event ending.

Before the dynamic playlist feature was introduced, you could add inputs to the event, but only one at a time. Now, you can add as many inputs as you want in one command.

In a variation of this use case, the event may contain only two files. The event can be set up so that as soon as one input ends, the next one starts over again, but with different content specified inside that input.

Use case 6

An event is designed to process file content (perhaps a movie). Periodically, the file content should be replaced by different file content that contains ad content. Then the original file content should be resumed.

You create a dynamic playlist in the event that interleaves sections of the movie with ad content, so: movie, ad content, movie, ad content, movie, and so on. All the inputs are file inputs.

Each time you add the movie file as an input, you include the input clipping tags to create a clip out of a different segment of the movie: 0 to 20 minutes, 20 to 25 minutes, 25 to 40 minutes, and so on. Each time that the movie resumes, it will resume at the desired point.

The procedures

Topics

General procedure

Topics

- Step A: Design the dynamic playlist
- Step B: Create and start the event
- Step C: Add more inputs
- Step D: Prepare inputs
- Step E: Activate inputs
- Step F: Continue adding to the dynamic playlist
- Step G: Interrupt the currently active input
- Step H: Clean up the playlist
- Step I: Troubleshoot

Step A: Design the dynamic playlist

Identify inputs

1. Identify the inputs and the order in which they are to play.

- The first input can be a live asset or a file asset.
- The inputs can be all live assets, all file assets, or a mix of live and file assets.
- The same input can be repeated as many times as you want in the dynamic playlist.

Generally, it is a good idea to add the inputs to the event in the order in which they will be played. You should do this even if you plan to specify a start time for some or all inputs (below).

Identify inputs to prepare manually

2. Identify inputs that must first be prepared:

- Live streaming inputs: You must manually prepare each live input before it is due to be played.
- File inputs: There is no need to manually prepare file inputs, but you may do so if you prefer. Perhaps a reason to prepare is that you know you have to prepare live inputs and you want to follow the same steps for both live and file inputs.

Identify inputs to activate manually

3. Identify inputs that must start at a specific time, rather than simply starting after the previous input ends.

If you follow the recommended procedure (Step E: below), you will need a placeholder input. Read the procedure in Step E: so that you can prepare this input now.

Step B: Create and start the event

4. Create the event with only the first input. (If the event contains more than one input, the others will be deleted when you create the dynamic playlist.)

Use the REST API Create Event command or use the web interface. See below for tips on setting up the event for the different use cases.

5. Start the event. Use the REST API or the web interface. Processing starts on that single input.

Step C: Add more inputs

6. Once the event has been started, you can create the dynamic playlist in order to add more inputs to the event.

There are two ways to add inputs to the dynamic playlist:

- Use the Add Dynamic Playlist Inputs command to create an array of inputs that are appended to the inputs currently in the event. Up to 29 inputs can be added in this way. For more information, see Add dynamic playlist inputs.
- Use the Replace Dynamic List command to create an array of inputs that replaces the existing dynamic playlist (except for the currently running input). You can add any number of inputs this way. For more information, see Replace dynamic playlist.

Step D: Prepare inputs

7. If you have identified inputs that must be manually prepared, then as soon as one of those inputs becomes “next in line”, you prepare it by calling Prepare dynamic playlist input.

- Time the preparation correctly: For a live input, you must prepare enough time in advance to allow Elemental Live to inspect the entire next-in-line input and decode at least one frame. But you should not prepare it too far in advance because once preparation starts, this is what happens:

  Elemental Live inspects the entire input then decodes the first frame in the pipeline. If this input is still not due to run, it discards that frame and decodes the frame that is now first in the pipeline. It discards that frame and decodes the frame that is now first in the pipeline, on and on until the input is due to run. This decoding obviously uses processing resources.

- This timing rule does not apply to file inputs.
- You can optionally specify a prepare time. The input will transition to the Prepare state but no preparing will actually occur until the specified time.
- Keep in mind that you can only manually prepare one input at a time: if you prepare input A, do not prepare the next input until input A is active. If you do, input A may not start as expected.

For more information about the rules around preparation, see Details on preparing inputs.

Step E: Activate inputs

8. For each input that must start at a specific time, set the activate time when the input is next in line.

Activating using placeholders

Scheduling inputs is a bit tricky because an Elemental Live event must always be encoding: there is no concept of waiting for a frame to arrive. Therefore, if you do not time input B to start immediately after input A ends, then Elemental Live will move to the next input that does not have an activate time (perhaps the input that appears after input B in the XML). But it is very difficult to get this timing between inputs exact enough.

The workaround is to use a “placeholder” input between inputs A and B, as follows:

- Estimate the expected end time of input A.
- Assign an activate time for input B that is as close to the end time without being before it. (In this way, input A is not interrupted.)
- Between input A and input B insert a “placeholder” file input (this input could be a blackout slate, for example). Do not assign a start to this placeholder but do set the loop_source tag to true for this input in order to play the content repeatedly until it is time to return to the live input.

This will be the result: if input A end time does not exactly match the activate time for input B, then the placeholder will play. When the activate time for input B arrives, the placeholder will be interrupted by input B. Input B will start at the scheduled time.

Step F: Continue adding to the dynamic playlist

9. As inputs are run, add more inputs to the event.

- So long as the event is still running, you can continue to add inputs to it; the event will not end.
- Add inputs using either the Add Dynamic Playlist Inputs command or the Replace Dynamic List command.

Step G: Interrupt the currently active input

Occasionally, you may need to interrupt the current active input and start a different input (input B). See Implementing use case 3 for an example of an unanticipated interruption.

- If necessary, add input B to the dynamic playlist. There are two ways to add:
  - Rebuild the entire playlist, with input B inserted at the top, using the Replace Dynamic Playlist command. The currently active input will not be removed when you do this. For more information, see Replace dynamic playlist.
  - Append input B to the end of the playlist, using the Add Dynamic Playlist Inputs command. The drawback to this method is that the input appears at the end of the playlist on the web interface, but then gets activated, which may confuse your operators. For more information, see Add dynamic playlist inputs.
- If input B is a live input, prepare it using the Prepare Dynamic Playlist command. For more information, see Prepare dynamic playlist input.
- Use the Activate Dynamic List Input command (without an activate time) to immediately activate the input. For more information, see Activate dynamic playlist input.

Step H: Clean up the playlist

10. The dynamic playlist is never automatically cleaned up; even after an input has been processed, it remains on the dynamic playlist.

- If you use the Replace Dynamic Playlist command, the entire playlist is replaced, which effectively cleans up old inputs. Fore more information, see Replace dynamic playlist.
- If you use the Add Dynamic Playlist Inputs command, then you may want to occasionally use the Delete Dynamic Playlist Input command. For more information, see Delete dynamic playlist input.

Step I: Troubleshoot

If the dynamic playlist is not behaving as expected, see the information about preparation and activation, starting here. You may have broken a preparation or activation rule.

Sample implementations

Implementing use case 1

Use case 1 is described here.

1. Create an event that has the live feed as the input.
2. Once the event starts, create a dynamic playlist that consists of the following:
   - Second input – ad content – from file.
   - Third input – live input (identical to first input).
   - Fourth input - ad content – from file.
   - And so on.
3. Immediately set an activate time for the second input. The first input will be interrupted by the second input at this time.
4. After the second input becomes Active, prepare the third input.

   When the second input ends, the third input will immediately become Active.
5. After the third input becomes Active, set an activate time for the fourth input. The third input will be interrupted by the fourth input at this time.

Implementing use case 2

Use case 2 is described here.

1. Create an event that has the live feed as the input.
   - Once the event starts, create a dynamic playlist that consists of the following:
     - Second input – ad content – from file.
     - Fourth input and others – file input, as required.
     - Finally, create an input to return to the live feed (same input source as the first input).
2. Immediately set an activate time for the second input in order to interrupt the live feed at the desired time.

The third input and others will play one after the other, one starting when the previous has completed.

3. When the last file input becomes Active:
   - Optionally set an activate time to return to the live feed. Or omit an activate time and let the live feed resume when the last file has completed.
   - Prepare the live input you are returning to.

Implementing use case 3

Use case 3 is described here.

1. Create an event that has the live feed as the input.
2. Once the event starts, create a dynamic playlist that consists of the following:
   - Second input – a file that displays the desired content. Include the loop_source tag for this input in order to play the content repeatedly until it is time to return to the live input.
   - Third input – live input (identical to the first input).
3. If an unanticipated event occurs, switch to the second input: either use the REST API (Activate Dynamic Playlist Input) or let the operator manually activate this input using the web interface control.
4. When you want to resume live input, prepare the third input and then switch to the third input.
5. If another unanticipated event occurs, you can switch again to the second input.

Implementing use case 4

Use case 4 is described here.

1. Create an event that has the live feed as the input.
2. Once the event starts, create a dynamic playlist that consists of the following:
   - Second input – a live input from a different live source.
3. Follow the desired action:
   - Optionally set an activate time to return to the live feed. Or omit an activate time and let the live feed resume when the last file has completed. Or omit the activate time and manually switch to the second input: either use the REST API (Activate Dynamic Playlist Input) or let the operator manually activate the second input using the web interface control.

- Prepare the live input you are returning to.

Implementing use case 5

Use case 5 is described here.

1. Create an event that has the first file as the input. In the event, set loop_all_inputs to true.
2. Once the event starts, create a dynamic playlist that consists of the following:
   - Second input – a file input.
3. Once the second input has become Active:
   - Modify the first input to point to a different file source. Change other tags as required (for example, the audio selectors).
   - Optionally set an activate time for the first input.

   When the second input has ended, the first input will become Active again.
4. Once the first input has become Active again:
   - Modify the second input to point to a different file source. Change other tags as required (for example, the audio selectors).
   - Optionally set an activate time for the second input.
5. Repeat as required.

Implementing use case 6

Use case 6 is described here.

1. Create an event that has the live feed as the input.
2. Once the event starts, create a dynamic playlist that consists of the following:
   - Second input – a file input such as a movie. Include the input_clipper tags to clip content. For example, clip it to run from the 0 mark to the 20 minute mark.
   - Third input – ad content – from file.
   - Fourth input – file input identical to the second input. Include the input_clipper tags to clip content, for example, to clip it to run from the 20 minute mark to the 35 minute mark.
   - Fifth input – ad content – from file.
   - Continue switching between the movie and ads.

- Finally, create a dynamic playlist to return to the live feed.

3. Let each input complete. The next input in the XML will automatically start.
4. When the last file input becomes Active:
   - Optionally set an activate time to return to the live feed. Or omit an activate time and let the live feed resume when the last file has completed.
   - Prepare the live input you are returning to.

Details on preparing inputs

Read this section if your inputs are not being prepared as expected.

File inputs vs live streaming inputs

- Preparing a live input involves a one-time inspection of the stream to determine what audio, video, and data tracks it holds and then continually decoding the frames. So: frame X gets decoded; the input does not become Active so frame X is discarded; frame Y gets decoded; the input does not become Active so frame Y is discarded; and so on decoding and discarding until finally the input does become Active.

  As you can see, this preparation may be expensive in terms of processing power.
- File inputs are automatically prepared just before they are Due-to-be-processed. Preparation of a file input involves decoding the first frame; after that, processing pauses until the input becomes Due-to-be-processed. Preparation of file inputs is not expensive.

Rules for preparing

- The input must not be currently Active.
- Only one input can be in the Prepared state at a time. If you manually prepare input A and then manually prepare input B before A becomes Active, input A will become unprepared.
- An input is put in the Prepared state only when the Prepare command is called on that input. So if a file input that is unprepared is Due-to-be-processed, it transitions directly to the Active state, where it gets both prepared and Active. This means that you can safely prepare a live input several inputs ahead of time, if you want.
- Timing of the manual prepare is important:
  - You should probably prepare a live input only when it is the Next-in-line.

  - You must prepare enough time in advance to allow Elemental Live to inspect the entire third input and decode at least one frame.
  - If the current input is quite long, you may want to wait until it is close to completion before preparing the Next-in-line input.
- If you prepare an input and specify a prepare time, note that the input will transition to the Prepare state but no preparing will occur until the specified time.
- If you both manually prepare an input with a prepare time and manually activate it with an activate time, make sure the prepare time is before the activate time; Elemental Live does not check.
- When you call Prepare Immediately on an input that has previously been set up to be prepared at a time that is still in the future, the input will prepare immediately and the future preparation will be cleared. It will not be prepared both immediately and in the future.

  For example, you prepare with a prepare time of 2:00 p.m., then you prepare it immediately, then the input becomes Active and completes at 1:45 p.m. The input will not get prepared again at 2:00 p.m.
- If you prepare input X with a prepare time, any previously prepared input will immediately become unprepared, even though the preparation of input x is still in the future. although input x is not being prepared, it is still considered to be in the Prepared state; only one input can be Prepared at a time, so the previous input becomes unprepared.

Details on activating inputs

Dynamic playlist order rules

When an input starts to be processed (it is “playing”), that input becomes “active”.

A file input typically moves through these stages:

- Idle and not prepared.
- Next-in-line and not prepared.
- Due-to-be-processed and not prepared.
- Active (prepared and processing).
- Idle (completed)

A live input typically moves through these stages:

- Idle and not prepared.
- Next-in-line and prepared.
- Due-to-be-processed and prepared.
- Active (processing).
- Idle (completed)

Idle

Any number of inputs can be idle.

The list of idle inputs includes inputs that have completed and those that have not been processed.

Next-in-Line

Only one input can be Next-in-line. This is the input that is either:

- The input that is after the currently active input, based on the order in which the inputs appear in the event XML and on the web interface.
- The input that has an activate time that exactly aligns with the end time of the current input.

Ideally, you should order the inputs in the XML so that an input that has an activate time also appears in its natural order. For example, it does not appear at the end of the list; this placement will cause confusion to your operators.

A next-in-line input may be prepared or unprepared.

Due-to-be-processed input

The input that is currently Next-in-line becomes Due-to-be-processed at the moment that the currently Active input completes.

This input then instantaneously transitions to Active.

Active input

This is the input that is being processed. An input becomes active as follows:

- If the input is a live input has been prepared, it will be activated and processing will start seamlessly.

- If the input is a live input has not yet been prepared, it will still be activated; it will not be skipped. Instead, it will first be prepared, then activated. There will be a noticeable delay while the input is prepared.
- If the input is a file input, it will be prepared and activated.

Rules for activating

The rules apply to activating an input.

- When you call Activate Immediately on an input, the currently Activated input will immediately transition to Idle and processing of that input will stop.
- When you call Activate Immediately on an input that has previously been set up to activate at a time that is still in the future, the input will activate immediately and the future activation will be cleared. It will not be activated both immediately and in the future.
- When you call Activate with Specified Time and then call Prepare, the activate time is not cleared: the input will be prepared (either immediately or later, depending on what you set up), then the input will be activated at the specified time.
- If you call Activate with Specified Time on an input that has not been prepared, the input will first be prepared (at the specified time), then it will be activated. There will be a noticeable delay in activation. In other words, activating with a specified time does not automatically set up any prepare schedule.
- If you both manually prepare an input with a prepare time and manually activate it with an activate time, make sure the prepare time is before the activate time; Elemental Live does not check.

Monitoring activity through the web interface

The operator can monitor dynamic playlist activity through the Elemental Live web interface.

1. On the web interface, display the Event Control tab.
2. If the blue button specifies Control Panel, then the Details panel is currently displayed. Click the Control Panel button.
3. On the Control Panel, click Input Controls (below the Preview panel) to expand that section. The dynamic playlist appears.

Status information

Input background	Icon in control column	State
Green	Spinner icon	Active
Green	Arrow icon	Prepared
Brown	Arrow icon	Being prepared
Gray	Arrow icon	Idle

The orange numbers down the left side are on-screen numbers, for display purposes only.

The numbers in the ID column are the REST IDs of the inputs.

Controls

The operator can click the triangle to switch to that input. The input will become Active. Processing will stop on the current Active input.

Monitoring activity through the API

Fine points and pitfalls

You can obtain a list of inputs currently in the dynamic playlist by using the Get event command or the Get event status command.

You can get information about the status of each input in the dynamic playlist by using the Get Event Status command. for more information, see Get event status.

Using the REST API

Topics

List of commands

Nickname	Action	Signature	Description
Add dynamic playlist inputs	POST	/live_events/<event ID>/inputs	In the specified event (which must be currently running), add the specified input or inputs to the end of the dynamic playlist.

Nickname	Action	Signature	Description
Replace dynamic playlist	POST	/live_events/<event ID>/playlist	In the specified event (which must be currently running), remove all non-Activ e inputs from the dynamic playlist and append the specified input or inputs. After this command, only the Active input remains from the original dynamic playlist.
Get event	GET	/live_events/<event ID>	Gets the contents of the event, including the list of inputs.
Modify dynamic playlist input	PUT	/live_events/<even t ID>/inputs/ <input ID>	In the specified event (which must be currently running), modify the specified dynamic playlist input (which must be non-Active).
	PUT	/live_events/<event ID>/inputs/ by_label/ <input_label>	In the specified event (which must be currently running), modify the specified input (which must be non-Active).

Nickname	Action	Signature	Description
Delete dynamic playlist input	DELETE	/live_events/<event ID>/inputs/ <input ID>	In the specified event (which must be currently running), delete the specified non-Active input from the dynamic playlist.
	DELETE	/live_events/<event ID>/inputs/ by_label/ <input_label>	In the specified event (which must be currently running), delete the specified non-Active input from the dynamic playlist.
Activate dynamic playlist input	POST	/live_events/<event ID>/ activate_input	In the specified event (which must be currently running), activate the specified dynamic playlist input either at the specified time or immediately.
Prepare dynamic playlist input	POST	/live_events/<event ID>/ prepare_input	In the specified event (which must be currently running), prepare the specified dynamic playlist input and optionally activate encoding at the specified time or immediately.

Nickname	Action	Signature	Description
Get status	GET	/live_events/<event ID>status	Gets the status of the specified event, including information about the stage and state of each input.

Add dynamic playlist inputs

In the specified event (which must be currently running), add the specified input or inputs (maximum 29 inputs for a total of 30, including the Active input) to the end of the dynamic playlist.

HTTP request and response

HTTP URL

```
POST http://<Live IP address>/live_events/<event ID>/inputs
```

Body of HTTP

XML content consisting of one:

- One inputs element that contains:
  - One or more input elements that each contains one or more of the regular input tags. See Elements and tags in an event input XML.

Tips for elements and tags

Here are some notes about tags of particular interest:

- input_label: It is recommended that you include the input_label tag. If you include this tag, you can reference the input later on using this label (rather than using the input ID, which you first have to query for using Get Event).

  An input label must be unique among all the inputs in the event when you add inputs – the existing inputs and the new inputs.

If you add an input with label X and later remove the input, you can add another input with label X, even if the input actually has different content. Elemental Live does not track the content, it just enforces the rule for label uniqueness at a given time.

- loop_source: This tag can be used to loop the dynamic playlist. If you set this tag to true for input X, it is a good idea to set an activate time for the next intended input, otherwise input X will process forever. See Activating using placeholders for a typical use case for loop_source.

  When this tag is true and Elemental Live is at the last input listed in the event XML, the first input in the dynamic playlist may be considered as the Next-in-line input.
- order: This tag is ignored in determining the dynamic playlist order.
- For the input source, include one of: network_input or device_input or router_input or file_input. Different inputs can have a different source.
- audio_selector, caption_selector: All inputs must have the same number of these elements: none or more than one. So if the first input has two audio_selector elements, all inputs must have two audio_selectors. This rule applies to the lifetime of the event: as soon as it starts, the count of audio_selector and caption_selectors is fixed and cannot vary.
- input_clipping: Can be used to process only a specific section of a live or file input.

Response

200 OK for a successful request.

Example

This request adds two inputs to the event with the ID 31. The first input is a live input, the second is a file input.

```
POST http://10.4.136.92/live_events/31/inputs
---------------------------------------------
<inputs>
  <input>
    <input_label>movie08E45_section_1</input_label>
    <loop_source>false</loop_source>
    <network_input>
      <quad>false</quad>
      <uri>udp://10.0.0.1:5005</uri>
    </network_input>
```

```
    <audio_selector>
      <default_selection>true</default_selection>
      <track>1</track>
    </audio_selector>
  </input>
  <input>
    <input_label>enigmatic_car_ad</input_label>
    <loop_source></loop_source>
    <file_input>
      <uri>/data/server/ad13978.mp4</uri>
    </file_input>
    <audio_selector>
      <default_selection>true</default_selection>
      <track>1</track>
    </audio_selector>
  </input>
</inputs>
```

Replace dynamic playlist

In the specified event (which must be currently running), remove all non-Active inputs from the dynamic playlist and append the specified file input or inputs. After this command, only the Active input remains from the original dynamic playlist; the rest of the dynamic playlist consists of new inputs.

The dynamic playlist can only consist of file inputs; if you want to add live inputs or live and file inputs, use Add Dynamic Playlist Inputs.

There is no maximum to the number of inputs that can be added using this command, and there is no maximum to the number inputs that result in the event. For example, you can replace the 15 inputs currently in the event with 900 new inputs. Compare this lack of restrictions to the restrictions that apply with Add Dynamic Playlist Inputs.

You can use Replace Dynamic Playlist to clear the dynamic playlist: create a Body consisting of an empty inputs element.

HTTP request and response

HTTP URL

```
POST http://<Live IP address>/live_events/<event ID>/playlist
```

Body of HTTP

XML content consisting of:

- One inputs element that contains:
  - One or more input elements that each contains one or more of the regular input tags. See Elements and tags in an event input XML.

Response

200 OK for a successful request.

Example

This request specifies two inputs to the event with the ID 31. The first input is a live input, the second is a file input. This list of inputs will replace the current dynamic playlist, not including the currently active input.

```
POST http://10.4.136.92/live_events/31/playlist
---------------------------------------------
<inputs>
  <input>
    <input_label>movie08E45_section_1</input_label>
    <loop_source>false</loop_source>
    <network_input>
      <quad>false</quad>
      <uri>udp://10.0.0.1:5005</uri>
    </network_input>
    <audio_selector>
      <default_selection>true</default_selection>
      <track>1</track>
    </audio_selector>
  </input>
  <input>
    <input_label>enigmatic_car_ad</input_label>
    <loop_source></loop_source>
    <file_input>
      <uri>/data/server/ad13978.mp4</uri>
    </file_input>
    <audio_selector>
      <default_selection>true</default_selection>
      <track>1</track>
```

```
    </audio_selector>
  </input>
</inputs>
```

Get event

Get a list of the inputs in the specified event.

There is no explicit command to get the dynamic playlist. But you can get the event in order to get information about the dynamic playlist.

Get Event is useful for obtaining the IDs of the inputs in the dynamic playlist and for parsing for the order in which they are listed in the XML.

Get Event includes the status of each input in these tags:

- active tag. Possible values are true and false.
- status tag. Possible values are preprocessing, pending, running, postprocessing, complete.
- active_input_id tag. Only the currently active input has this tag. For that input, the active tag and the input_ID specify the same value.

Get Event does not provide information about the prepare time or activate time on inputs on which you explicitly called Prepare with Specified Time or Activate with Specified Time. That information cannot be retrieved from Elemental Live; you must maintain the schedule outside of Elemental Live.

HTTP Request and Response

HTTP URL

```
GET http://<Live IP address>/live_events/<event ID>
```

Response

XML content consisting of one event element that contains:

- Various general tags.
- One or more input elements that each contain:
  - A unique ID tag.

- A unique input_label tag (optional).
- A status tag.
- An input element: complete network_input or device_input or router_input or file_input.
- A video_selector element.
- An optional audio_selector element.
- An optional caption_selector element.

- Other elements relating to input.
- Other elements relating to outputs.

Example

Request

This request gets the event with the ID 31.

```
GET http://10.4.136.92/live_events/31
```

Response

The event contains three inputs, with IDs 64, 98, 99, and with input_label tags "movie08E45_section_1", "enigmatic_car_ad" and "best_trowel_ad".

```
<?xml version="1.0" encoding="UTF-8"?>
<live_event; href="/live_events/31" product="Elemental Live" version="2.25.0.12345">
  <id>31</id>
  .
  .
  .
  <input>
    <active>false</active>
    <id>64</id>
    <input_label>movie08E45_section_1</input_label>
    <loop_source>false</loop_source>
    <status>pending</status>
    .
    .
    .
    <network_input>
      <id>4</id>
```

```
        .
        .
        .
        <uri>udp://10.0.0.1:5005</uri>
      </network_input>
      <video_selector>
        <id>2</id>
        .
        .
        .
      </video_selector>
      <audio_selector>
        <id>2</id>
        .
        .
        .
      </audio_selector>
      <input_info>
        .
        .
        .
      </input_info>
    </input>
    <input>
      <active>true</active>
      <id>98</id>
      <input_label>enigmatic_car_ad</input_label>
      <loop_source>false</loop_source>
      <status>pending</status>
      .
      .
      .
    </input>
    <input>
      <active>false</active>
      <id>99</id>
      <input_label>best_trowel_ad</input_label>
      <loop_source>false</loop_source>
      <status>pending</status>
      .
      .
      .
    </input>
    <active_input_id>98</active_input_id>
```

```
  <loop_all_inputs>true</loop_all_inputs>
  <status>running</status>
  .
  .
  .
</live_event>
```

Modify one dynamic playlist input

In the specified event (which must be currently running), modify the specified dynamic playlist input (which must be non-Active).

The input can be modified in any way, so long as it follows the rules for inputs in a dynamic playlist, as described in Tips for elements and tags.

HTTP Request and Response

HTTP URL

To specify the input by its REST ID:

```
PUT http://<Live IP address>/live_events/<event ID>/inputs/<input id>
```

To specify the input by the input label:

```
POST http://<Live IP address>/live_events/<event ID>/inputs/by_label/<input_label>
```

The input label is the value in the <input_label>; you may have included this tag when you first created the input. If you did not specify an input label, you cannot use this signature to modify.

Body of HTTP

XML content consisting of one input element that each contains only the tags to modify.

Response

200 OK for a successful request.

Example

Request

This request modifies the input with the ID 28 that is in the event with the ID 31. It modifies the input so to point to a different asset.

```
PUT http://10.4.136.92/live_events/31/inputs/28
-----------------------------------------------
<input>
  <network_input>
    <uri>udp://10.0.0.1:5005</uri>
  </network_input>
</input>
```

Delete dynamic playlist input

In the specified event (which must be currently running), delete the specified non-Active input from the dynamic playlist.

You can also delete the dynamic playlist using Replace Dynamic Playlist with an empty inputs in the Body. For more information, see Replace dynamic playlist.

HTTP Request and Response

HTTP URL

To specify the input by its REST ID:

```
DELETE http://<Live IP address>/live_events/<event ID>/inputs/<input ID>
```

To specify the input by the input label:

```
DELETE http://<Live IP address>/live_events/<event ID>/inputs/by_label/<input label>
```

The input label is the value in the <input_label>; you may have included this tag when you first created the input. If you did not specify an input label, you cannot use this signature to delete.

Response

200 OK for a successful request.

Example

Request

This request deletes the input with the label “curling_83399” that is in the event with the ID 31.

```
DELETE http://10.4.136.92/live_events/31/inputs/by_label/83399
--------------------------------------------------------------
```

Prepare dynamic playlist input

In the specified event (which must be currently running), manually prepare the specified dynamic playlist input either at the specified time or immediately.

The input being prepared must not be already Active.

HTTP Request and Response

HTTP URL

```
POST http://<Live IP address>/live_events/<event ID>/prepare_input
```

Body of HTTP

XML content consisting of one of the following:

- One input_id tag that contains the ID of the input to prepare.

Or

- One input_label tag that contains the input_label of the input to prepare.

Or

- One prepare_input element that contains:
  - One input_id tag that contains the ID of the input to prepare or one input_label tag that contains the input_label of the input to prepare.
  - One utc_time tag that contains the time at which to prepare the input, in UTC time, down to the seconds and (optionally) fractional seconds.

Response

The entire <input> element for the specified input.

Example 1

Request

In the event with the ID 31, prepare the input with the label “live_news_feed”.

```
POST http://10.4.136.92/live_events/31/prepare_input
----------------------------------------------------
<input_id>live_news_feed</input_id>
```

Response

The response returns the entire <input> element. In this example, this input has the ID 194.

```
<?xml version="1.0" encoding="UTF-8"?>
  <input>
    <active>true</active>
    <id>194</id>
    <input_label>live_news_feed</input_label>
    <loop_source>false</loop_source>
    <status>pending</status>
    .
    .
    .
    <network_input>
      <id>296</id>
      <uri>udp://10.255.10.41:5001</uri>
    </network_input>
    <video_selector>
      <id>2</id>
      .
      .
      .
    </video_selector>
    <audio_selector>
      <id>2</id>
      .
      .
      .
    </audio_selector>
```

```
    <input_info>
      .
      .
      .
    </input_info>
  </input>
```

Example 2

In the event with the ID 31, prepare the input with the ID 103 and activate the input at 2015123T235959.999:

```
<prepare_input>
  <input_id>103</input_id>
  <utc_time>2015123T235959.999</utc_time>
</prepare_input>
```

Activate dynamic playlist input

In the specified event (which must be currently running), activate the specified dynamic playlist input either at the specified time or immediately.

The input being prepared must not be already Active.

HTTP request and response

HTTP URL

```
POST http://<Live IP address>/live_events/<event ID>/activate_input
```

Body of HTTP

XML content consisting of one of the following:

- One input_id tag that contains the ID of the input to activate.

Or

- One input_label tag that contains the input_label of the input to activate.

Or

- One activate_input element that contains:
  - One input_id tag that contains the ID of the input to prepare or one input_label tag that contains the input_label of the input to prepare.
  - One utc_time tag that contains the time at which to activate the input, in UTC time, down to the seconds and (optionally) fractional seconds.

Response

The entire <input> element for the specified input.

Example 1

Request

In the event with the ID 31, activate the input with the input_label “syndicated_show_231”.

```
POST http://10.4.136.92/live_events/31/activate_input
-----------------------------------------------------
<input_label>syndicated_show_231</input_label>
```

Response

The response returns the entire <input> element. In this example, this input has the ID 64.

```
<?xml version="1.0" encoding="UTF-8"?>
  <input>
    <active>true</active>
    <id>64</id>
    <input_label>syndicated_show_231</input_label>
    <loop_source>false</loop_source>
    <status>pending</status>
    .
    .
    .
    <file_input>
      <id>206</id>
      <uri>/data/server/13978.mp4</uri>
    </file_input>
    <video_selector>
      <id>2</id>
      .
```

```
      .
      .
    </video_selector>
    <audio_selector>
      <id>2</id>
      .
      .
      .
    </audio_selector>
    <input_info>
      .
      .
      .
    </input_info>
  </input>
```

Example 2

In the event with the ID 31, activate the input with the ID 103 and activate the input at 2015123T235959.999:

```
<activate_input>
  <input_id>103</input_id>
  <utc_time>2015123T235959.999</utc_time>
</activate_input>
```

Get event status

Get the status of the inputs in the specified event.

There is no explicit command to get the status of the inputs in the dynamic playlist. But you can get the event status in order to get status information about the dynamic playlist.

Warning: Get Event Status only works with the JSON format. Therefore, the request header must include Accept: application/json and Content-type:application/json

HTTP request and response

HTTP URL

```
GET http://<Live IP address>/live_events/<event ID>/status
```

Response

JSON content consisting of one live_event element that contains:

- One outputs element and one output_groups element that each includes various tags.
- One active_input tag that contains the REST ID of the currently Active input.
- One inputs element that lists the inputs. The inputs appear in the order in which they were originally listed when the dynamic playlist was created. The following information appears for each input:

Tag	Value	Description
id	Integer	The unique REST ID for this input
state	String	The state of the input: • clear: The input is Activated or Prepared. • quarantined: The input is either starting or is recovering from a failure condition. • pending: The input is Idle • errored: A failure condition (as defined by the failure_condition tag in the event XML) has been triggered.
input_label	String	The input label, if one was created.
uri	String	The URI for a file input.

Stage and state

The state tag and active_input tag can provide some information about the stage and state of each input:

State tag	active_input tag	Stage and state
pending	Does not specify this input	The input is one of: • Idle and Unprepared • Next-in-line and Unprepared
clear	Does not specify this input	The input is one of: • Idle and Prepared • Active • Next-in-line and Prepared
clear	Specifies this input	The input is: • Active
quarantined	Specifies this input	The input is: • Active
errored	Specifies this input	A failure condition (as defined by the failure_condition tag in the event XML) has been triggered.

Note that you cannot determine whether the input is Next-in-line from the response to this Get. You must maintain that information outside of Elemental Live.

Example

```
GET http://10.4.136.92/live_events/47/status
{
```

```
  "live_event": {
    "id": "47",
    "status": "running",
    "outputs": [
      .
      .
      .
    ],
    "output_groups": [
      .
      .
      .
    ],
      .
      .
      .
    "active_input": "533761",
      .
      .
      .
    "inputs": [
      {
        "id": "29",
        "state": "pending",
        "input_label": "live_curling",
      },
      {
        "id": "30",
        "state": "clear",
      .
      .
      .

      },
      {
        "id": "30",
        "state": "pending",
        "input_label": null,
        "uri": "http://10.4.99.22/ads/best_trowel_ad.m3u8?_=0ap3000000580895"
      },
      {
        "id": "31",
        "state": "pending",
        "input_label": null,
```

```
        "uri": "http://10.4.99.22/ads/magic_scraper_ad/master.m3u8"
      },
      {
        "id": "32",
        "state": "pending",
        "input_label": null,
        "uri": "http://10.4.99.22/ads/enigmatic_car_ad/master.m3u8"
      },
      {
        "id": "33",
        "state": "pending",
        "input_label": null,
        "uri": "http://10.4.99.22/ads/chouette_toy_ad.m3u8"
      },
      .
      .
      .
    ],
    "alerts": [],
    "audio_level": "-7",
    "elapsed_time_in_words": "01:10:28"
  }
}
```


Elements and tags in an event input XML

This section lists all the elements and tags that could appear in the input element of an event XML. This section does not include exhaustive information about rules for if and when an element or tag can be included. For tips on some tags, see Add dynamic playlist inputs.

```
<href></href>
<version></version>
<product></product>
<input>
  <deblock_enable></deblock_enable>
  <deblock_strength></deblock_strength>
  <error_clear_time></error_clear_time>
  <failback_rule></failback_rule>
  <hot_backup_pair></hot_backup_pair>
  <input_label></input_label>
  <loop_source></loop_source>
  <name></name>
  <no_psi></no_psi>
```

```
  <order></order>
  <program_id></program_id>
  <service_name></service_name>
  <service_provider_name></service_provider_name>
  <image_inserter>
    <image_x></image_x>
    <image_y></image_y>
    <opacity></opacity>
    <image_inserter_input>
      <uri></uri>
      <username></username>
      <password></password>
      <interface></interface>
    </image_inserter_input>
  </image_inserter>
  <network_input>
    <enable_fec_rx></enable_fec_rx>
    <interface ></interface >
    <password></password>
    <quad></quad>
    <udp_igmp_source></udp_igmp_source>
    <uri></uri>
    <username></username>
  </network_input>
  <device_input>
    <channel></channel>
    <channel_type></channel_type>
    <device_id></device_id>
    <device_name></device_name>
    <device_number></device_number>
    <device_type></device_type>
    <input_format></input_format>
    <fec_settings>
      <udp_igmp_source></udp_igmp_source>
      <uri></uri>
    </fec_settings>
    <hdmi_settings>
      <input_format></input_format>
    </hdmi_settings>
    <sdi_settings>
      <input_format></input_format>
      <scte104_offset></scte104_offset>
    </sdi_settings>
  </device_input>
```

```
  <router_input>
    <input_number></input_number>
    <input_number_end></input_number_end>
    <quad></quad>
    <router_ip></router_ip>
    <router_type></router_type>
  </router_input>
  <file_input>
    <certificate_file></certificate_file>
    <interface></interface>
    <password></password>
    <uri></uri>
    <username></username>
  </file_input>
  <failover_condition>
    <description></description>
    <duration></duration>
    <order></order>
  </failover_condition>
  <video_selector>
    <color_space></color_space>
    <default_afd></default_afd>
    <name></name>
    <order></order>
    <pid></pid>
    <program_id></program_id>
  </video_selector>
  <audio_selector>
    <default_selection></default_selection>
    <external_audio_file_input></external_audio_file_input>
    <infer_external_filename></infer_external_filename>
    <language_code></language_code>
    <name></name>
    <offset></offset>
    <order></order>
    <pid></pid>
    <program_selection></program_selection>
    <strict_language_selection></strict_language_selection>
    <strict_pid_option></strict_pid_option>
    <track></track>
    <unwrap_smpte337></unwrap_smpte337>
  </audio_selector>
  <audio_selector_group>
    <audio_selector_name></audio_selector_name>
```

```
    <name></name>
  </audio_selector_group>
  <caption_selector>
    <order></order>
    <source_type></source_type>
    <language_code></language_code>
    <embedded_source_settings>
      <autodetect_scte20></autodetect_scte20>
      <source_608_channel_number> </source_608_channel_number>
      <source_608_track_number ></source_608_track_number>
      <upconvert_608_to_708></upconvert_608_to_708>
    </embedded_source_settings>
    <file_source_settings>
      <time_delta></time_delta>
      <upconvert_608_to_708></upconvert_608_to_708>
      <source_file>
        <certificate_file></certificate_file>
        <interface></interface>
        <password></password>
        <uri></uri>
        <username></username>
      </source_file>
    </file_source_settings>
    <teletext_source_settings>
      <page_number></page_number>
    </teletext_source_settings>
    <dvb_sub_source_settings>
      <pid></pid>
    </dvb_sub_source_settings>
    <scte27_source_settings>
      <pid></pid>
    </scte27_source_settings>
  </caption_selector>
  <input_clipping>
    <end_timecode></end_timecode>
    <order></order>
    <start_timecode></start_timecode>
  </input_clipping>
```

Broken down into individual components, you can see:

Function of code excerpt	Example code excerpt
Various tags	`    <href></href>` `<version></version>` `<product></product>` `<input>` `  <deblock_enable></deblock_enable>` `  <deblock_strength></deblock` `_strength>` `  <error_clear_time></error_c` `lear_time>` `  <failback_rule></failback_rule>` `  <hot_backup_pair></hot_back` `up_pair>` `  <input_label></input_label>` `  <loop_source></loop_source>` `  <name></name>` `  <no_psi></no_psi>` `  <order></order>` `  <program_id></program_id>` `  <service_name></service_name>` `  <service_provider_name></se` `rvice_provider_name>`
Image inserter for this input	`  <image_inserter>` `    <image_x></image_x>` `    <image_y></image_y>` `    <opacity></opacity>` `    <image_inserter_input>` `      <uri></uri>` `      <username></username>` `      <password></password>` `      <interface></interface>` `    </image_inserter_input>` `  </image_inserter>`
Input source	`  <network_input>` `    <enable_fec_rx></enable_fec_rx>` `    <interface ></interface >` `    <password></password>` `    <quad></quad>`

Function of code excerpt	Example code excerpt
	`` <udp_igmp_source></udp_igmp _source> <uri></uri> <username></username> </network_input> <device_input> <channel></channel> <channel_type></channel_type> <device_id></device_id> <device_name></device_name> <device_number></device_number> <device_type></device_type> <input_format></input_format> <fec_settings> <udp_igmp_source></udp_igmp _source> <uri></uri> </fec_settings> <hdmi_settings> <input_format></input_format> </hdmi_settings> <sdi_settings> <input_format></input_format> <scte104_offset></scte104_o ffset> </sdi_settings> </device_input> <router_input> <input_number></input_number> <input_number_end></input_n umber_end> <quad></quad> <router_ip></router_ip> <router_type></router_type> </router_input> <file_input> <certificate_file></certifi cate_file> <interface></interface> <password></password> <uri></uri> <username></username> ```

Function of code excerpt	Example code excerpt
	`</file_input>`
Failover	`<failover_condition>` `  <description></description>` `  <duration></duration>` `  <order></order>` `</failover_condition>`
Source video	`<video_selector>` `  <color_space></color_space>` `  <default_afd></default_afd>` `  <name></name>` `  <order></order>` `  <pid></pid>` `  <program_id></program_id>` `</video_selector>`

Function of code excerpt	Example code excerpt
Source audio	```<audio_selector> <default_selection></default_selection> <external_audio_file_input></external_audio_file_input> <infer_external_filename></infer_external_filename> <language_code></language_code> <name></name> <offset></offset> <order></order> <pid></pid> <program_selection></program_selection> <strict_language_selection></strict_language_selection> <strict_pid_option></strict_pid_option> <track></track> <unwrap_smpte337></unwrap_smpte337> </audio_selector> <audio_selector_group> <audio_selector_name></audio_selector_name> <name></name> </audio_selector_group>```

Function of code excerpt	Example code excerpt
Source captions	<pre><caption_selector> <order></order> <source_type></source_type> <language_code></language_code> <embedded_source_settings> <autodetect_scte20></autode tect_scte20> <source_608_channel_number> </ source_608_channel_number> <source_608_track_number ></ source_608_track_number> <upconvert_608_to_708></upc onvert_608_to_708> </embedded_source_settings> <file_source_settings> <time_delta></time_delta> <upconvert_608_to_708></upc onvert_608_to_708> <source_file> <certificate_file></certifi cate_file> <interface></interface> <password></password> <uri></uri> <username></username> </source_file> </file_source_settings> <teletext_source_settings> <page_number></page_number> </teletext_source_settings> <dvb_sub_source_settings> <pid></pid> </dvb_sub_source_settings> <scte27_source_settings> <pid></pid> </scte27_source_settings> </caption_selector></pre>

Function of code excerpt	Example code excerpt
Input clipping	`<input_clipping> <end_timecode></end_timecode> <order></order> <start_timecode></start_timecode> </input_clipping>`

Overview of graphic overlay solutions

You can insert a graphic onto the video in an Elemental Live event using the graphic overlay feature. There are two types of graphic overlay available — static and motion.

Topics

- Static overlay
- Motion graphic overlay
- Inserting both types of overlay

Static overlay

The static overlay feature lets you insert an image on the video at a specified time and to display it for a specified duration.

The static overlay can be a BMP, PNG, or TGA file.

You can insert up to eight overlays through the web interface and any number of overlays through the REST interface. The overlays are all independent of each other, which means that they can be set up to all appear on the underlying video at the same time (or not), and they can be set up to physically overlap each other (or not). This feature includes fade-in/fade-out capability and opacity.

For more information, see the section called "Graphic overlay: Static overlay".

Motion graphic overlay

The motion graphic overlay feature lets you insert one animated overlay on the underlying video at a specified time for a specified duration.

You set up one overlay for the event. When the event is running, you use the REST interface to modify the start time of the single overlay, and to change the content. In this way, you achieve the effect of several motion overlays played at different times in the event.

You can insert an animation in one of the following ways:

- HTML Files – You can use an HTML asset that you are continually publishing to a location outside of Elemental Live, and insert the asset into the video as an overlay. Use one of the following methods to control when the image appears:
  - You can let the authoring system control everything.
  - You can enter REST API commands to show or hide the image.
  - Your content provider can insert SCTE-35 messages to show and hide the image.
- MOV Files – Using a `.mov` file is a straightforward way to insert an animation onto the video of your event. You provide the file and set up your event to use it.
- Series of PNG Files – You can specify a series of `.png` files to be inserted one after the other to create an animation.

For more information, see the section called "Graphic overlay: Motion overlay".

Inserting both types of overlay

You can insert both static overlays and one motion overlay onto the underlying video.

Insert a motion graphic overlay in Elemental Live

This guide shows you how to insert a motion graphic overlay in AWS Elemental Live for video that it encodes. You can insert one asset into the event so that you can use it as a motion graphic overlay on all video outputs.

You can set up the event to use one of the following assets:

- An HTML5 asset

- A `.mov` file
- A set of ordered still `.png` files as assets for motion image overlays

The motion graphic overlay will always be inserted in all outputs in the event.

The motion overlay will be burned into the video after decoding and input-specific processing, and before encoding and creation of individual streams and outputs.

License requirement

To insert a motion overlay in any event, the software installed on the Elemental Live appliance must include the motion graphics license.

Topics

How to insert a motion overlay with HTML5

You can insert a motion overlay with HTML5. The overlay could be an animated overlay such as a sports scoreboard, a stock price ticker tape, or the current time and temperature. You use an HTML5 *authoring system* to create and publish an HTML asset. This system usually has an authoring component that lets you create the overlay content, and a control component that lets you control how and when the overlay is visible.

You configure the Elemental Live event with all the information about the motion overlay, including the location of the asset. After the event starts, Elemental Live pulls the asset that is being published, and includes the overlay in the video, when appropriate. Elemental Live doesn't provide any features for manipulating the content or position of the overlay.

Topics

Step A: Choose the method for show/hide

You must speak to the operator of the authoring system and decide how the overlay will be controlled. There are several options.

Authoring system control

The authoring system might set up the asset so that it controls how and when the motion overlay appears. In the output video, the overlay is always present, but sometimes it contains no content. For example, the authoring system might set up so that when the motion overlay shouldn't appear on the video, the overlay is present but contains no content (it is transparent).

Your role is to set up in Elemental Live so that the event shows the overlay as soon as the event starts. You never hide the overlay. You (the Elemental Live operator) don't have to coordinate with the operator of the authoring system.

It is the job of the authoring system to make sure that the asset shows content sometimes and appears as transparent (no content) at other times.

REST API control

The authoring system might set up the asset with the expectation that you (the Elemental Live operator) will show and hide the overlay at the appropriate times.

Elemental Live continually pulls the asset from the authoring system.

Your role is to enter REST API commands to show and hide the overlay. You (the Elemental Live operator) and the operator of the authoring system must coordinate the schedule for showing overlays.

SCTE 35 control

The authoring system might set up the asset with the expectation that the source content includes specific SCTE 35 messages that Elemental Live reads to know when to show or hide the overlay.

You have no role in showing or hiding the overlay.

Instead, the overlay is controlled as follows. Elemental Live is continually pulls the asset from the authoring system. In addition, Elemental Live automatically shows or hides the overlay when it encounters the SCTE 35 messages.

To use this method, the content must already contain the appropriate SCTE 35 messages. You can't use Elemental Live to insert these messages.

The authoring system, the content provider (who must insert the SCTE 35 messages in the content), and you (the Elemental Live operator) must coordinate the SCTE 35 messages and the schedule.

Elemental Live reacts to well-formed SCTE 35 messages of type `time_signal`. The message must include the following tags:

- `segmentation_type_id`: Provider Overlay Placement Opportunity Start (0x38) or Provider Overlay Placement Opportunity End (type 0x39).
- `segmentation_duration_flag`: true or false
- `segmentation_duration`: Must be included if the `segmentation_duration_flag` is true.

Step B: Prepare the HTML5 asset

You use an authoring system to create the asset and to manage the content, including implementation of features such as fade or opacity.

1. Choose an authoring system and create the asset – Use the authoring system to create the asset. The HTML5 content must meet these requirements:

   - It can be any HTML5 authoring system that uses standard browser-based rendering techniques.
   - It can use any HTML5 tags except video and audio.
   - It can incorporate javascript that interacts with a backend system to dynamically control the asset that is being published to the source URL. You should size the content to be the same size or smaller than the width and height of the largest video rendition in your channel. You can also use responsive HTML (HTML that resizes automatically to different frame sizes).

   See the list after this procedure for more guidelines for preparing the asset.
2. Make a note of the URL of the asset. This URL must be accessible to Elemental Live.
3. If the location of the motion graphics asset requires login in order for Elemental Live to download the asset, make a note of the user name and password.

Supported authoring features

The asset can include features that are supported in Chrome version 84.0.4147.125. If you include features that are supported only in a later version, the asset might not render properly in Elemental Live.

CPU requirements

An HTML5 asset increases CPU utilization for each enabled event by *up to* 20%, depending on complexity of the asset. For example, if your event currently uses 30% of CPU (without HTML5 motion graphics), it might use approximately 36% with HTML5 motion graphics.

Resolution

Elemental Live renders the asset to match full-frame for the resolution of the first video input. If the input resolution changes during the event, Elemental Live will continue to render at the initial resolution, but will scale up or down to match the asset's new resolution.

We recommend that you set up the asset to have the same pixel ratio as the video. The resolution of the asset can change through the life of the event, but its ratio shouldn't change.

Color space

If you set up the event to convert the color space of the video, Elemental Live will convert the asset in the same way. For example, it will convert the color space to HDR10. To perform this conversion, Elemental Live will assume that the asset color space is SDR.

Step C: Set up the event

You configure the event with all the information about the motion overlay. After you start the event, you won't be able to change any information about the motion overlay. You can only show or hide it.

You can configure the information using the web interface or the REST API.

Using the web interface

To configure the event using the web interface

1. Obtain the asset URL and user credentials (if required) from the administrator of the authoring system that is publishing the HTML5 asset.

2. In the **Global Processors** section, go to the **Image Inserter** field and choose **On**. More fields appear.
3. Complete these fields:
    - **Insertion Mode**: Choose **HTML**.
    - **Input**: Enter the location of the HTML5 asset.

      If access to your local or mounted directory requires authentication, enter the user name and password.
4. Set the following fields to match the control that you're using.

Option for control	Value for Active	Value for <enable_rest>	Value for <enable_scte35>
Authoring system control	Unchecked	Unchecked	Unchecked
REST API control	Checked or unchecked, depending on whether you want the motion overlay to show when the event starts	Checked	Unchecked
SCTE 35 control	Unchecked	Unchecked	Checked

For detailed information about the fields, see the section called “Fields for an HTML5 asset”.

Using the REST API

This description assumes that you are familiar with using the REST API and with the XML body for a `live_event`.

To configure the event using the REST API

1. Enter a POST or PUT command in the usual way. Use POST to create a new event. Use PUT to modify an existing event:

```
POST http://<Live IP Address>/live_events
```

```
PUT http://<Live IP Address>/live_events/<live event id>
```

2. In the body of the request, include one `motion_image_inserter` element inside the `live_event` tag. Complete these tags:

   - `insertion_mode`: Set to HTML.
   - `motion_image_inserter_input`: Enter the location and file name of the HTML5 asset.

     If access to your local or mounted directory requires authentication, enter the user name and password.

3. Set the following tags to match the control you are using.

Option for control	Value for <active>	Value for <enable_rest>	Value for <enable_scte35>
Authoring system control	false	false	false
REST API control	true or false, depending on whether you want the motion overlay to show when the event starts	true	false
SCTE 35 control	false	false	true

For detailed information about the tags, see the section called “Fields for an HTML5 asset”.

```
<motion_image_inserter>
    <active>
    <enable_rest>
    <enable_scte35>
    <insertion_mode>
    <motion_image_inserter_input>
```

```
        <uri>
        <password>
        <username>
      </motion_image_inserter_input>
</motion_image_inserter>
```

4. The response repeats back the data that you posted with `<ID>` tags for many elements, including IDs for the following motion image inserter elements:

   - `motion_image_inserter`
   - `motion_image_inserter_input`

Example

The following request creates an event with the name myLiveEvent and turns on the REST API control option.

```
POST http://198.51.100.22/live_events
-----------------------------------
  <?xml version="1.0" encoding="UTF-8"?>
      <live_event>
          <name>myLiveEvent</name>
          . . .
          <motion_image_inserter>
              <active>true</active>
              <enable_rest>true</enable_rest>
              <insertion_mode>html</insertion_mode>
              <motion_image_inserter_input>
                  <uri>http://myAuthorSystem/output/output?aspect=16.9</uri>
              </motion_image_inserter_input>
          </motion_image_inserter>
      </live_event>
```

Fields for an HTML5 asset

Field on web interface	Tag in the XML	Type	Description
Insertion Mode	<insertion_mode>	String	Choose HTML.
Input	<uri>	String	

Field on web interface	Tag in the XML	Type	Description
Username Password	<username> <password>	String	
Active	<active>	Boolean	Always select this box.
Enable REST Control	<enable_rest>	Boolean	Select this field only if you chose to use the REST API to control the asset.
Enable SCTE 35 Control	<enable_scte35>	Boolean	Select this field only if you chose to use SCTE 35 messages in the source content to control the asset.

Step D: Showing and hiding the motion overlay

If you have configured the motion overlay for REST control, start the event , then enter REST API commands to show and hide the motion overlay.

If you have configured the motion overlay for authoring system control or SCTE 35 control, Elemental Live automatically shows and hides the motion overlay. You don't have to intervene. You can stop reading this section.

To show or hide the motion overlay on a running event

1. Enter a POST command as follows:

```
POST http://<Live IP Address>/live_events/<live event id>/motion_image_inserter
```

2. In the body of the request, include one `motion_image_inserter` element that contains only the `active` tag. Set the `active` to true to show the motion overlay, or set it to `false` to hide the motion overlay. See the examples after this procedure.

3. The response repeats back the data that you posted with `<ID>` tags for `motion_image_inserter` and `motion_image_inserter_input`.

Example 1

The following example shows the motion overlay immediately.

```
POST http://198.51.100.22/live_events/33/motion_image_inserter
---------------------------------------------------------------
<motion_image_inserter>
    <active>true</active>
</motion_image_inserter>
```

Example 2

The following example hides the motion overlay immediately.

```
POST http://198.51.100.22/live_events/33/motion_image_inserter
---------------------------------------------------------------
<motion_image_inserter>
    <active>false</active>
</motion_image_inserter>
```

How to insert a motion overlay with QuickTime MOV

You can use a `.mov` file as an asset for the motion image overlay.

Here are some examples of how motion image overlay works with a MOV file:

- Example 1: “Coming up” motion overlay – You want to insert an asset that will run as a 10-second motion overlay 59 minutes into the runtime of the event. You want the motion overlay to be placed in the lower right corner of the video frame.
- Example 2: Animated corporate logo – You want to insert an asset and run it every 5 minutes, starting 20 minutes into the runtime of the event. You specify the single asset in the event and specify the first runtime. You start the event and, after the motion overlay has run once, you send a REST API command to modify the start time of the asset to a new time. You repeat this command (each time with a new start time) as many times as you want.

Typically, you set up the event as follows:

- You prepare a MOV file and store it at a location that is accessible to Elemental Live.
- You configure the event with the location URL and start time of the first motion overlay you want to run, with position and size information, and with an instruction to run the motion overlay once or to loop it. You set up the motion overlay to be active when the event starts, and you enable control via the REST API.
- When you're ready, you start the event. The motion overlay configured in the event runs at the specified time. You then enter REST API commands to specify different content (a different URL), a start time, and a new position and size. After that motion overlay plays, you can enter another command to run different content, as required, for the duration of the event.

Step A: Prepare the MOV asset

1. Create a file – Use a third-party process to create a MOV file encoded with Apple QuickTime Run Length Encoding. Other codecs in the MOV container are not supported. Take care with the following settings:

   - Aspect ratio: The motion overlay can have any aspect ratio. It does not have to match the aspect ratio of the video output.
   - Frame rate: The frame rate of the motion overlay asset must match the frame rate of the underlying video.
   - Size: The motion overlay can be any size, in pixels, up to the size of the underlying video. The motion overlay must be prepared in the desired size; there is no way to resize it when setting it up in the event.
   - Position: The motion overlay cannot be positioned so that part of the motion overlay runs beyond the right edge or bottom edge of the underlying video.
     - If you set up a motion overlay so that it is too big or it overruns and Elemental Live can identify this error at event creation time, then an error message will appear.
     - If Elemental Live cannot identify the error in advance, an error message will appear while the event is running. The event will not stop but the insertion request will fail.
2. Place the file – Place the MOV file in a location accessible to the Elemental Live: on a local directory, on a remote filesystem accessible via mount point, or in an Amazon S3 bucket.

   You can specify the location in one of the following ways:

   - Local to the Elemental Live appliance. For example, `/data/assets/motion overlay.mov`

- A remote server via a mount point. For example, `/data/mnt/assets/motion overlay.mov`
- An Amazon S3 bucket, using SSL. For example, `Amazon S3ssl://company.test/DOC-EXAMPLE-BUCKET/motion overlay.mov`
- An Amazon S3 bucket, without SSL. For example, `S3://company.test/DOC-EXAMPLE-BUCKET/motion overlay.mov`

Step B: Set up the event

You configure the event with information about the first motion overlay. You can configure this information when creating an event. Or you can change the existing motion overlay configuration on a non-running event. You can configure the information using the web interface or the REST API.

Topics

- Using the web interface
- Using the REST API
- Fields for a MOV asset

Using the web interface

To configure the event using the web interface

1. In the **Global Processors** section, go to the **Image Inserter** field and choose **On**. More fields appear.
2. Complete the fields. You can configure some or all the information in the non-running event, but you must at least set the following fields:
   - **Insertion Mode**
   - **Enable REST Control**

   You must also configure the motion overlay behavior when the event first starts:

   - If you want the motion overlay to appear as soon as the event starts, specify all the fields. Make sure that you check **Active**, and make sure that you leave the **Action Time** empty.

- If you don't want the motion overlay to appear as soon as the event starts, leave **Active** unchecked.

You can configure the remaining fields after you've started the event, when you want to run the first motion overlay.

For detailed information about the fields, see the section called “Fields for a MOV asset”.

Using the REST API

This description assumes that you are familiar with using the REST API and with the XML body for a `live_event`.

To configure the event using the REST API

1. Enter a POST or PUT command in the usual way. Use POST to create a new event. Use PUT to modify an existing event:

```
POST http://<Live IP Address>/live_events
```

```
PUT http://<Live IP Address>/live_events/<live event id>
```

2. In the body of the request, include one `motion_image_inserter` element inside the `live_event` tag. The XML is structured as follows.

   You can configure some or all the information in the non-running event, but you must at least set the following tags:

   - `insertion_mode`
   - `enable_rest`

   Make sure that you configure for the desired motion overlay behavior when the event first starts:

   - If you want the motion overlay to appear as soon as the event starts, specify all the tags. Make sure that you set `active` to `true`, and make sure that you leave `action_time` empty.

- If you don't want the motion overlay to appear as soon as the event starts, set `active` to `false`.

You can configure the remaining tags after you've started the event, when you want to run the first motion overlay.

For detailed information about the tags, see the section called “Fields for a MOV asset”.

```
<motion_image_inserter>
    <action_time>
    <active>
    <enable_rest>
    <full_frame>
    <image_x>
    <image_y>
    <insertion_mode>
    <loop_input>
    <motion_image_inserter_input>
      <uri>
      <password>
      <username>
    </motion_image_inserter_input>
</motion_image_inserter>
```

3. The response repeats back the data that you posted with `<ID>` tags for many elements including IDs for the following motion image inserter elements:

   - `motion_image_inserter`
   - `motion_image_inserter_input`

Example

The following request creates an event with the name myLiveEvent. The event includes a `motion_image_inserter` section that inserts the motion overlay at a specific time:

```
POST http://198.51.100.22/live_events
-----------------------------------
  <?xml version="1.0" encoding="UTF-8"?>
      <live_event>
          <name>myLiveEvent</name>
```

```
        . . .
        <motion_image_inserter>
            <action_time>10:16:23:10</action_time>
            <active>true</active>
            <enable_rest>true</enable_rest>
            <full_frame>false</full_frame>
            <left>100</left>
            <top>200</top>
            <insertion_mode>mov</insertion_mode>
            <loop_input>true</loop_input>
            <motion_image_inserter_input>
                <uri>/data/logo001.mov</uri>
            </motion_image_inserter_input>
        </motion_image_inserter>
    </live_event>
```

Fields for a MOV asset

Field on web interface	Tag in the XML	Type	Description
Insertion Mode	<insertion_mode>	String	Choose MOV.
Input	<uri>	String	The location and file name of the MOV file. For Amazon S3, use `sse=true` to turn on Amazon S3 Server Side Encryption (SSE) and `rrs=true` to turn on Reduced Redundancy Storage (RRS). Default values for RRS and SSE are `false`.
Username Password	<username> <password>		If access to your local or mounted directory requires a user name

Field on web interface	Tag in the XML	Type	Description
			and password, click the lock icon next to the **Input** field to show the **Username** and **Password** fields. For Amazon S3, enter the access key ID in the user name field. Enter the secret access key in the password field.
Left	<image_x>	Integer	Placement of the left edge of the motion overlay relative to the left edge of the video frame, in pixels. 0 is the left edge of the frame. Take note of the width of the motion overlay and make sure that the position of the left edge of the motion overlay does not cause the right edge to be cropped.

Field on web interface	Tag in the XML	Type	Description
Top	<image_y>	Integer	Placement of the top edge of the motion overlay relative to the top edge of the video frame, in pixels. 0 is the top edge of the frame. Default 0. Take note of the height of the motion overlay and make sure that the position of the top edge of the motion overlay does not cause the bottom edge to be cropped.
ActionTime	<action_time>	String	The start time for the motion overlay. Specify the start time in one of the formats discussed in detail below this table.

Field on web interface	Tag in the XML	Type	Description
Loop Input	<loop_input>	Boolean	• Select to loop the motion overlay indefinitely. The motion overlay will run until the event ends. To stop the motion overlay earlier, see the section called "Step C: Manage at runtime". • Clear the check box to run the motion overlay only once.

Field on web interface	Tag in the XML	Type	Description
Full Frame	<full_frame>	Boolean	Expand the motion overlay to fit the video frame. In this case, make sure Left and Top are set to 0. If this field is selected and the motion overlay has a different aspect ratio to the underlying video, the motion overlay will be scaled until one of the following applies: • The motion overlay fits in the length. The motion overlay will then be positioned with equal space on the left and right. • The motion overlay fits in the width. The motion overlay will then be positioned with equal space above and below. Note that the **Stretch to output** field in the **Stream** section does

Field on web interface	Tag in the XML	Type	Description
			not affect the motion overlay; even if the video is stretched, the motion overlay is not stretched.
Active	<active>	Boolean	Always select this box when initially setting up the motion overlay. After the initial setup, the value of this tag can be changed via the REST API to manage the content and behavior of the motion overlay.
Enable REST Control	<enable_rest>	Boolean	Check this field only if you plan to manage motion overlays via the REST API, after this initial setup via the web interface . Typically, you will want this tag to be `true`.

Action time formats

Option 1: Timecode format (HH:MM:SS:FF).

The value to enter for HH:MM:SS:FF depends on the method used to calculate the *output timecode*. This method is specified in the **Timecode Config > Source** field. Identify the Source method set for your event, then set the action time to match the timecode of the frame where you want the action to occur.

- If **Source** is **Embedded**: The output timecode is extracted from the timecode that is carried with the input media. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output. For example, 10:24:25:20, 10:24:25:21, and so on.
- If **Source** is **Start at 0**: The output timecode for the first frame is 00:00:00:00 and then the output timecode counts up with each successive frame in the entire output. For example, 00:00:00;01: 00;00:00:02, and so on.
- If **Source** is **System Clock** or **Local System Clock**: The output timecode for the first frame is the system time at which the frame is decoded. Then the output timecode counts up with each successive frame in the entire output. For example, if the first frame is decoded at 09:45:51, then that timecode is 09:45:51:01. The timecode for the next frame is 09:45:51:02, and so on.
- If **Source** is **Specified Start**: The output timecode for the first frame is the time you specified when you selected this option as the timecode source. Then the output timecode counts up with each successive frame in the entire output. For example, if you set the time to 15:30;00, then the timecode for the first frame is 15;30:00:01, and so on.
- If **Source** is **External Reference Connector**: The timecode is extracted from external LTC source. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output. For example, the timecode for the first frame is 20:03:19:01, then 20:03:19:01, and so on.

Option 2: ISO 8601 UTC time with no dashes or colons. For example, 20160102T030405.678Z. In this case, the start time for every motion overlay will be the UTC time.

Option 3: You can only use this option while adding or modifying a motion overlay in a running event. Set the `action_time` tag to an empty string to set the start time to “now”. With this option, the start time is never exact. You cannot use this option when creating an event or modifying a non-running event.

Step C: Manage the motion overlay on a running event

After the event starts, the motion overlay that is configured in the event runs at the specified time. When the event is running, you can work with the motion overlay only via the REST API. You can enter REST API commands to make the following changes:

- A different URL, to specify different content.
- A start time, the new position and size, and the specified loop behavior.

> **Note**
>
> Commands sent during an event change the event XML. Therefore, if you export the XML and create a new event with it, the new event will have any motion overlays set up as they were set during the course of the event, not as they were when the event was created.

To modify the motion overlay on a running event

1. Decide which fields you want to modify. For information about the types of changes that you can make, see the table later in this section.
2. Enter a POST command to modify any of the image overlay parameters in the running event:

```
POST http://<Live IP Address>/live_events/<live event id>/motion_image_inserter
```

3. In the body of the request, include one `motion_image_inserter` element that contains the required tags. For more information about the tags, see the section called “Using the REST API”.
4. The response repeats back the data that you posted with `<ID>` tags for `motion_image_inserter` and `motion_image_inserter_input`.

Example 1

The following example request modifies the running event with the ID 33. It sets up the currently defined motion overlay MOV to run again at 20160102T030405.678Z.

```
POST http://198.51.100.22/live_events/33/motion_image_inserter
-------------------------------------------------------------
<motion_image_inserter>
    <action_time>20160102T030405.678Z</action_time>
    <active>true</active>
</motion_image_inserter>
```

Example 2

The following example request modifies the running event with the ID 33. It stops the motion overlay. Notice that you include **action_time** with an empty value so that you clear the time currently specified in the XML. If you forget to do this, Elemental Live might try to apply that time to the next show/hide request.

```
POST http://198.51.100.22/live_events/33/motion_image_inserter
--------------------------------------------------------------
<motion_image_inserter>
    <action_time></action_time>
    <active>false</active>
</motion_image_inserter>
```

Types of changes

This table provides information about some of the changes you can make to a running event via the REST API:

State of motion overlay	Desired action	How to use the `motion_image_inserter` command	Tags to change
Not running	Run the motion overlay immediately.	Enter the command to set the `active` tag to `true` and set the <action_time> tag to empty.	`active` `action_time`
Not running	Run the motion overlay again at a specified time.	Enter the command to set the `active` tag to `true` and set the <action_time> tag to the desired start time.	`active` `action_time`
Not running	Run a different motion overlay of the same file type.	Enter the command to change the motion	All tags that apply to the current motion overlay type.

State of motion overlay	Desired action	How to use the **motion_image_inserter** command	Tags to change
		overlay to point to a different asset. Set the `<active>` tag to `true` and set the `<action_time>` tag to the time you want the start the new motion overlay. Set the `<loop>` tag to `true` or `false`.	
Running	Stop the running motion overlay.	Just before you want the motion overlay to stop, enter the command to set the `active` tag to `false`.	`active`
Running	Start the motion overlay.	To start the motion overlay again, enter the command to set the `<active>` tag to `true` and the `<action_time>` tag set to the time you want the motion overlay to begin.	`active`

State of motion overlay	Desired action	How to use the `motion_image_inserter` command	Tags to change
Running	Change to a different motion overlay.	Enter the command to change the motion overlay to a different file.	All tags, not just the ones you want to change. If you exclude a tag, the default value will apply.

Step D: Run the event again

You might want to run the same event again. If you do so, make sure that the motion overlay is set up correctly. You must do this because when you enter REST API commands to control the motion overlay when the event is running, the event XML for the motion overlay changes.

For example, you might set up the event with motion overlay A and a start time of 9:00 am. You then run the event and change the content and start time of the motion overlay several times. The last motion overlay to run in the event is motion overlay B with a start time of 4:00 pm. That is the information now in the event XML. Before you start the event the next day, review and modify the motion overlay information.

How to insert a motion overlay with a set of PNG files

You can use a set of `.png` files as an asset for the motion image overlay.

Here are some examples of how motion image overlay works with a set of PNG files:

- Example 1: “Coming up” motion overlay – You want to insert an asset that will run as a 10-second motion overlay 59 minutes into the runtime of the event. You want the motion overlay to be placed in the lower right corner of the video frame.
- Example 2: Animated corporate logo – You want to insert an asset and run it every 5 minutes, starting 20 minutes into the runtime of the event. You specify the single asset in the event and specify the first runtime. You start the event. After the motion overlay has run once, you send a REST API command to modify the start time of the asset to a new time. You repeat this command (each time with a new start time) as many times as you want.

Typically, you set up the event as follows:

- You prepare a set of PNG files and store them at a location that is accessible to Elemental Live.
- You configure the event with the location URL and start time of the first motion overlay you want to run, with position and size information, and with an instruction to run the motion overlay once or to loop it. You set up the motion overlay to be active when the event starts, and you enable control via the REST API.
- When you're ready, you start the event. The motion overlay configured in the event runs at the specified time. You then enter REST API commands to specify different content (a different URL), a start time, and a new position and size. After that motion overlay plays, you can enter another command to run different content, as required, for the duration of the event.

Step A: Prepare the png asset

1. Create a file – Use a third-party process to convert an animation asset to a series of PNG files.
2. Take care with these aspects of the conversion:

    - File count: When you insert the files into the video, you will specify the frame rate for the motion overlay. Therefore, make sure that the conversion results in a file count that aligns with the intended frame rate. (The frame rate of the motion overlay does not necessarily have to match the frame rate of the underlying video.) For example, if the motion overlay will run for 10 seconds at 30 frames/second, make sure that the conversion produces 300 files. If the file count and frame rate do not align, the quality of the animation might suffer.
    - The PNG files must contain RGB and one alpha channel. The alpha channel will be used for per-pixel blending.
    - File names: Make sure that the file names of the converted files include a sequential number. Numbering must start at 0. There can be any number of digits in the numerical part of the file name, as long as it is the same for each file. For example, 000 to 200 (three digits in both files) but not 00 to 200 (two digits in one file and three digits in the other file).
    - Aspect ratio: The motion overlay can have any aspect ratio. It does not have to match the aspect ratio of the video output.
    - Size: The motion overlay can be any size, in pixels, up to the size of the underlying video. The motion overlay must be prepared in the desired size; there is no way to resize it when setting it up in the event.

- Position: The motion overlay cannot be positioned so that part of the motion overlay runs beyond the right edge or bottom edge of the underlying video.
  - If you set up a motion overlay so that it is too big or it overruns and Elemental Live can identify this error at event creation time, then an error message will appear.
  - If Elemental Live cannot identify the error in advance, an error message will appear while the event is running. The event will not stop but the insertion request will fail.

3. Place the file – Place the converted file in a location accessible to Elemental Live: On a local directory, on a remote filesystem accessible via mount point, or in an Amazon S3 bucket. Choose a location as described in the section called “Fields for a PNG asset”, then note the location for setting up the motion overlay in the event.

   You can specify the location in one of the following ways:

   - Local to the Elemental Live appliance. For example, `/data/assets/motion overlay_001.png`
   - A remote server via a mount point. For example, `/data/mnt/assets/motion overlay_001.png`
   - An Amazon S3 bucket, using SSL. For example, `Amazon S3ssl://company.test/DOC-EXAMPLE-BUCKET/motion overlay_001.png`
   - An Amazon S3 bucket, without SSL. For example, `S3://company.test/DOC-EXAMPLE-BUCKET/motion overlay_001.png`

Step B: Set up the event

You configure the event with information about the first motion overlay. You can configure this information when creating an event. Or you can change the existing motion overlay configuration on a non-running event. You can configure the information using the web interface or the REST API.

Topics

Using the web interface

To configure the event using the web interface

1. In the **Global Processors** section, go to the **Image Inserter** field and choose **On**. More fields appear.
2. Complete the fields. You can configure some or all the information in the non-running event, but you must at least set the following fields:

   - **Insertion Mode**
   - **Enable REST Control**

   You must also configure the motion overlay behavior when the event first starts:

   - If you want the motion overlay to appear as soon as the event starts, specify all the fields. Make sure that you check **Active**, and make sure that you leave the **Action Time** empty.
   - If you don't want the motion overlay to appear as soon as the event starts, leave **Active** unchecked.

   You can configure the remaining fields after you've started the event, when you want to run the first motion overlay.

   For detailed information about the fields, see the section called “Fields for a PNG asset”.

Using the REST API

This description assumes that you are familiar with using the REST API and with the XML body for a `live_event`.

To configure the event using the REST API

1. Enter a POST or PUT command in the usual way. Use POST to create a new event. Use PUT to modify an existing event:

   ```
   POST http://<Live IP Address>/live_events
   ```

   ```
   PUT http://<Live IP Address>/live_events/<live event id>
   ```

2. In the body of the request, include one `motion_image_inserter` element inside the `live_event` tag. You can configure some or all the information in the non-running event, but you must at least set the following tags:

   - `insertion_mode`
   - `enable_rest`

   Make sure that you configure for the desired motion overlay behavior when the event first starts:

   - If you want the motion overlay to appear as soon as the event starts, specify all the tags. Make sure that you set `active` to `true`, and make sure that you leave `action_time` empty.
   - If you don't want the motion overlay to appear as soon as the event starts, set `active` to `false`.

   You can configure the remaining tags after you've started the event, when you want to run the first motion overlay.

   For detailed information about the tags, see the section called “Fields for a PNG asset”.

```
<motion_image_inserter>
    <action_time>
    <active>
    <enable_rest>
    <duration>
    <framerate_denominator>
    <framerate_numerator>
    <full_frame>
    <image_x>
    <image_y>
    <insertion_mode>
    <loop_input>
    <motion_image_inserter_input>
      <uri>
      <password>
      <username>
    </motion_image_inserter_input>
</motion_image_inserter>
```

3. The response repeats back the data that you posted with `<ID>` tags for many elements including IDs for the following motion image inserter elements:

   - `motion_image_inserter`
   - `motion_image_inserter_input`

Example

The following request creates an event with the name myLiveEvent. The event includes a `motion_image_inserter` section that inserts the motion overlay at a specific time:

```
POST http://198.51.100.22/live_events
-----------------------------------
  <?xml version="1.0" encoding="UTF-8"?>
      <live_event>
          <name>myLiveEvent</name>
          . . .
          <motion_image_inserter>
              <action_time>10:16:23:10</action_time>
              <active>true</active>
              <enable_rest>true</enable_rest>
              <framerate_denominator>1001</framerate_denominator>
              <framerate_numerator>30000</framerate_numerator>
              <full_frame>false</full_frame>
              <left>100</left>
              <top>200</top>
              <insertion_mode>png</insertion_mode>
              <loop_input>true</loop_input>
              <motion_image_inserter_input>
                  <uri>/data/logo001.png</uri>
              </motion_image_inserter_input>
          </motion_image_inserter>
      </live_event>
```

Fields for a PNG asset

Field on web interface	Tag in the XML	Type	Description
Insertion Mode	<insertion_mode>	String	Choose png.

Field on web interface	Tag in the XML	Type	Description
Input	<uri>	String	The path and file name of the PNG files. Provide the path and file name of the first PNG file in the series. All files in the series must have the same number of digits in the numerical part of the file name. For example, if the files are stored on /mnt/storage/motion_logos/ and the files are named logo_hi_001 to logo_hi_357, enter **`/mnt/storage/motion_logos/logo_hi_001`** . When using Amazon S3, you can optionally append the path as follows: • Use `sse=true` to turn on Amazon S3 Server Side Encryption (SSE).

Field on web interface	Tag in the XML	Type	Description
			• Use `rrs=true` to enable Reduced Redundancy Storage (RRS). Default values for RRS and SSE are `false`.
Username Password	<username> <password>		If access to your local or mounted directory requires a user name and password, click the lock icon next to the **Input** field to show the **Username** and **Password** fields. For Amazon S3, enter the access key ID in the username field. Enter the secret access key in the password field.

Field on web interface	Tag in the XML	Type	Description
Left	<image_x>	Integer	Placement of the left edge of the motion overlay relative to the left edge of the video frame, in pixels. 0 is the left edge of the frame. Take note of the width of the motion overlay and make sure that the position of the left edge of the motion overlay does not cause the right edge to be cropped.

Field on web interface	Tag in the XML	Type	Description
Top	<image_y>	Integer	Placement of the top edge of the motion overlay relative to the top edge of the video frame, in pixels. 0 is the top edge of the frame. Default 0. Take note of the height of the motion overlay and make sure that the position of the top edge of the motion overlay does not cause the bottom edge to be cropped.
ActionTime	<action_time>	String	The start time for the motion overlay. Specify the start time in one of the formats discussed in detail below this table.

Field on web interface	Tag in the XML	Type	Description
Numerator Denominator	framerate_numerator framerate_denominator	Integer	On the web interface, enter the frame rate as a numerator over a denominator. For example, 29.97 fps is a numerator of 30000 and a denominator of 1001. Enter numbers that give a frame rate ratio between 1 and 120. When using the REST API, enter the numerator and denominator separately.

Field on web interface	Tag in the XML	Type	Description
Loop Input	<loop_input>	Boolean	• Select to loop the motion overlay indefinitely. The motion overlay will run until the event ends. To stop the motion overlay earlier, see the section called "Step C: Manage at runtime". • Clear the check box to run the motion overlay only once.

Field on web interface	Tag in the XML	Type	Description
Full Frame	<full_frame>	Boolean	Expand the motion overlay to fit the video frame. In this case, make sure **Left** and **Top** are set to 0. If this field is selected and the motion overlay has a different aspect ratio to the underlying video, the motion overlay will be scaled until one of the following applies: • The motion overlay fits in the length. The motion overlay will then be positioned with equal space on the left and right. • The motion overlay fits in the width. The motion overlay will then be positioned with equal space above and below. Note that the **Stretch to output** field in the **Stream** section does

Field on web interface	Tag in the XML	Type	Description
			not affect the motion overlay; even if the video is stretched, the motion overlay is not stretched.
Active	<active>	Boolean	Always select this box when initially setting up the motion overlay. After the initial setup, the value of this tag can be changed via the REST API to manage the content and behavior of the motion overlay.
Enable REST Control	<enable_rest>	Boolean	Select this field only if you plan to manage motion overlays via the REST API, after this initial setup via the web interface . Typically, you will want this tag to be true.

Action time formats

Option 1: Timecode format (HH:MM:SS:FF).

The value to enter for HH:MM:SS:FF depends on the method used to calculate the *output timecode*. This method is specified in the **Timecode Configuration > Source** field. Identify the source method set for your event, then set the action time to match the timecode of the frame where you want the action to occur.

- If **Source** is **Embedded**: The output timecode is extracted from the timecode that is carried with the input media. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output. For example, 10:24:25:20, 10:24:25:21, and so on.
- If **Source** is **Start at 0**: The output timecode for the first frame is 00:00:00:00 and then the output timecode counts up with each successive frame in the entire output. For example, 00:00:00;01: 00;00:00:02, and so on.
- If **Source** is **System Clock** or **Local System Clock**: The output timecode for the first frame is the system time at which the frame is decoded. Then the output timecode counts up with each successive frame in the entire output. For example, if the first frame is decoded at 09:45:51, then that timecode is 09:45:51:01. The timecode for the next frame is 09:45:51:02, and so on.
- If **Source** is **Specified Start**: The output timecode for the first frame is the time you specified when you selected this option as the timecode source. Then the output timecode counts up with each successive frame in the entire output. For example, if you set the time to 15:30;00, then the timecode for the first frame is 15;30:00:01, and so on.
- If **Source** is **External Reference Connector**: The timecode is extracted from external LTC source. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output. For example, the timecode for the first frame is 20:03:19:01, then 20:03:19:01, and so on.

Option 2: ISO 8601 UTC time with no dashes or colons. For example, 20160102T030405.678Z. In this case, the start time for every motion overlay will be the UTC time.

Option 3: You can only use this option while adding or modifying a motion overlay in a running event. Set the `action_time` tag to an empty string to set the start time to “now”. With this option, the start time is never exact. You cannot use this option when creating an event or modifying a non-running event.

Step C: Manage the motion overlay on a running event

After the event starts, the motion overlay configured in the event runs at the specified time. When the event is running, you can work with the motion overlay only via the REST API. You can enter REST API commands to make the following changes:

- A different URL, to specify different content.
- A start time, the new position and size, and the specified loop behavior.

> **Note**
>
> Commands sent during an event change the event XML. Therefore, if you export the XML and create a new event with it, the new event will have any motion overlays set up as they were set during the course of the event, not as they were when the event was created.

To modify the motion overlay on a running event

1. Decide which fields you want to modify. For information about the types of changes that you can make, see the section called "Fields for a PNG asset".
2. Enter a POST command to modify any of the image overlay parameters in the running event:

   ```
   POST http://<Live IP Address>/live_events/<live event id>/motion_image_inserter
   ```

3. In the body of the request, include one **motion_image_inserter** element that contains the required tags. For more information about the tags, see the section called "Using the REST API".
4. The response repeats back the data that you posted with `<ID>` tags for `motion_image_inserter` and `motion_image_inserter_input`.

Example 1

The following example request modifies the running event with the ID 33. It sets up the currently defined motion overlay png to run again at 20160102T030405.678Z.

```
POST http://198.51.100.22/live_events/33/motion_image_inserter
--------------------------------------------------------------
<motion_image_inserter>
    <action_time>20160102T030405.678Z</action_time>
    <active>true</active>
</motion_image_inserter>
```

Example 2

The following example request modifies the running event with the ID 33. It stops the motion overlay. Notice that you include <action_time> with an empty value so that you clear the time currently specified in the XML. If you forget to do this, Elemental Live might try to apply that time to the next show/hide request.

```
POST http://198.51.100.22/live_events/33/motion_image_inserter
-------------------------------------------------------------
<motion_image_inserter>
    <action_time></action_time>
    <active>false</active>
</motion_image_inserter>
```

Types of changes

This table provides information about some of the changes you can make to a running event in the REST API:

State of motion overlay	Desired action	How to use the **motion_image_inserter** command	Tags to change
Not running	Run the motion overlay immediately	Enter the command to set the `active` tag to true and set the <action_time> tag to empty.	`active` `action_time`
Not running	Run the motion overlay again at a specified time	Enter the command to set the `active` tag to `true` and set the <action_time> tag to the desired start time.	`active` `action_time`
Not running	Run a different motion overlay of the same file type.	Enter the command to change the motion overlay to point to a different asset.	All tags that apply to the current motion overlay type.

State of motion overlay	Desired action	How to use the **`motion_image_inserter`** command	Tags to change
		Set the `<active>` tag to `true` and set the `<action_time>` tag to the time you want the start the new motion overlay. Set the `<loop>` tag to `true` or `false`.	
Running	Stop the running motion overlay	Just before you want the motion overlay to stop, enter the command to set the `active` tag to `false`.	`active`
Running	Start the motion overlay	To start the motion overlay again, enter the command to set the `<active>` tag to `true` and the `<action_time>` tag set to the time you want the motion overlay to begin.	`active`

State of motion overlay	Desired action	How to use the `motion_image_inserter` command	Tags to change
Running	Change to a different motion overlay.	Enter the command to change the motion overlay to a different file.	All tags, not just the ones you want to change. If you exclude a tag, the default value will apply.

Step D: Run the event again

You might want to run the same event again. If you do so, make sure that the motion overlay is set up correctly. You must do this because when you enter REST API commands to control the motion overlay when the event is running, the event XML for the motion overlay changes.

For example, you might set up the event with motion overlay A and a start time of 9:00 am. You then run the event and change the content and start time of the motion overlay several times. The last motion overlay to run in the event is motion overlay B with a start time of 4:00 pm. That is the information now in the event XML. Before you start the event the next day, review and modify the motion overlay information.

Static graphic overlay

You can insert a static overlay into the event so that it appears on the video in one or more of the video outputs. You can configure the static overlay with a start time and duration. You can insert the static overlay at any position on the video frame, as specified by x/y coordinates. You can configure with an opacity and with fade-in and/or fade- out.

You can insert any number of static overlays in the event.

Topics

- Insertion options and the effect on outputs
- Multiple overlays and layers

Insertion options and the effect on outputs

In the input section: Insert in portion of all outputs

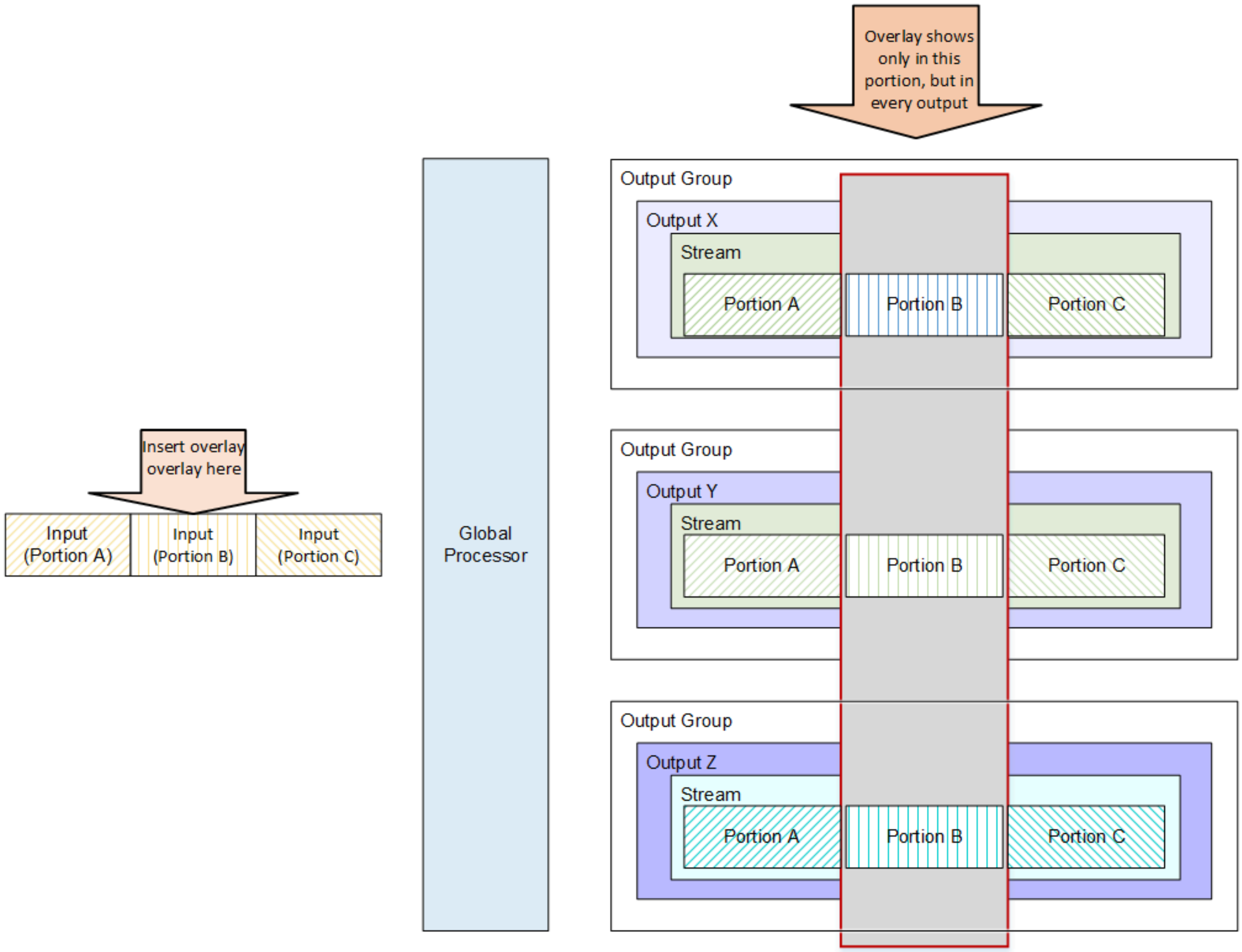
The static overlay can be inserted in the input section in a specific input.

Result: the portion of the output that is sourced from the given input includes the static overlay.

A typical use case for inserting in this section is if some of the inputs already have a static overlay in the desired location; you do not want to insert another static overlay over this existing static overlay. Therefore, you would insert the static overlay into the inputs that do not have an existing static overlay and omit it from inputs that do have an existing static overlay.

The static overlay will be burned into the video in this input after the input is decoded and before any global processors. This means that the static overlay "sticks" to its input; if the specified start time and duration extend beyond the end of the input, the static overlay will end when the input ends.

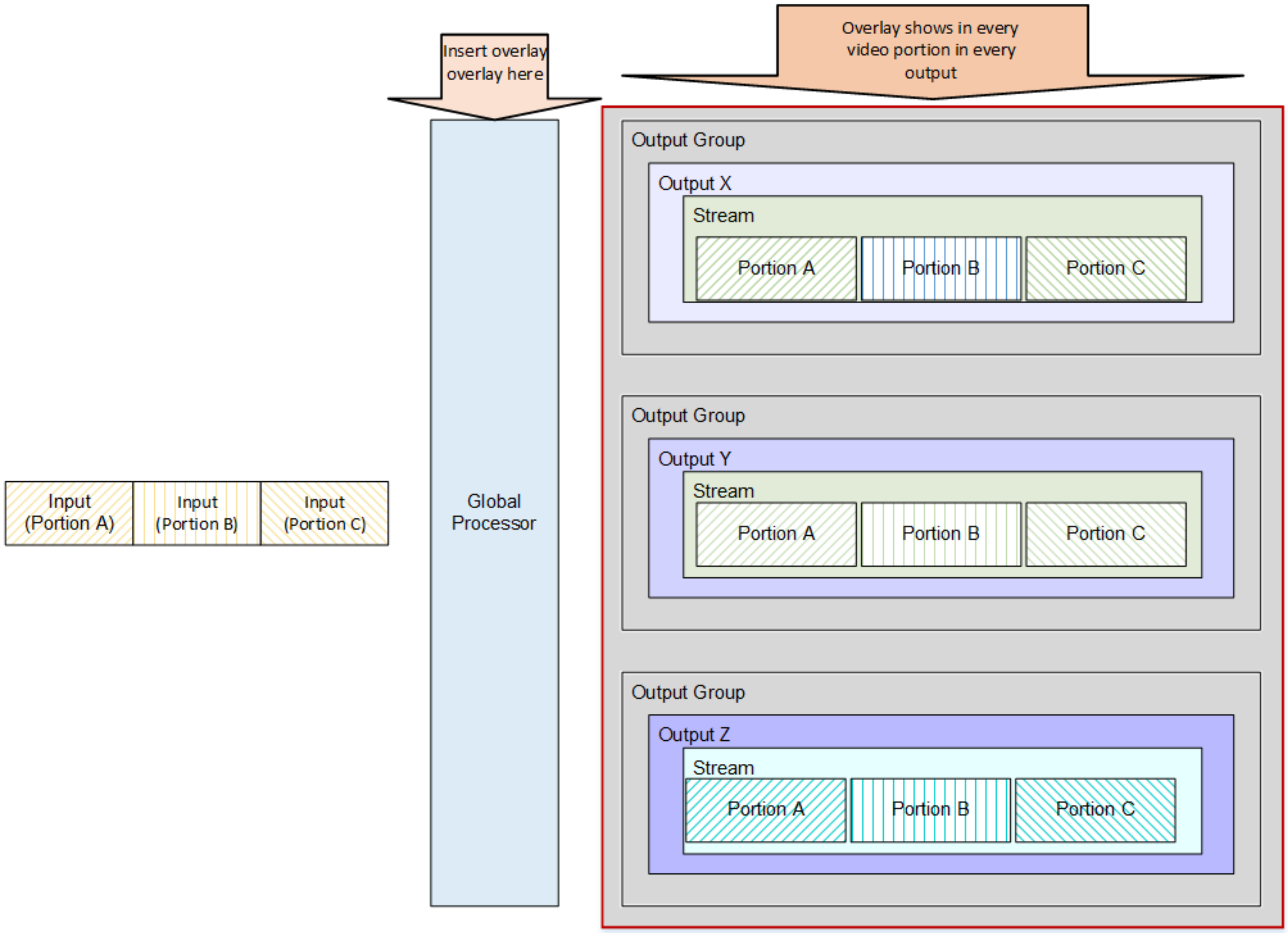
If you insert at the Input stage, be very careful with the start time and durations of the static overlays to avoid the static overlay ending abruptly.



In the Global Processors section: Insert in all outputs

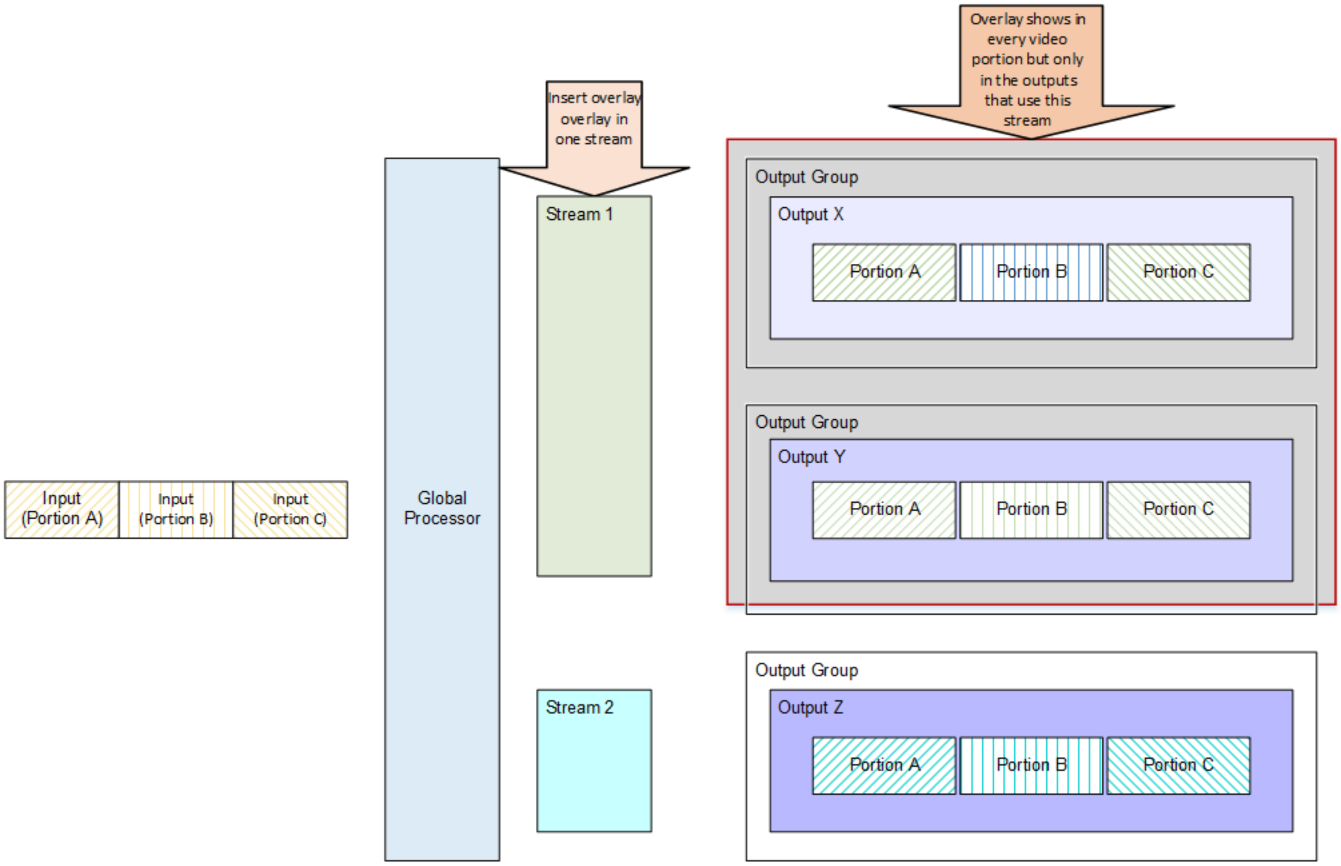
The static overlay can be inserted in the Global Processors section. Result: The static overlay will be inserted in all outputs.

The static overlay is burned into the video after decoding and input-specific processing and before encoding and creation of individual streams and outputs.



In the Output section: Insert in one stream

The static overlay can be inserted in individual streams.



Result: The static overlay is inserted only in the outputs that are associated with those streams. The static overlay will be burned into the video only in the specified streams.

Overlay scaling for each insertion option

The following lists the overlay scaling for each insertion option:

- If you insert the overlay in an input, then when (and if) the underlying video in each output is scaled up or down, the overlay is similarly scaled. The underlying video and overlay maintain their relative ratios. For example, if the overlay covers one quarter of the underlying video in the input, it will cover one quarter of the underlying video in every output.
- If you insert the overlay in the global processors section, then when (and if) the underlying video in each stream assembly is scaled up or down, the overlay is similarly scaled.
- If you insert the overlay in a stream assembly section, then when (and if) the underlying video in each stream assembly is scaled up or down, the overlay is not scaled. The overlay is added only after scaling. Therefore, if the event has two stream assemblies, the final effect of the overlay may be different. The overlay in one stream assembly may take up more room on the underlying video than the overlay in a stream assembly with a different resolution. But the advantage

of adding in the stream assembly section is that you can specify different overlays for each assembly, each sized appropriately for that stream's resolution.

Multiple overlays and layers

You can set up the event to insert more than one static overlay. Overlays are stored in the event in a queue that has a maximum of 8 layers.

This means that you can display up to:

- 8 static overlays if you are using only the web interface to set up the event.
- As many static overlays as you want over the duration of the event if you are using the REST API. See the section called "Step C". A maximum of 8 static overlays can be "queued" at one time (one in each layer).

Combining overlays and insertion options

You can set up the event to insert a static overlay in more than one way. For example, you can insert a static overlay in one layer in the Input section and insert a static overlay in another layer in the Global Processors section.

Examples

- Example 1 – You want to insert a static overlay at a specific time and run it for 10 seconds. You want the static overlay to be placed in the lower right corner of the video frame. You want the static overlay to be 50% opaque and to fade in from nothing to full 50% opacity over 2 seconds, then to fade out to nothing starting 2 seconds before the end of the insertion.

  You can implement this use case via the web interface or the REST API.
- Example 2 – You want to insert 2 static overlays so that they both appear on the video frame either at the same time, or with some overlap. You want the display of the static overlays to slightly overlap, so that one static overlay appears in a location and then while that static overlay is still showing, another static overlay appears in a specified location. If the locations overlap either partially or completely, you want to be able to specify which static overlay appears on top.

  When you want to insert between 1 and 8 static overlays, you can implement this use case via the web interface or the REST API.

- Example 3 – You want to insert 20 static overlays over the duration of the video. Some of the static overlays repeat, while some are unique. Some of them appear at the same time as others – up to 8 static overlays can appear at the same time.

  When you want to insert more than 8 static overlays, you must use the REST API.

- Example 4 – You want the same static overlay to appear repeatedly over the duration of the video, either in the same location each time or in different locations. This use case is simply a variation of use cases 2 or 3 with the same static overlay being used each time.

Procedure

	Step	Methods available
1.	Create the event with the desired static overlays.	Use the REST API or the web interface.
2.	Start the event.	Use the REST API or the web interface.
3.	While the event is running, set up more static overlays as desired.	Use the REST API.

You can implement all features described in this section via REST as well as the web interface. This section describes how to use the web interface to insert a graphic overlay. For instructions on doing so via REST, see the section called "Static overlays with REST API".

The following are valid web interface and REST API combinations:

- Initial setup via web interface and run-time changes via the REST API.
- Initial setup and run-time changes via the REST API.

If you use the web interface to perform the initial setup, note the following restrictions:

- You cannot specify more than 8 static overlays in the event.

- You must set up all the static overlays before the event starts to run; there is no mechanism for changing static overlays at runtime via the web interface.

Topics

Step A: Prepare the overlay asset

1. Create a file with the following characteristics:

   - File type: A BMP, PNG, or TGA file.
   - Aspect ratio: The overlay can have any aspect ratio. It does not have to match the aspect ratio of the underlying video.
   - Size, in pixels: The overlay can be any size up to the same size as the underlying video. The overlay cannot be positioned so that part of the overlay runs beyond the right edge or bottom edge of the underlying video.
     - If you set up an overlay so that it is too big or it overruns an edge, if Elemental Live can identify this error at event creation time, an error message will appear then.
     - If Elemental Live cannot identify the error in advance, an error message will appear while the event is running. The event will not stop but the insertion request will fail.
2. Place the prepared file in a location accessible to the Elemental Live node. You can specify the location in one of the following ways:

   - Local to the Elemental Live appliance. E.g. `/data/assets/overlay.mov`
   - A remote server via a mount point. E.g. `/data/mnt/assets/overlay.mov`
   - An S3 bucket, using SSL. E.g. `s3ssl://company.test/sample_bucket/overlay.mov`
   - An S3 bucket, without SSL. E.g. `s3://company.test/sample_bucket/overlay.mov`

   Use sse=true to enable S3 Server Side Encryption (SSE) and rrs=true to enable Reduced Redundancy Storage (RRS). Default values for RRS and SSE are false.

   Example: `s3://<hostname>/sample_bucket/ encrypted?rrs=true&sse=true`

3. Make a note of the location.

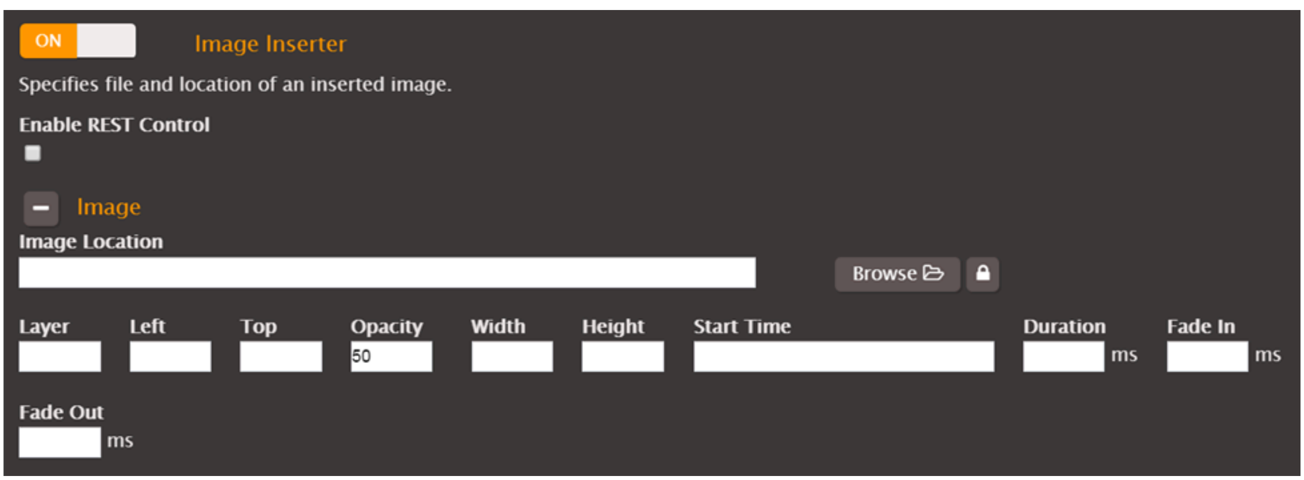
Step B: Initial setup

Create or modify the event as follows:

1. Determine the location or locations in the event where the static overlay should be inserted, then display the appropriate section:

   - Input section: In the desired input or inputs, click Advanced. More fields appear. In the Image Inserter section, click On. More fields appear; see the table in the next step.
   - Global Processors section: In the Global Processors section, go to the Image Inserter field and click On. More fields appear; see the table in the next step.
   - Output section: In the desired output or outputs, determine the stream this output is associated with. In the corresponding Stream section, click Advanced. More fields appear; see the table in the next step.

   For all locations, the following fields appear. (Note that the following image is from the Global Processors section, but the fields are the same in all sections.)



2. Complete the fields as follows:

Field	Description
Image Location	The location and filename of the PNG or BMP image file.

Field	Description
	For file requirement details, information about where to store this file, and how to specify its location, see the section called “Step A”. For S3, use `sse=true` to enable S3 Server Side Encryption (SSE) and `rrs=true` to enable Reduced Redundancy Storage (RRS). Default values for RRS and SSE are `false`.
Username Password	If access to your local or mounted directory requires a username and password, click the lock icon next to the Image Location field to show the Username and Password fields. For S3, Enter the Access Key ID in the username field. Enter the Secret Access Key in the password field.
Layer	A number from 0 to 7 for the Z order of the static overlay. “Z order” means that static overlays with higher values of layer will be inserted on top of static overlays with lower values of layer. Default is 0.
Left	Placement of the left edge of the motion overlay relative to the left edge of the video frame, in pixels. 0 is the left edge of the frame. Take note of the width of the motion overlay and make sure that the position of the left edge of the motion overlay does not cause the right edge to be cropped.

Field	Description
Top	Placement of the top edge of the motion overlay relative to the top edge of the video frame, in pixels.0 is the top edge of the frame. Default is 0. Take note of the height of the motion overlay and make sure that the position of the top edge of the motion overlay does not cause the bottom edge to be cropped.
Opacity	The opacity of the static overlay, as a number from 0 to 100. 0 is transparent. 100 is fully opaque. Default is 50.
Width	The width of the static overlay when inserted in the video, in pixels. Leave blank to use the native width of the static overlay. The original static overlay will be scaled up or down, to the specified width.
Height	The height of the static overlay when inserted in the video, in pixels. Leave blank to use the native height of the static overlay. The original static overlay will be scaled up or down, to the specified height.
Start Time	The start time for the overlay. Specify the start time in one of the formats listed at the section called “Start time formats”.

Field	Description
Duration	The amount of time, in milliseconds, for the overlay to remain on the video. If this field is left blank, the static overlay will remain on the video as follows: • In the Input section: Until this input ends. • In the Global Processor section: Until the event ends. • In the Output section: Until the event ends. The total running time of the static overlay is Fade in + Duration + Fade out.
Fade In	The duration, in milliseconds, for the static overlay fade-in. This time is inserted before the start time of the static overlay.
Fade Out	This field is valid only if the Duration field is completed. The duration, in milliseconds, for the static overlay fade-out. This time is added to the static overlay duration.
Enable Rest Control	Check this field only if you plan to manage motion overlays via the REST API, after this initial setup via the web interface. Typically, you will want this tag to be true.

3. If desired, click Add Image and enter the information for another static overlay, up to a maximum of 8 static overlays.

   - Assign a unique Layer number to each static overlay. The layers do not have to appear in any particular order on the screen, but each number must be used once only.

- The static overlay Start Time and Duration can be set so that any static overlay overlaps the display time of any other static overlay.
- The Left/Top fields can be set so that any static overlay physically overlaps any other static overlay, as much as you want; the static overlays are displayed according to their Layer value.

Note: If you are using input switching to provide input redundancy (with or without the hot-hot backup mode), then make sure you insert the static overlay in both input pairs.

Start time formats

Option 1: Timecode format (HH:MM:SS:FF). The overlay start is determined by comparing the specified start to the appropriate timecode.

- If the overlay is specified in the Input section: The start time for the static overlay will be compared to the input timecode (the timecode for a given input). The source for the input timecode is specified separately for each input (Input > Timecode Source field). The input timecode is calculated as follows:
  - If Timecode Source is Embedded: The timecode associated with each frame is extracted from the timecode carried with the input media. Note that each input will have its own timecode and the timecode may not align well from one input to another.
  - If Timecode Source field is Start at 0: The timecode of the first frame of the input is 00:00:00:00 and the timecode counts up with each successive frame. The timecode starts over with each input.
  - If Timecode Source field is System Clock or Local System Clock (AWS Elemental Live only): The timecode of each frame in the input is the system time at which the frame is decoded.
- If the overlay is specified in the Global Processor section: The overlay start is compared to the output timecode (which is shared by all outputs). The source for the output timecode is specified for the entire event, in the Timecode Config > Source field. The output timecode is calculated as follows:
  - If Source is Embedded: The timecode is extracted from the timecode carried with the input media. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output.
  - If Source is Start at 0: The output timecode for the first frame is 00:00:00:00 and then the output timecode counts up with each successive frame in the entire output.

  - If Source is System Clock or Local System Clock (AWS Elemental Live only): The output timecode for the first frame is the system time at which the frame is decoded. Then the output timecode counts up with each successive frame in the entire output.
  - If Source is Specified Start: The output timecode for the first frame is the time you specified when you selected this option as the timecode source. Then the output timecode counts up with each successive frame in the entire output.
  - If Source is External Reference Connector (AWS Elemental Live only): The timecode is extracted from external LTC source. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output.
- If the static overlay is specified in the Output section: The start time for the static overlay is calculated in the same way as a static overlay in the Global Processor section.

Option 2: ISO 8601 UTC time with no dashes or colons. For example, 20160102T030405.678Z. In this case, the start time for every overlay (regardless of whether it is defined in the input, the global processor or the output) will be the UTC time.

Option 3: Only when adding or modifying an overlay in a running event (not when creating an event or modifying a non-running event), set this tag to an empty string to set the start time to “now”. With this option, the start time is never exact.

Step C: Manage overlays on a running event

Once the event has started, you can work with static overlays only via the REST API. You cannot work with static overlays on a running event via the web interface.

Change the static overlay or overlays on a running event:

1. If you did not set the `<enable_rest>` element to `true` when you created the event, modify the event (PUT Event) and set this value. For the location of this element, see the section called “Modify static overlay”.
2. Send the Modify Static Overlay command (see the section called “Static overlays with REST API”) to make the desired change to the static overlays in the event.

Runtime REST commands change the event XML

When you send REST commands during an event, the event XML is permanently changed. Any data sent via the REST call goes into the XML.

For example, you might put a scoreboard overlay on your event during a sporting event. If you do not send a REST call to deactivate the overlay once the game has ended, the scoreboard will appear again at the same time the next day.

Therefore, make sure to do one of the following:

- If you plan to run the event (event A) again with different video content but with the same graphic overlays, make sure to export your XML for re-use before starting the event. Then create a new event (event B) using the exported XML. Do not start event A again.
- If you are running a 24/7 channel and you do not want your overlay to recur, remember to send a REST command to set <activate> to false once the overlay has run. This will delete the entire <insertable_images> element from the event XML.

  You could also specify an absolute start time for each overlay by using the ISO 8601 UTC time format to specify an absolute time and date for the overlay. The overlays will not run again the next day.

Options for insertion - which outputs are affected

You can insert static overlays in one of the following places in the running event. These places are the same as the places when inserting in a new event or modifying a non-running event.

- In an individual input. The input must be currently running or be in the future.

  If you include a start time in the XML body, that start time must fall within the scope of the specified input. It must correspond to a time that is valid for that input. For example, if the input runs from 1:00 p.m. to 2:00 p.m. by the clock, the start time must correspond to a clock time between 1:00 p.m. and 2:00 p.m., otherwise the insertion will be ignored. The start time can be in the past, so long as it is within the scope of the input; in this case, the overlay will be inserted immediately.
- In all outputs.

  If you include a start time in the XML body, the overlay will be inserted at that at start time. If that start time is in the past, the overlay will be inserted immediately.
- n the outputs associated with one stream assembly.

  If you include a start time in the XML body, the overlay will be inserted at that at start time. If that start time is in the past, the overlay will be inserted immediately.

Types of changes

Add an overlay in a layer

You can add an overlay in a layer. For example, if you did not fill all 8 layers when creating the event, you can add more static overlays, up to a total of 8 for the event. Or if you already deleted a layer (as described below), you can fill it with a new static overlay.

You must enter a Modify Static Overlay command (the section called “Create or modify a non-running event”) and include the following tags in the XML body:

- layer: The (unused) layer where you want to add the static overlay.
- activate: Set to true. Note that this tag is not part of the XML body for creating the event.
- Other tags: Set all other tags as desired to specify the content and its characteristics, start time and duration.

Modify an existing overlay

You can modify the existing content in a layer. You can do the following:

- Change a static overlay that has not yet run.
- Change a static overlay that is currently running. The static overlay will change in mid-stream.
- Change a static overlay that has run in order to re-use the layer.

You can change the static overlay’s start time or duration. Or you can change its position. Or you can change the actual overlay that runs.

You must enter a Modify Static Overlay command (the section called “Create or modify a non-running event”) and include the following tags in the XML body:

- layer: The layer whose static overlay you want to modify.
- activate: Set to true. Note that this tag is not part of the XML body for creating the event.
- Other tags: Set all other tags as desired to specify the content and its characteristics, start time and duration. Only the tags you specify will be changed.

Delete an overlay

You can delete the existing content in a layer. If the static overlay has not yet run, it will not run. If the static overlay is currently running, it will be removed. If the static overlay has already run, there is not really any need to delete the content.

You must enter a Modify Static Overlay command (the section called “Create or modify a non-running event”) and include the following tags in the XML body:

- layer: The layer to delete.
- activate: Set to false. Note that this tag is not part of the XML body for creating the event.

Using the REST API for static overlays

Topics

Static graphic overlay commands

Nickname	Action	Signature	Description
Create Event	POST	`<Live IP address>/live_events`	Create an event that includes static overlay information.
Modify Event	PUT	`<Live IP address>/live_events/live_event/<event ID>`	Modify an event (that is not running) and add, modify or delete static overlay information.
Modify Static Overlay, Running Event	POST	`/live_event/<event ID>/image_inserter`	All outputs: add, modify or delete static information in a running event.

Nickname	Action	Signature	Description
Modify Static Overlay, Running Event	`POST`	`<Live IP Address>/live_events/<event ID>/image_inserter/output/<output id>` Where `<output id>` is the unique ID automatically assigned to this output when the event is created.	Specific output: add, modify or delete static information in a running event.

Nickname	Action	Signature	Description
Modify Static Overlay, Running Event	POST	`<Live IP Address>/live_events/<event ID>/image_inserter/output/by_stream/<stream id>` Where `<stream id>` is the unique ID automatically assigned to this stream when the event is created. The ID can change while the event is running (for example, if another stream is deleted), so you may need to obtain the current ID before sending this command.	All outputs associated with a specific stream: add, modify or delete static information in a running event.

Nickname	Action	Signature	Description
Modify Static Overlay, Running Event	POST	`<Live IP Address>/live_events/<event ID>/image_inserter/input/<input id>` Where `<input id>` is the unique ID automatically assigned to this input when the event is created or when the input is added to the event.	Specific input, all outputs associated with it: add, modify or delete static information in a running event.
Modify Static Overlay, Running Event	POST	`<Live IP Address>/live_events/<event ID>/image_inserter/input/by_label/<input label>` Where `<input label>` is the input label you assigned when you created this event or created this input. Input labels are always optional.	Specific input, all outputs associated with it: add, modify or delete static information in a running event.

Create or modify a non-running event with static graphic overlay

Create or modify an Elemental Live event and include one or more static graphic overlays. This description assumes that you are familiar with the XML body for a live_event aside from the data for the graphic overlay.

HTTP request and response

HTTP URL

```
POST http://<Live IP Address>/live_events
```

or:

```
PUT http://<Live IP Address>/live_events/<event ID>
```

Body of request – one input

To insert the static overlay in one input element:

XML content consisting of one live_event element that contains:

- All the usual elements and tags.
- One input element that contains:
  - All the usual elements and tags.
  - One image_inserter element that contains:
    - One enable_rest tag.
    - One insertable_images element that contains:
      - 1 to 8 insertable_image elements that contain the tags listed in the following table.

For a representation of the XML structure, see the section called “XML structure”.

Body of request – all outputs

To insert the static overlay in all outputs:

XML content consisting of one live_event element that contains:

- All the usual elements and tags.
- One image_inserter element that contains:

  - One enable_rest tag.
  - One insertable_images element that contains:
    - 1 to 8 insertable_image elements that contain the tags listed in the table on the following page.

For a representation of the XML structure, see the section called “XML structure”.

Body of request – outputs for one stream assembly

To insert the static overlay in the outputs associated with a given stream. XML content consisting of one live_event element that contains:

- All the usual elements and tags.
- One stream_assembly element that contains:
  - All the usual elements and tags.
  - One image_inserter element that contains:
    - One enable_rest tag.
    - One insertable_images element that contains:
      - 1 to 8 insertable_image elements that contain the tags listed in the following table.

For a representation of the XML structure, see the section called “XML structure”.

Child elements of <insertable_image> and <image>

Element	Type	Required	Description for creating
activate	Boolean	Required	Required when adding or modifying an overlay in a running event (not when creating an event or modifying a non-running event). This tag has no effect when creating or

Element	Type	Required	Description for creating
			modifying a non-running event. • Set to true when adding any overlay or modifying an overlay (in specified the layer) in an existing event. • Set to false to delete the overlay (specified in the layer tag) from the underlying video. Note that the entire overlay is also deleted from the event XML. (If you do not specify a layer tag in the body, the overlay won't be deleted.) **Note:** <activate > is distinct from <active>, which is used with motion image inserter.

Element	Type	Required	Description for creating
duration	Integer	Optional Default: until end of event	The time in milliseconds for the overlay to remain on the video. When creating an event, if this field is left blank, the overlay will remain on the video as follows: • In the Input section: Until this input ends. • In the Global Processor section: Until the event ends. • In the Output section: Until the event ends. The total running time of the overlay is fade_in + duration + fade_out.
fade_in	Integer	Optional	The duration, in milliseconds, for the overlay fade-in. This time is inserted before the start time of the overlay.

Element	Type	Required	Description for creating
fade_out	Integer	Optional	This field is valid only if the Duration field is completed. The duration, in milliseconds, for the overlay fade-out. This time is added to the overlay duration.
height	Integer	Optional Default: native height of overlay	The height of the overlay when inserted in the video, in pixels. When creating an event, leave blank to use the native height of the overlay. The original overlay will be scaled up or down, to the specified height.

Element	Type	Required	Description for creating
width	Integer	Optional Default: native width of overlay	The width of the overlay when inserted in the video, in pixels. When creating an event, leave blank to use the native height of the overlay. The original overlay will be scaled up or down, to the specified width.
image_inserter_input	Location	Required	Overlay PNG or BMP file to insert. For file requirement details, information about where to store this file, and how to specify its location, see the section called “Step A”.

Element	Type	Required	Description for creating
image_x	Integer	Required	Placement of the left edge of the overlay relative to the horizontal axis for the video frame, in pixels. 0 is the left edge of the frame. When creating an event, cannot be omitted. Take note of the width of the overlay and make sure that the position of the left edge of the overlay does not cause the right edge to be cropped.

Element	Type	Required	Description for creating
image_y	Integer	Required	Placement of the top edge of the overlay relative to the vertical axis for the video frame, in pixels. 0 is the top edge of the frame. When creating an event, cannot be omitted. Take note of the height of the overlay and make sure that the position of the top edge of the overlay does not cause the bottom edge to be cropped.

Element	Type	Required	Description for creating
layer	Integer	Required	A number from 0 to 7 to specify the Z order of the inserted overlay. "Z order" means that overlays with higher values of layer will be inserted on top of overlays with lower values of layer. Must always be specified. In the XML for modifying a static overlay at runtime, if the XML has more than one image container, then each layer tag must have a different value. So each overlay must be in its own layer.
opacity	Integer	Optional Default: 50	The opacity of the overlay, as a number from 0 to 100. 0 is transparent. 100 is fully opaque. When creating an event, cannot be omitted.

Element	Type	Required	Description for creating
start_time	String	Optional Default: beginning of the event	The start time for the overlay. Specify the start time in one of the formats discussed below this table.

Start time formats

Option 1: Timecode format (HH:MM:SS:FF). The overlay start is determined by comparing the specified start to the appropriate timecode.

- If the overlay is specified in the Input section: The start time for the static overlay will be compared to the input timecode (the timecode for a given input). The source for the input timecode is specified separately for each input (Input > Timecode Source field). The input timecode is calculated as follows:
  - If Timecode Source is Embedded: The timecode associated with each frame is extracted from the timecode carried with the input media. Note that each input will have its own timecode and the timecode may not align well from one input to another.
  - If Timecode Source field is Start at 0: The timecode of the first frame of the input is 00:00:00:00 and the timecode counts up with each successive frame. The timecode starts over with each input.
  - If Timecode Source field is System Clock or Local System Clock (AWS Elemental Live only): The timecode of each frame in the input is the system time at which the frame is decoded.
- If the overlay is specified in the Global Processor section: The overlay start is compared to the output timecode (which is shared by all outputs). The source for the output timecode is specified for the entire event, in the Timecode Config > Source field. The output timecode is calculated as follows:
  - If Source is Embedded: The timecode is extracted from the timecode carried with the input media. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output.
  - If Source is Start at 0: The output timecode for the first frame is 00:00:00:00 and then the output timecode counts up with each successive frame in the entire output.

  - If Source is System Clock or Local System Clock (AWS Elemental Live only): The output timecode for the first frame is the system time at which the frame is decoded. Then the output timecode counts up with each successive frame in the entire output.
  - If Source is Specified Start: The output timecode for the first frame is the time you specified when you selected this option as the timecode source. Then the output timecode counts up with each successive frame in the entire output.
  - If Source is External Reference Connector (AWS Elemental Live only): The timecode is extracted from external LTC source. That timecode becomes the output timecode for the first transcoded frame. Then the output timecode counts up with each successive frame in the entire output.
- If the static overlay is specified in the Output section: The start time for the static overlay is calculated in the same way as a static overlay in the Global Processor section.

Option 2: ISO 8601 UTC time with no dashes or colons. For example, 20160102T030405.678Z. In this case, the start time for every overlay (regardless of whether it is defined in the input, the global processor or the output) will be the UTC time.

Option 3: Only when adding or modifying an overlay in a running event (not when creating an event or modifying a non-running event), set this tag to an empty string to set the start time to "now". With this option, the start time is never exact.

Example

The following request creates an event with the name myLiveEvent and includes one static overlay to insert the file logo.bmp. The overlay is inserted directly inside the live_event, which means it will appear in all outputs.

```
POST http://198.51.100.22/live_events
```

```
<?xml version="1.0" encoding="UTF-8"?>
  <live_event>
    <name>myLiveEvent</name>
    ...
      <image_inserter>
        <enable_rest>true</enable_rest>
        <insertable_images>
          <insertable_image>
            <duration>30000</duration>
            <fade_in>10</fade_in>
```

```
            <fade_out>10</fade_out>
            <height>900</height>
            <left>300</left>
            <top>400</top>
            <layer>0</layer>
            <start_time>16:09:09:10</start_time>
            <width>800</width>
            <image_inserter_input>
              <uri>mnt/storage/logo.bmp</uri>
            </image_inserter_input>
          </insertable_image>
        </insertable_images>
      </image_inserter>
     ...
</live_event>
```

Modify static overlay on a running event

In a running event, you can use the REST API to add more static overlays, modify the behavior of an existing static overlay, or delete an existing static overlay.

> **Note**
>
> Commands sent during an event change the event XML. Therefore, if you export the XML and create a new event with it, the new event will have any overlays set up as they were set during the course of the event, not as they were when the event was created.

HTTP request and response

HTTP URL - one input

To add, modify, or delete the static overlay in one input element.

```
POST http://<Live IP Address>/live_events/<id>/image_inserter/input/<input id>
```

Where `<input id>` is the unique ID automatically assigned to this input when the event is created or when the input is added to the event.

or:

```
POST http://<Live IP Address>/live_events/<id>/image_inserter/input/by_label/<input
 label>
```

Where `<input label>` is the input label you assigned when you created this event or created this input. Input labels are always optional.

HTTP URL - all outputs

To add, modify, or delete the static overlay in all outputs.

```
POST http://<Live IP Address>/live_events/<id>/image_inserter
```

HTTP URL - one output

To add, modify, or delete the static overlay in one output.

```
POST http://<Live IP Address>/live_events/<id>/image_inserter/output/<output id>
```

Where `<output id>` is the unique ID automatically assigned to this output when the event is created.

HTTP URL - outputs for one stream assembly

To add, modify, or delete the static overlay in the outputs associated with a given stream.

```
POST http://<Live IP Address>/live_events/<id>/image_inserter/output/by_stream/<stream
 id>
```

Where `<stream id>` is the `<ID>` automatically assigned to this stream when the event is created. The ID can change while the event is running (for example, if another stream is deleted), so you may need to obtain the current ID before sending this command.

Body of request

XML content consisting of:

- One or more image_inserter elements that each contains:
  - 1 to 8 <image> elements that each contains the tags in the table in the section the section called “Create or modify a non-running event”.

For information about the tags to include for different actions, see the section called “Types of changes”.

Response

The response repeats back the data that you posted with `<ID>` tags for `image_inserter` and each overlay. If the event is not running, the message “Event <ID> is not running” is returned.

Example

The following request modifies the overlays in the event with the ID 16. It modifies the start time on the existing overlay in layer 3.

It also adds one overlay (in layer 4); if an overlay already exists in layer 4, that overlay is replaced with the new overlay (even if that overlay is currently running). If an overlay does not exist in layer 4, the overlay is added.

```
POST http://198.51.100.22/live_events/16/image_inserter
```

```
<?xml version="1.0" encoding="UTF-8"?>
  <image_inserter>
    <image>
      <activate>true</activate>
      <layer>3</layer>
      <start_time>17:09:09:10</start_time>
    </image>
    <image>
      <activate>true</activate>
      <duration>30000</duration>
      <fade_in>10</fade_in>
      <fade_out>10</fade_out>
      <height>900</height>
      <left>300</left>
      <top>400</top>
      <layer>4</layer>
      <start_time>16:09:09:10</start_time>
      <width>800</width>
      <image_inserter_input>
        <uri>mnt/storage/logo.bmp</uri>
      </image_inserter_input>
    </image>
  </image_inserter>
```

Static overlay plus dynamic content switching

You can combine the static overlays with the dynamic content switching feature. You can use dynamic content switching to continually add and modify inputs in an event that is running. As you add inputs, you can insert static or motion overlays, as desired. For information about dynamic content switching, see Dynamic input switching.

Scheduling inputs and overlays

The scheduling of the inputs and the overlays is completely decoupled.

A given input X might be added several times to the dynamic playlist, for example, so that it plays at 2:00 p.m. and then plays again at 3:10 p.m.

If you want an overlay to appear the first time that input X plays, you might set its start time for 2:10 p.m. The next time that input X plays, there is no logic to play the same overlay again because 2:10 p.m. has passed. Therefore, if you want the overlay to appear again on input X, you must send a Modify Overlay call again.

Behavior with different insertion options for static overlays

Following are points to remember when using static overlay with dynamic content switching:

Playlist input plus overlay at the input stage

You specify the overlay in the input section of the event. Therefore, whenever you include this input in the playlist, the overlay will be included. If you want to the image the first time the input appears in the playlist but you don't want to include it the next time the input appears, you must unless enter an action to remove it.

- If you want to include the overlay in a given repetition of the input, you might need to change the start time of the overlay. See the information about the Start Time in the table under the section called “Step B”.
- If you do not want to include the overlay in a given repetition of that input, you must use the REST interface to enter a Modify Static Overlay command and delete the overlay. See the section called “Commands”.

Playlist input plus overlay at the global processing stage

The scheduling of the inputs and the overlays is completely decoupled.

A given input X might be added several times to the dynamic playlist, for example, so that it plays at 2:00 p.m. and then plays again at 3:10 p.m.

If you want an overlay to appear the first time that input X plays, you might set its start time for 2:10 p.m. The next time that input X plays, there is no logic to play the same overlay again because 2:10 p.m. has passed.

Therefore, if you want the overlay to appear again on input X, you must enter the Modify Static Overlay command again.

Playlist input plus overlay at the output stage

The same comments as for global processing stage apply.

XML structure

Topics

- Static overlay, creating and modifying a non-running event
- Static overlay: Modifying overlay on a running event

Static overlay, creating and modifying a non-running event

Overlays at top level

<image_inserter>			
	enable_rest		
	<insertable_image>		
		duration	
		fade_in	
		fade_out	
		height	
		image_x	

		image_y	
		layer	
		opacity	
		start_time	
		width	
		<image_inserter_input>	
			certificate_file
			interface
			password
			uri
			username
		</image_inserter_input>	
	</insertable_image>		
</image_inserter>			

Overlays in input section

Data in <input> element

<input>				
	<image_inserter>			
		enable_rest		

		<insertab le_image>		
			duration	
			fade_in	
			fade_out	
			height	
			image_x	
			image_y	
			layer	
			opacity	
			start_time	
			width	
			<image_in serter_input>	
				certificate_file
				interface
				password
				uri
				username
			</image_i nserter_input>	
		</inserta ble_image>		

Overlays in Ssream assembly section

Data in <stream_assembly>

<stream_assembly>						
	<video_description>					
		<video_preprocessors>				
			<image_inserter>			
				enable_rest		
				<insertable_image>		
					duration	
					fade_in	
					fade_out	
					height	
					image_x	
					image_y	
					layer	
					opacity	
					start_time	

					width	
					<image_in serter _input>	
						certifica te_file
						interface
						password
						uri
						username
					</image_i nserter _input>	
				</inserta ble_image >		
			<image_in serter>			

Static overlay: Modifying overlay on a running event

<image_inserter>			
	<image>		
		activate	
		duration	

		fade_in	
		fade_out	
		height	
		image_x	
		image_y	
		layer	
		opacity	
		start_time	
		width	
		<image_inserter_input>	
			certificate_file
			interface
			password
			uri
			username
		</image_inserter_input>	
	</image>		
</images_inserter>			

Input switching

You can configure an Elemental Live event with multiple inputs, and then implement an input switching mechanism to switch from ingesting one of those inputs to ingesting another.

There are several ways to set up for input switching.

Dynamic input switching

The dynamic input switching feature lets use REST API commands to switch inputs in an event. This input switching feature has two key characteristics:

- The switching control (the REST API) is built into Elemental Live. You don't need an external server to control switching.
- You can optionally modify the playlist of inputs without stopping the event. This ability to dynamically change at runtime the playlist is particularly useful when you can't identify in advance (before starting the For more information about dynamic input switching, see the section called “Dynamic input switching”.

Virtual input switching

The virtual input switching feature lets you set up so that a POIS controls switching from one input to another. This feature has these key switching methods:

- Virtual input switching via SCTE-35 messages

  This type of input switching works with SCTE-35 messages that are in the input. We also refer to this switching as SCTE-35-triggered input switching.

  Virtual input switching via asynchronous ESAM messages that the POIS sends.

  This type of input switching is based only on decisions from the POIS. It doesn't rely on messages that are in the input. We also refer to this switching as asynchronous input switching.

  For more information about virtual input switching, see the section called “Virtual input switching”.

Automatic input switching

This type of switching is applicable when an event contains file inputs that aren't set up to loop.

If Elemental Live gets to the end of a file input that isn't set up to loop, then Elemental Live switches to the next input in the list of inputs in the event.

We don't recommend relying on automatic switching.

Combining different input switching features

We strongly recommend against implementing a combination of these input switching features. You shouldn't give control of the input switching logic to both a POIS and a REST-based application. There should be only one mechanism controlling the input switching.

Inserting Nielsen watermarks

Starting with Elemental Live version 2.22.0, you can set up to create new Nielsen watermarks and insert them into your audio. Typically, only content and distribution providers use Nielsen watermarks. If you're not working with The Nielsen Company to implement watermarks, you don't need to read this section.

If your content already contains watermarks, you might choose to convert them to ID3 metadata and include that metadata in the output. For information about conversion to ID3, see the section called "Nielsen watermarks to ID3".

The Nielsen watermark feature requires the license (the Nielsen Audio Watermark Package). Contact your AWS sales manager.

Topics

Audio requirements

Supported audio

The audio must meet the following requirements:

- Sample-rate frequency: 48 kHz (48000 samples per second).
- Number of channels: Up to 8 audio channels.
- Interleaved samples.
- The audio must conform to one of the coding modes and channel layouts specified in the following table.

In the table, read across each row to identify the channel layout for the coding mode that is identified in the first cell.

Coding mode	Ch 1	Ch 2	Ch 3	Ch 4	Ch 5	Ch 6	Ch 7	Ch 8
Mono	Left							
Stereo	Stereo left	Stereo right						
5.1 audio	Front left	Front right	Center	LFE	Surround left	Surround right		
5.1 audio plus stereo	Front left	Front right	Center	LFE	Surround left	Surround right	Stereo left	Stereo right

Recommended minimum bitrate

We recommend the minimum audio bitrates listed in the following table. If you set the audio bitrates lower than the recommended values, your watermarks might not be reliably detected.

Codec	Coding mode	Minimum bitrate (kbps)
Dolby Digital	Stereo	192
	5.1	384
Dolby Digital Plus	Stereo	192

Codec	Coding mode	Minimum bitrate (kbps)
	5.1	192
AAC with the LC profile	Stereo	128
AAC with the HEV1 profile	5.1	256
MPEG-1, layer II	Stereo	96

Getting ready

To get ready for watermarks

1. Determine if you should insert NAES II or CBET watermarks. NAES II are used in the United States. CBET are used in Canada. You can insert one or both types in the same audio encode.
2. Obtain the following information from your contact at The Nielsen Company:
    - For Critical Band Encoding Technology (CBET) encoding:
        - CBET Source Identification (CSID) code.
        - CBET check digit code.
    - For NAES II encoding:
        - Source Identification Code (SID).
        - NAES check digit code.

    You must obtain separate sets of values for each Elemental Live event.
3. If you are setting up CBET watermarks, decide how you want to handle watermarks that are already in the source audio. The options are the following:
    - Remove all the existing watermarks and replace them with new ones.
    - Keep the existing watermarks. Elemental Live will insert new marks only in portions of the audio stream where there are no watermarks.

Setting up Nielsen watermarks

You can create new Nielsen watermarks on the output audio encoding. To pass through existing Nielsen watermarks or convert them to ID3, see the section called "Nielsen watermarks to ID3".

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event. It also assumes that you have already set up the audio encodes (outputs) that will contain the watermarks.

To create Nielsen watermarks

1. On the **Event** page of the web interface, go to the **Streams** section, then go to the specific stream that contains the audio that you want to set up with watermarking.
2. Choose the **Audio** tab. Then open the **Advanced** section. More fields appear.
3. Set **Nielsen Watermarking** to **On**. More fields appear.
4. Set the following fields to suit your requirements. For information about a field, hover on the upper-right corner of the field and choose the **?** icon.
    - Process Type
    - Distribution Type
5. Complete the remaining fields with the values you determined when getting ready.

Converting Nielsen watermarks to ID3

If one or more inputs in an event includes Nielsen watermarks in the audio, you have the option to set up the event to convert those watermarks to ID3 metadata. These watermarks are part of the measurement and analytics capabilities supported by Nielsen.

This option applies only in the following scenario:

- One or more inputs in your event includes Nielsen watermarks in the audio.
- Your event has at least one output group that can include the Nielsen ID3 tag. For example, an HLS output group.

- You know that at least some of your playback devices implement the Nielsen SDK. This SDK provides functionality to handle the ID3 tags.

Converting the watermarks to ID3 tags doesn't remove the original watermarks. Outputs where you include the ID3 tags will contain both the watermark and the ID3 tags. Outputs that don't include the ID3 tags will contain only the watermark.

You can't remove the watermarks from the audio, but if your playback devices don't implement the Nielsen SDK, the devices ignore the watermarks.

To set up watermarks as ID3 tags

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event. It also assumes that you have already set up the output groups that will contain the ID3 tags.

1. On the **Event** page of the web interface, go to the **Nielsen Configuration** section. This section is before the **Global Processors** section.
2. Set the fields to enable and configure the feature:
   - **Enable Nielsen PCM to ID3 tagging**: Choose the check box to enable the feature.
   - **Distributor ID**: This field appears when you enable the feature. Optionally, enter the distributor ID that you obtained from Nielsen. If you enter an ID here, it is added to the ID3 metadata along with the source ID (SID) that is always in the source watermark.

   Information following these fields describes the supported Nielsen SDK and the vendor ID that ID3 tags will use.
3. Go to the output group where you want to include the ID3 tags. The output group must be an Archive, Apple HLS, or UDP/TS group. You can set up one or more output groups to include ID3 tags.
4. If the output group doesn't have an output, choose the **Add Output** button. An **Outputs** section appears.
5. In the **Outputs** section, take the appropriate action:

  - For an Archive output group, in **Container**, choose **MPEG-2 Transport Stream**. This is the only type of container that supports ID3 tags.
  - For an Apple HLS output group, in **Segment Type**, choose **TS**. This is the only segment type that supports ID3 tags.
  - For a UDP/TS output group, you don't need to set anything.

6. Open the **PID Control** section to see more fields, including **Nielsen ID3**.
7. In **Nielsen ID3**, choose the check box to enable passthrough in this output.

Output locking

You can implement output locking to produce video outputs that are *frame accurate* with each other. Implement output locking to enhance output redundancy, or to implement distributed encoding.

When outputs are locked together, they are frame accurate with each other. Frame accuracy means that two frames with the same timecode are identical in the following ways:

- The same content—the same picture on the video frame.
- The same segment number, manifest data, and so on.
- The same presentation timestamp (PTS).

There are two ways to implement:

- Standard output locking.
- Epoch locking.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

Topics

About output locking and frame accuracy

You can implement output locking to produce video outputs that are *frame accurate* with each other. The frames from several outputs are *locked* together.

Frame accuracy means that two frames with the same timecode are identical in the following ways:

- The same content—the same picture on the video frame.
- The same segment number, manifest data, and so on.
- The same presentation timestamp (PTS).

Output locking use cases

Implement output locking for either or both of these reasons:

- To enhance output redundancy.
- To implement distributed encoding.

Topics

- Use case 2: Distributed encoding
- Use case 3: Distributed encoding with output redundancy

Use case 1: Enhancing output redundancy

You can implement output locking to enhance output redundancy. You can set up output redundancy to work on the same or different Elemental Live appliances:

- If you set up both events on the same appliance, you achieve resiliency if there is a problem with one event. However, if the entire appliance fails, both outputs stop, and there is no fallback.

- If you set up each event on a different appliance, you achieve resiliency if there is a problem with one event or if the entire appliance fails. If the entire appliance fails, the other event (on the other appliance) still provides output to the downstream system.

Adding output locking

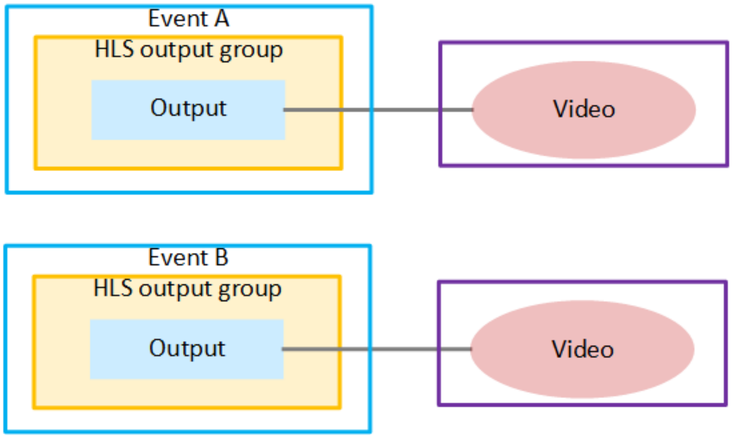
You can add output locking to output redundancy. When you do this, the failover from one output to the other is seamless:

- Without output locking, the two outputs are probably not frame accurate with each other. At a specific timecode, the content in the frame in one output is not identical to the content in the frame in the other output. When the downstream system switches between outputs, there might be a noticeable repetition of a few frames. Or there might be a noticeable discontinuity.
- With output locking, the two outputs are frame accurate with each other. At a specific timecode, the content in both outputs is identical. The downstream system can use the timecode to synchronize the content when it switches from one output to the other. In this way, the switch is seamless. There are no duplicate frames and no missing frames.

Output locking ensures a seamless failover because the outputs are frame accurate with each other. The exact same frame has the exact same timecode. The outputs are locked together.

When you add output locking to an output redundancy setup, you must set up each event on a different appliance.

The following diagram illustrates a typical setup of two events that are a redundant pair.



Use case 2: Distributed encoding

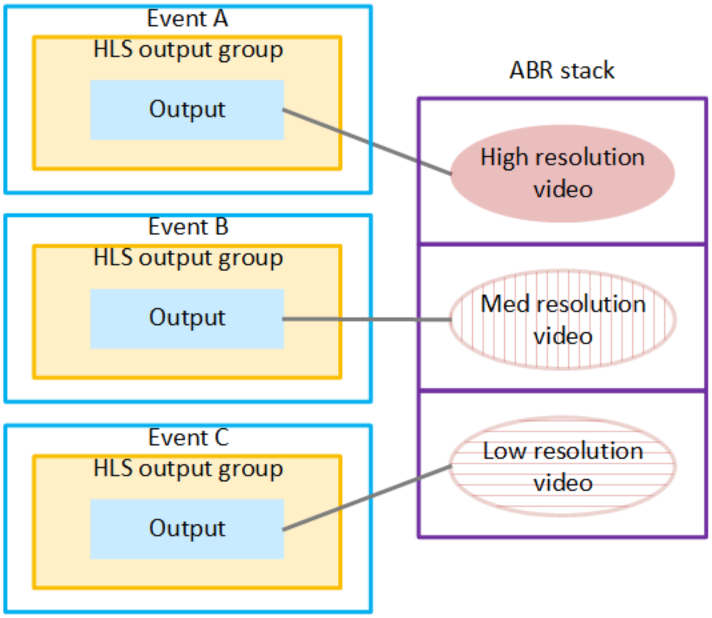
You can set up output locking to implement distributed encoding. A typical use case for distributed encoding is to build an ABR stack that consists of different renditions (resolutions) of the same video content.

You can set up two or more events, each on a separate appliance. All the events handle the same sources, but each event produces different parts of the ABR stack. The source content for all the events is always identical, so the video picture in all the outputs is identical.

You set up all the events so that their outputs are locked together. In this way, all the outputs are frame accurate with each other—the exact same frame has the exact same timecode.

The downstream system receives all of the outputs. The downstream system has been set up to put the ABR stack together. The downstream system can use the timecodes to synchronize the frames. In this way, when the player switches from one rendition to another, the switch occurs with no missing frames and with no duplicate frames.

The following diagram illustrates a setup of several events that together produce the outputs for an ABR stack.



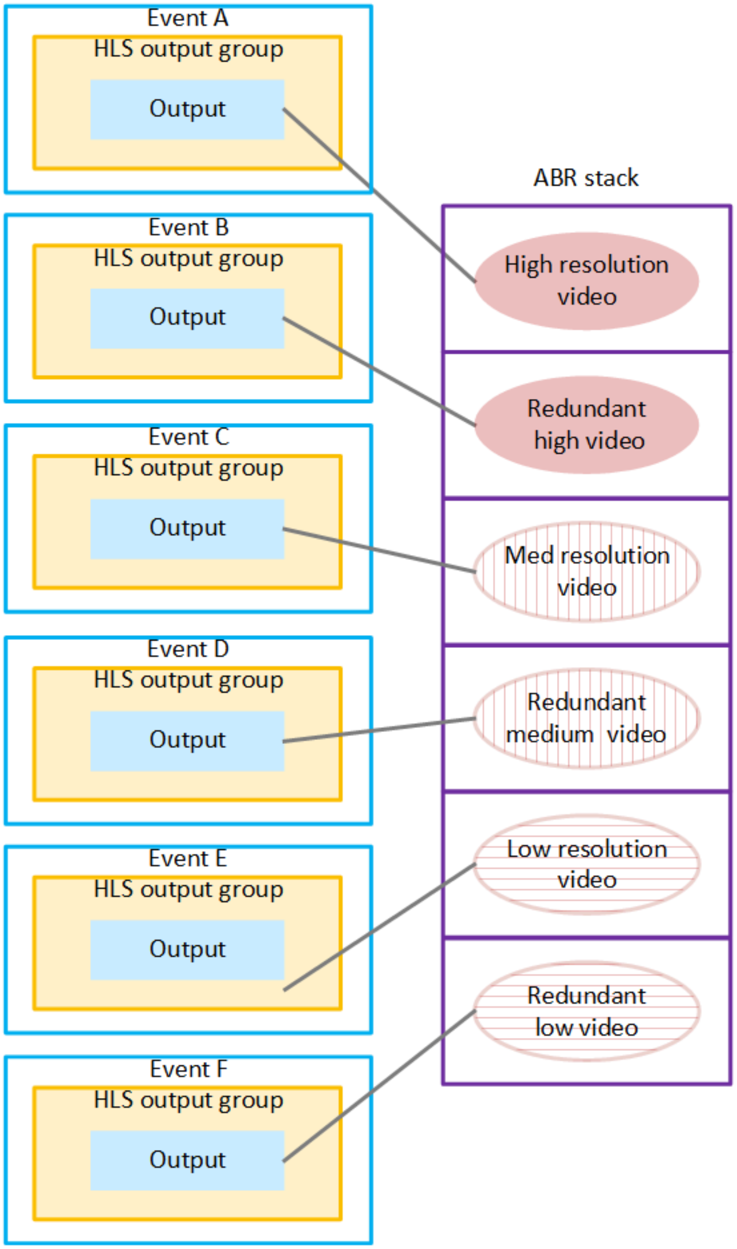
Use case 3: Distributed encoding with output redundancy

You might combine distributed encoding with output redundancy, and also include output locking. For example, you might create three events, each producing a separate output. To add redundancy, duplicate each of those events (outputs) to create *event pairs*. Then to include output locking, make sure that each event in the pair is on a different appliance.

In the following illustration, event A and event D are a pair. The high-resolution video that event A produces is identical to the high-resolution video that event D produces. Similarly, event B and event E are a pair, and event C and event F are a pair.

In this scenario, the downstream system must determine that there are duplicates of each rendition. The downstream system decides which outputs to choose initially, to construct the ABR stack. Then later, if there is a problem with one of the initial renditions of the stack, the downstream system must be able switch to the other rendition.

The following diagram illustrates a setup that combines distributed encoding and redundancy. There are three pairs of redundant events. In each pair, the two events produce the same rendition.



How output locking works

This section shows how Elemental Live locks outputs when an event first starts, and how resync works when an event is running.

Output locking has these key requirements:

- All inputs must have a timecode in the source. You can't set up output locking in events that use the system clock.
- The events must be able to communicate with each other over multicast or unicast.

How the initial lock occurs

When you start or restart all of the events, one event is designated as the leader. Subsequent events are followers. At the first segment boundary, each follower uses its own timecode and the leader's timecode to align the frame that it emits with the frame that the leader is emitting. The events are now all locked together.

How resynchronization works

Elemental Live continually checks the timecode for the same segment in all the locked events. If the segment in one event has a different timecode, then it is considered to be out sync.

Elemental Live resynchronizes the output of an event by inserting a short segment and a presentation timestamp (PTS) discontinuity tag.

Keep this detect-and-correct behavior in mind. Output locking doesn't guarantee that output always staty in sync. Rather, the guarantee is that when a drift occurs, Elemental Live resynchronizes.

Inability to detect and correct

As the event runs, there might be times when the event can't detect and correct. For example, if the current source doesn't include a timecode because it is slate content.

Even in this case, Elemental Live continues to attempt to synchronize. For example, if the content changes so that a timecode is present again in the source, then Elemental Live will resync again.

How epoch locking works

You can set up to use the epoch locking variation of output locking. Or you can set up to use standard output locking.

Epoch locking requires that all of the inputs have a timecode that is in the source. The timecode must show the epoch time in UTC.

> **Note**
>
> We recommend against using epoch locking, for the following reasons:
>
> - Epoch locking requires that the sources have an epoch timecode. It might be difficult to arrange for this timecode to be inserted in the source.

- Epoch locking doesn't work well with ad insertion workflows.

How the initial lock occurs

- When each event starts, the event uses its own source timecode to determine the sequence number and timecode to assign to the first segment and to every following segment:
  - The first sequence number is the number that would be assigned if the event had been running since the start of epoch time. For example, if the event output has segment lengths of 2.02 seconds, the first sequence number is the current epoch time divided by 2.02.
  - The timecode for that sequence number is the current epoch time.
- The event can now predict the sequence numbers and timecodes for each segment:
  - The expected sequence numbers are predictable. They always increment by 1.
  - The expected timecode is predictable. It should increment by 2.02 seconds in each segment.
- The event now locks to that expectation. As the event continues encoding, it continually compares the timecode of each sequence to the expected timecode.
- Every event locks in this way. They all lock to the epoch timecode, which means that in effect they are locked to each other.

How resynchronization works

Each event checks the timecode for the current segment to the expected timecode. If it is different, a drift has occurred. The event will resynchronize by inserting a short segment and a presentation timestamp (PTS) discontinuity tag.

Keep this detect-and-correct behavior in mind. Output locking doesn't guarantee that output never becomes out of sync. Rather, the guarantee is that when a drift occurs, Elemental Live resynchronizes.

Inability to detect and correct

As the event runs, there might be times when the event can't detect and correct. For example, if the current source doesn't include a timecode, perhaps because it is slate content.

Even in this case, Elemental Live continues to attempt to synchronize. For example, if the content changes so that a timecode is present again in the source, then Elemental Live will resync again.

Output locking pools

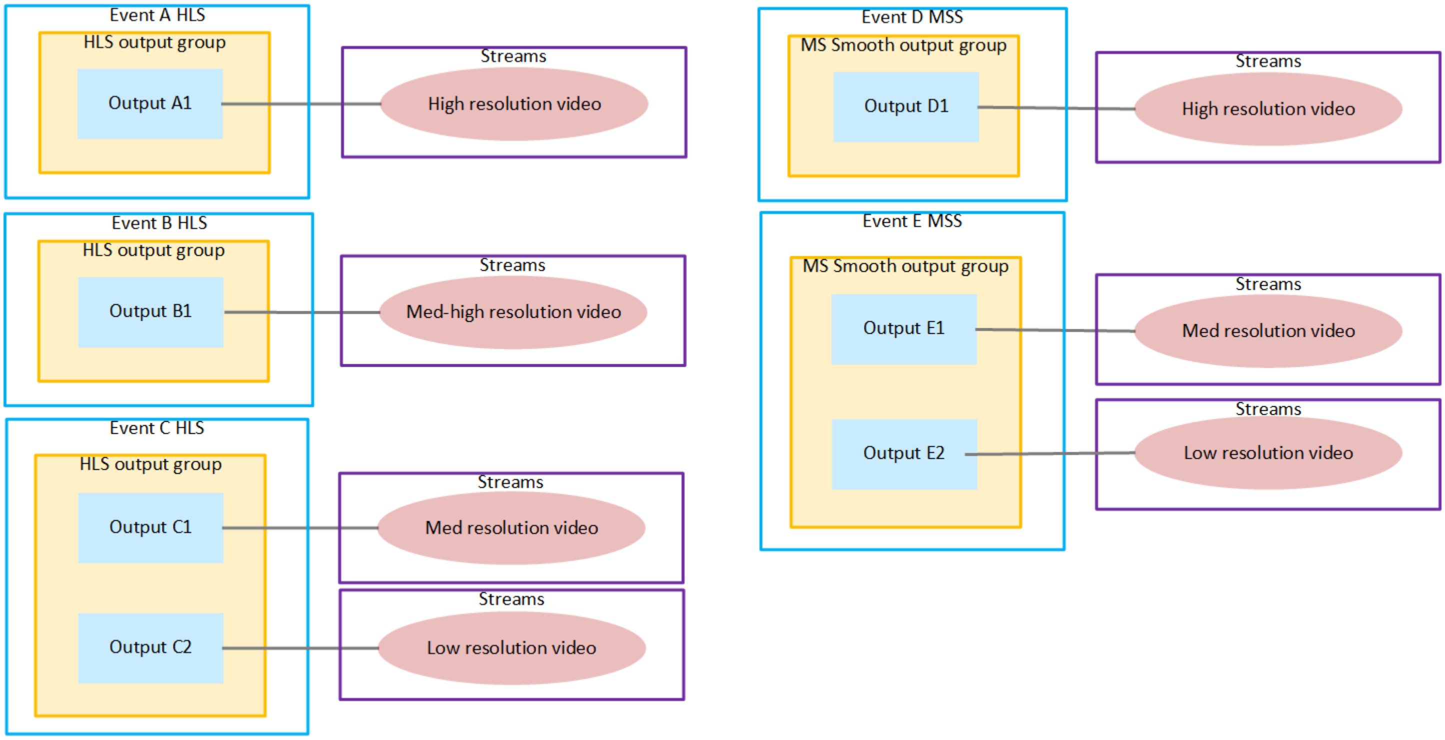
The events that are locked together are in a *pool of locked events*. The outputs in these events are part of a *pool of locked outputs*. The encodes in the locked outputs are part of a *pool of locked encodes*.

If you have a workflow with several types of output groups, it's important to keep track of the pools. If you don't keep track of the pools, you might break the rules for output locking.

The following diagram illustrates the event setup to produce an HLS ABR stack (with four renditions), and a Microsoft Smooth Streaming ABR stack (with three renditions), all from the same source. The following pools exist:

- Events A, B, and C are one pool of locked events for the HLS ABR stack.
- Events D and E are another pool for the Microsoft Smooth Streaming ABR stack.
- Outputs A1, B1, C1, and C2 are one pool of locked outputs.
- Outputs D1, E1, and E2 are another pool of locked outputs.

- The four streams in the HLS events are a pool of locked encodes.
- The three streams in the Microsoft Smooth Streaming events are another pool of locked encodes.



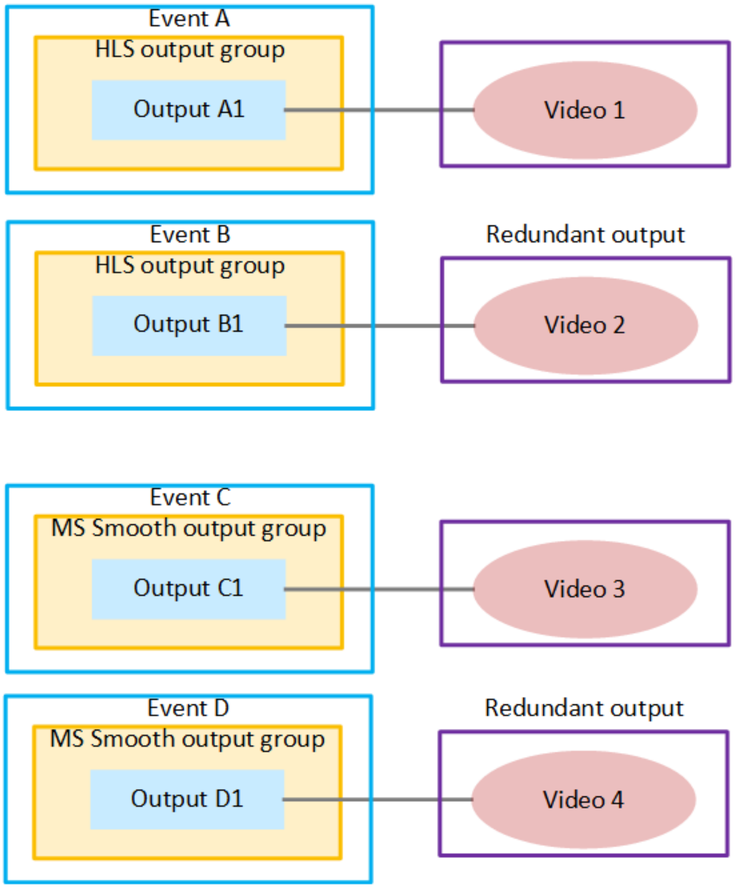
Output locking pairs

In an output redundancy implementation, there are also redundant pairs of events, outputs, and encodes. Encode pairs are typically identical.

If you have a workflow with several types of output groups, it's important to keep track of the pairs. If you don't keep track of the pairs, you might break the rules for output locking.

The following diagram illustrates the event setup to produce a redundant HLS output and a Microsoft Smooth Streaming output, all from the same source. The following pairs exist:

- Events A and B are a redundant pair of events. Events C and D are another redundant pair.
- Outputs A1 and B1 are a redundant pair of outputs. Outputs C1 and C1 are another redundant pair.
- Videos 1 and 2 are a redundant pair of encodes. Videos 3 and 4 are another redundant pair.



Requirements for inputs and outputs

This section describes the requirements for source inputs and outputs, their characteristics, and which work for output locking.

> **Note**
>
> This section refers to *pools* and *redundant pairs*. For an explanation of pools, see [the section called "Output locking pools"](). For an explanation of pairs, see [the section called "Output locking pairs"]().

Topics

- Supported input types
- Input requirements
- Supported output types

- Output encode requirements
- Output locking and SCTE 35

Supported input types

Elemental Live supports output locking with the following types of inputs:

- HLS network inputs. Output locking isn't supported with HLS file inputs.
- SDI sources: SDI inputs, 4K Quadrant SDI inputs, 4K 2SI inputs,
- TS sources: RTP network inputs, SMPTE 2022-6 inputs, SMPTE 2022-7 inputs, SRT inputs, UDP network inputs.
- RTMP inputs.
- RTSP inputs.

Input requirements

For output locking to be successful, the video sources must meet strict requirements.

Input timecode

The inputs must have a timecode in the source. If the timecode is missing from the source, the event will fail to start.

Different types of sources support different types of timecode:

- HLS live sources: Embedded timecode.
- Inputs that are SDI sources: Embedded or LTC timecode. All the inputs in one event must have the same timecode source. The inputs can't switch from embedded to LTC. All inputs across the pool of locked events must have the same timecode.
- Inputs that are transport stream sources: Embedded timecode.

The timecode with standard output locking must be HH:MM:SS:FF format.

The timecode with epoch locking must be epoch time in UTC.

> **Note**
>
> Both output locking and epoch locking require a timecode in every source.

Video is required

Output locking isn't supported in an audio-only event.

Video characteristics

The sources for all 7of the events must be identical. Specifically, the outputs won't be frame accurate if the following video characteristics vary across the sources:

- Frame rate. The frame rates in the different sources can have different frame rates that are whole number multiples of each other. For example, 30 fps and 60 fps.
- Scan type (progressive versus interlaced).

Note that these are the only characteristics that must be identical in all sources. Other characteristics, including resolution and GOP structure (I-frame, B-frame, and P-frame patterns) can be different.

Supported output types

Elemental Live supports output locking with the following types of outputs:

- HLS output group
- MS Smooth output group
- UDP/TS output group

> **Note**
>
> All output groups in the event must implement output locking. If an output group can't implement output locking, you can't include it in the pool of locked events.
> Even when an output group doesn't support output locking, you can't include it in the output locking event. For example, the event can't include a DASH output group.

Output encode requirements

For output locking to be successful, the video outputs must meet strict requirements.

Video codec

The video encodes must use H.264 or H.265.

Within a pool of locked encodes, the codecs can be different, as long as they're always H.264 or H.265.

In a redundant pair of encodes, the codecs must be the same.

Frame rate

There are two sets of rules for the frame rate:

- Rule for the frame rate conversion from the source to the output encode.
- Rule for the frame rates in all output encodes in the pool of events.

Rule 1: Conversions from the source to the output encode in the same output

Simple conversions are allowed. The conversion between the input frame and the desired output frame rate must be *simple*, which means that one of these statements must apply:

- The output encode frame rate must be a whole number multiple of the source frame rate. For example, the source frame rate might be 30 FPS, and the output encode frame rate might be 60 FPS.
- The source frame rate must be a whole number multiple of the output encode frame rate. For example, the source frame rate might be 60 FPS, and the output encode frame rate might be 30 FPS.

With these rules, it is possible for the frame rates to be integers. For example, if the source frame rate is 29.97 FPS and the output encode frame rate is 59.94 FPS.

Complex conversions aren't allowed. The following are examples of *complex* frame rates. You can't use the input if combinations like these apply to your channel:

- Source FPS is 59.4, output FPS is 60.
- Source FPS is 45, output FPS is 60.

Rule 2: Frame rates in the outputs in the output pool

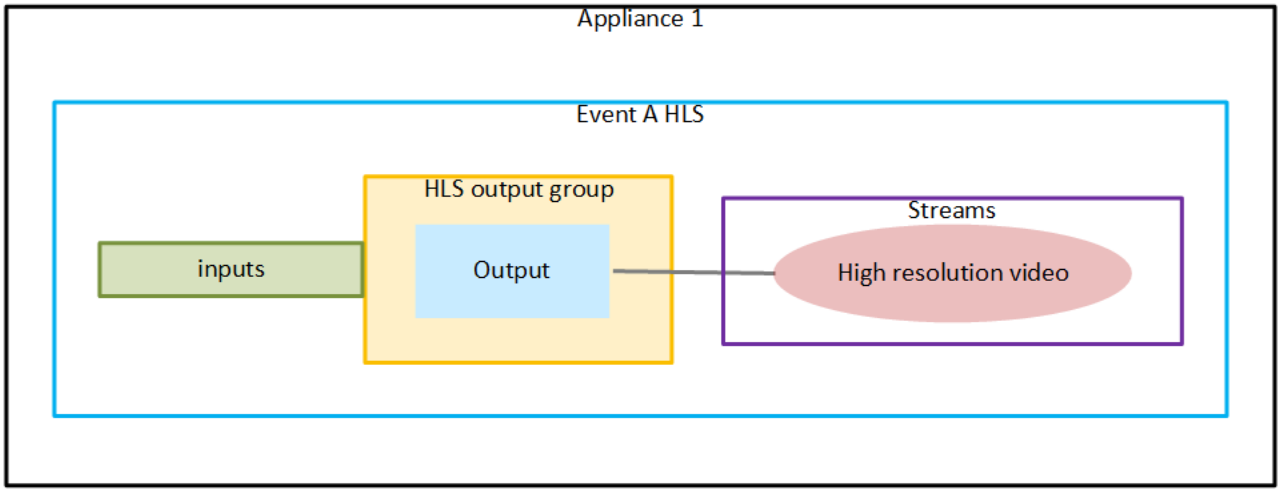
The frame rates in the output encodes in the pool of locked outputs can all be different. However, they must be whole number multiples of each other.

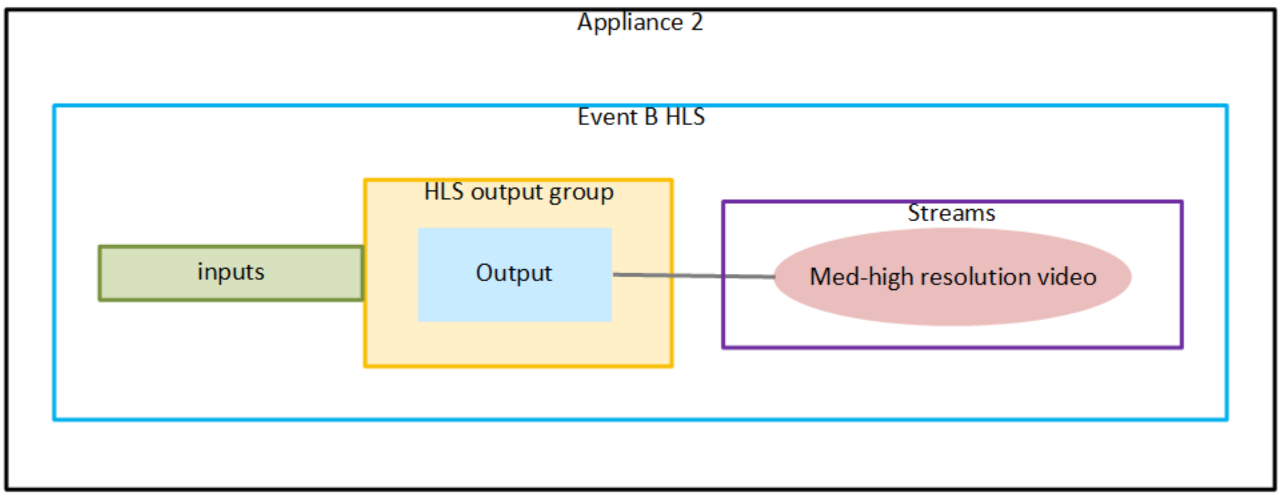
Example

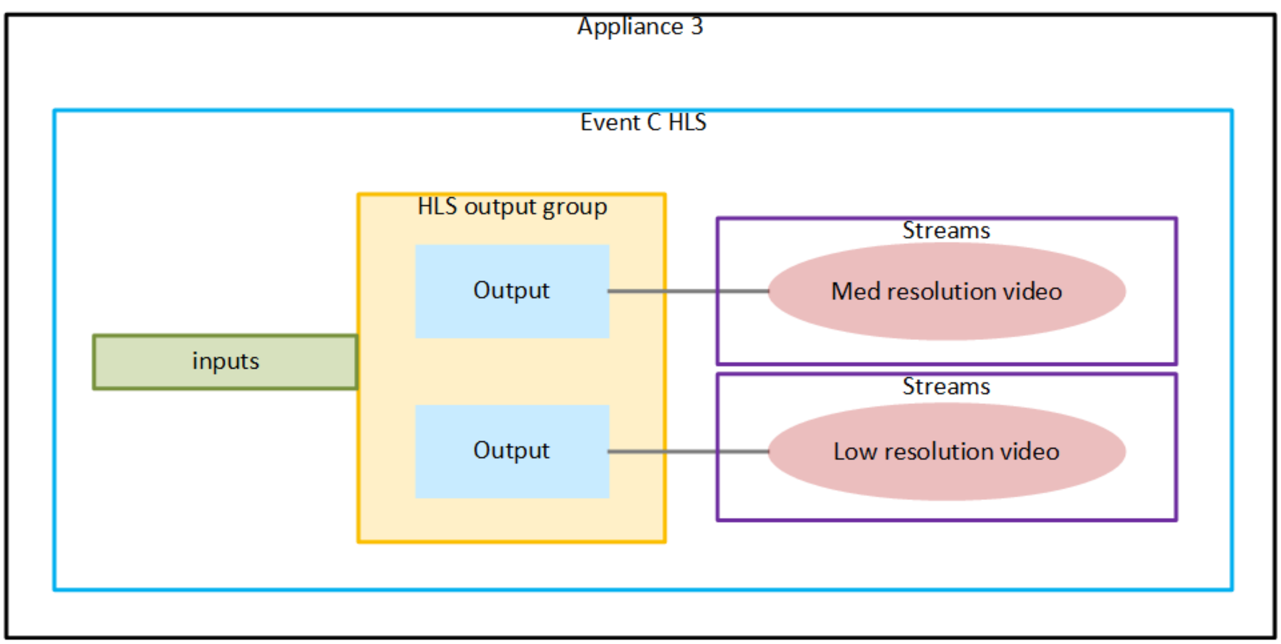
In the following diagram, all of the inputs are from the same source. Therefore, they have the same frame rate.

Rule 1: In each event, each video encode (the oval inside the stream) must have a frame rate that is a simple conversion of the source frame rate.

Rule 2: All the encodes can have a different fran rate, but they must be whole number multiples of each other.


Appliance 1
Event A HLS
HLS output group
Streams
inputs
Output
High resolution video


Appliance 2
Event B HLS
HLS output group
Streams
inputs
Output
Med-high resolution video


Appliance 3
Event C HLS
HLS output group
Streams
Output
Med resolution video
inputs
Streams
Output
Low resolution video

Output locking and SCTE 35

Epoch locking and SCTE 35

To include SCTE 35 messages in the outputs, you can't implement epoch locking. The presence of SCTE 35 messages is not compatible with the segment length requirements for epoch locking.

If the inputs contain SCTE 35 messages, make sure that you don't pass them through to the output.

Inserting SCTE 35 messages via the REST API

You can implement standard output locking if you want to include SCTE 35 messages in the outputs. The messages can be in the source and/or added when the event is running:

- The SCTE 35 messages can be messages that you pass through from the source.
- Or they can be messages that you insert into a running event, using the REST API.

  In this case, make sure to include a time buffer between the time that you enter the message and the time for that message. In other words, don't enter an *immediate* start time for the cue out or the cue in.

  A message with an immediate start can compromise the frame accuracy of the pool of locked outputs.

Step 1: Design the workflow

When you implement output locking, there are special considerations in your design of the inputs, output groups, outputs, and output encodes (video streams) in the events.

Topics

- Design the inputs
- Design the outputs
- Example of a workflow

Design the inputs

When you design a workflow that implements Elemental Live output locking, you must obtain information about the sources.

Topics

Determine the resources

Identify the number of sources, output encodes, and events that you need.

For output redundancy workflows

For output redundancy, you need two sources, two output encodes, and two events. Each event uses one input and produces one output encode.

For distributed encoding workflows

For distributed encoding, calculate the numbers as follows:

- The number of output encodes: You need one output encode for each video rendition in the ABR stack.
- The number of events: Decide how many output encodes each Elemental Live appliance can handle, based on the density capabilities of the appliance. An appliance might be able to produce only one output encode if that encode is a 4K rendition. However, an appliance might be able to produce several lower-resolution output encodes (renditions).

  When you know the number of appliances, you know the number of events—one event per appliance.
- The number of sources: You need one source for each event.

Decide how to produce the sources

Decide how you will produce the sources:

- You might produce the streams for each event using a different contribution encoder for each event.

- Instead, you might produce the sources once, then use a video router upstream of Elemental Live to product identical streams.

Make sure that the sources are identical.

Obtain information about the sources

Obtain the following information:

- The characteristics of the video.
- The type of timecode that each source includes.

Design the outputs

When you design a workflow that implements Elemental Live output locking, you must decide which output group types you will produce, design the outputs in the output groups, and decide on the video encoding characteristics of the output type or types.

This section covers considerations that affect your design.

Topics

Number of outputs in an output group

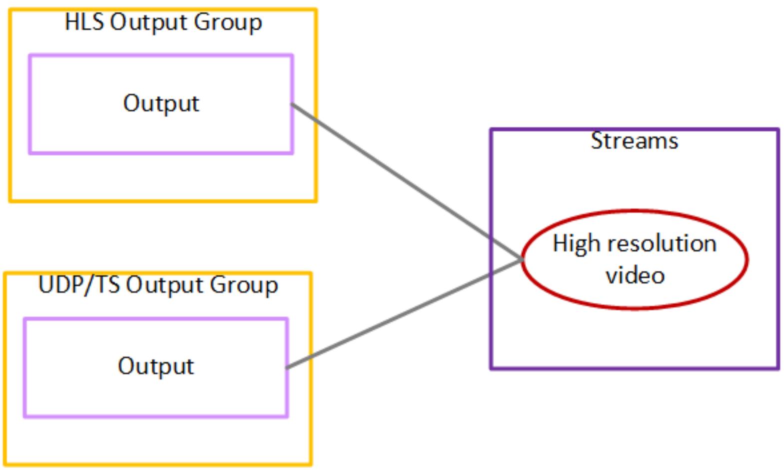
For distributed encoding, there will be many cases where one appliance can produce only one output encode. To set up these encodes, create an output group (for example, create an HLS output group), and create one output in that output group.

There might also be cases where you have determined that one appliance can produce several one lower-resolution output encodes. To set up these encodes, create an output group (for example, create an HLS output group), and create two outputs. Each output contains one of the encodes.

Number of encodes: video encode sharing

If you have multiple output group types in one event, several different output groups might be able to use the same video encode.

For example, if you're producing an HLS package and a UDP/TS package, and both packages are encoded with H.264, you might be able to create one video encode that the two output groups share. This sharing reduces the compute demands on the appliance. To share the encodes, create an output in one output group and an output in the other output group. Create one video stream. Set up both outputs to use the same encode.



Encodes across the locked events

You must set up the video encodes in the locked events as follows:

- For an output redundancy implementation, the pair of encodes in the same type of output group must be identical. For example, the encode in the output must be identical to the encode in the redundant output.
- For a distributed encoding implementation, each pair of encodes must be identical, so that the main ABR stack and the redundant ABR stack are identical. For example, you might have one ABR stack consisting of encodes A1, B1, C1, D1, and another ABR stack consisting of encodes A2, B2, C2, D2. Encode A1 must be identical to encode A2, B1 identical to B2, and so on.

For an explanation of *pairs of encodes*, see the section called "Output locking pairs".

Features in an output

Some features, such as static graphic overlay, can apply within an individual portion of an event.

Make sure that you apply the features in the same way in every locked event. For example, if you implement the static overlay in all outputs in one event, then make sure that you set it up on all outputs in the other event. For more information, including diagrams, see the section called "Insertion options".

Motion graphic overlay

We strongly recommend against combining motion graphic overlays and output locking. Frame accuracy can't apply to motion graphic overlays that are applied to the video content.

Example of a workflow

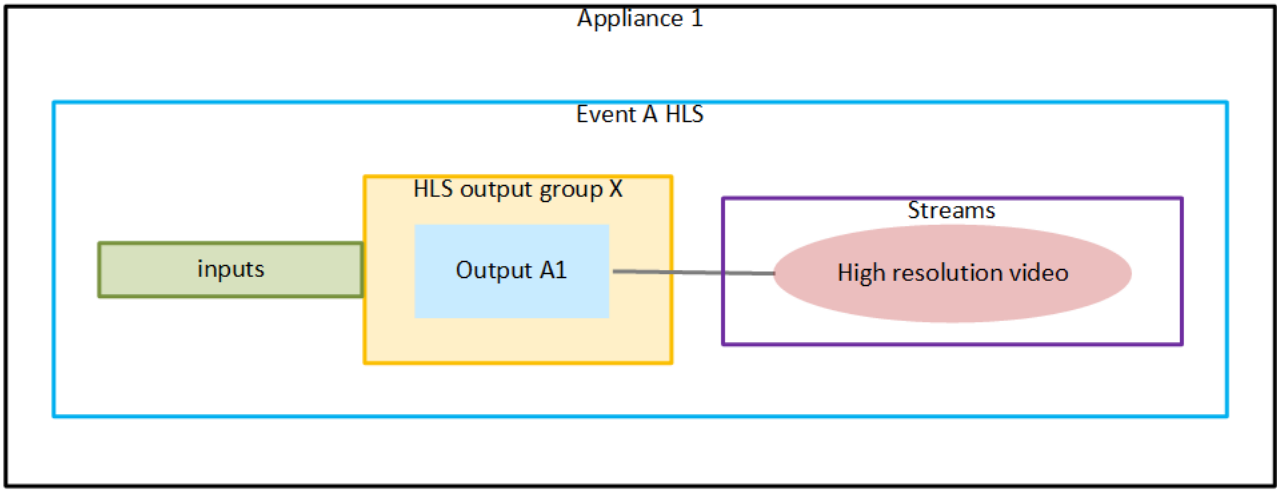
This section provides an example of the design of an output lock workflow that implements distributed encoding.

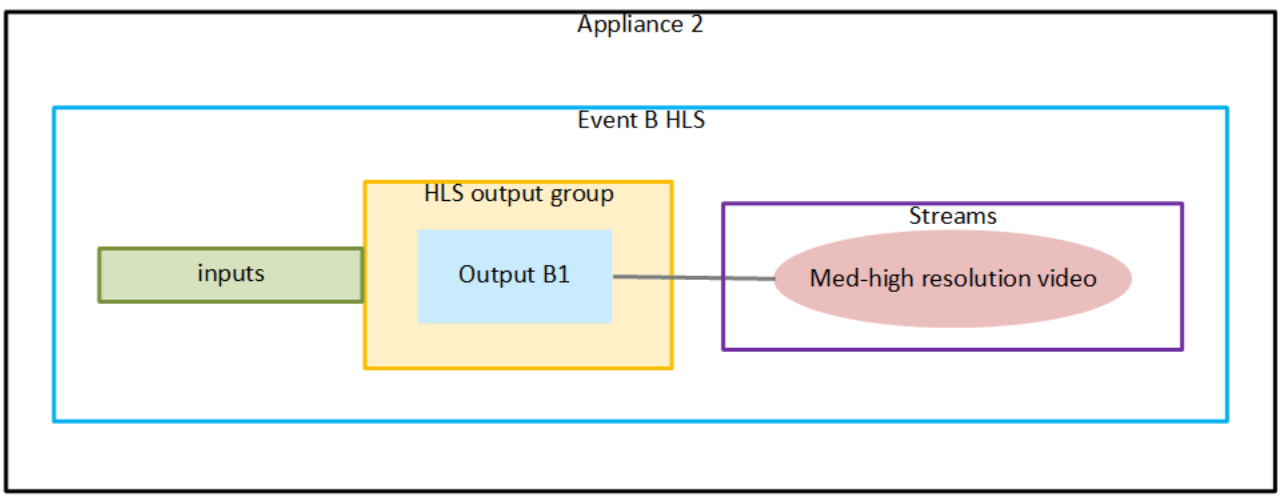
> **Note**
>
> This example refers to *pools*. For an explanation of pools, see the section called "Output locking pools".

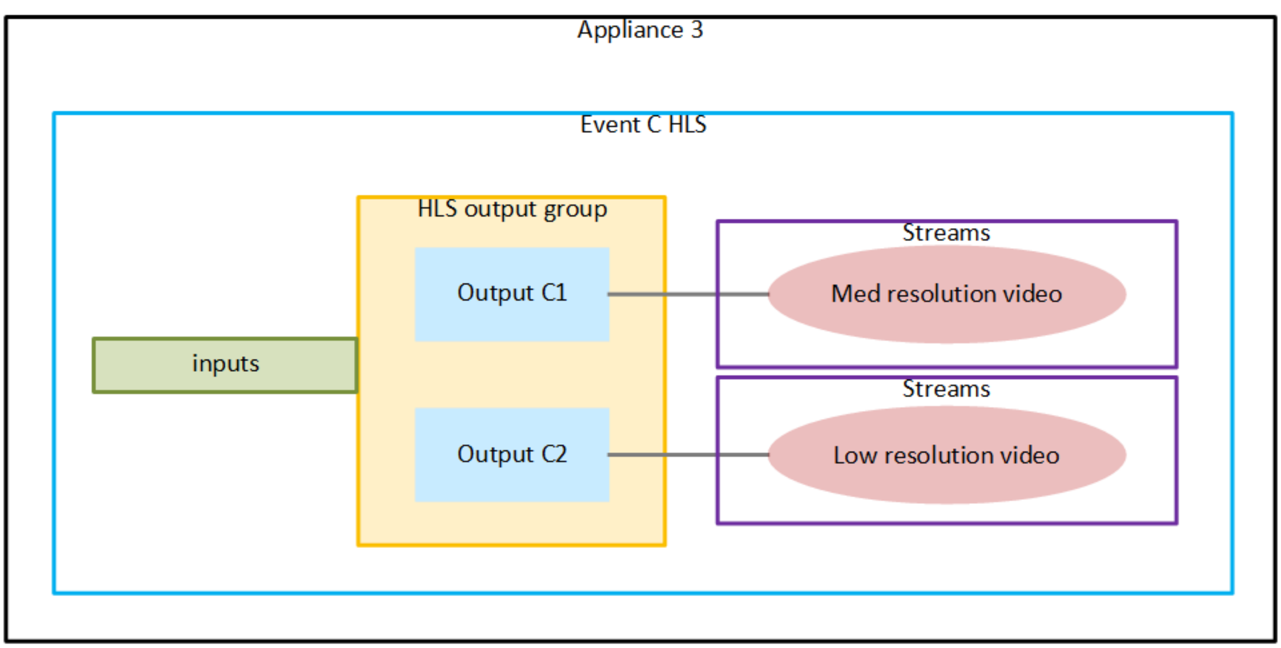
The following example workflow has two output packages: HLS and MS Smooth. Notice how the ABR stacks are divided into several events within each package. This division is driven by the processing demands on the encodes. Notice that when two outputs are on the same appliance, they can share the same output group.

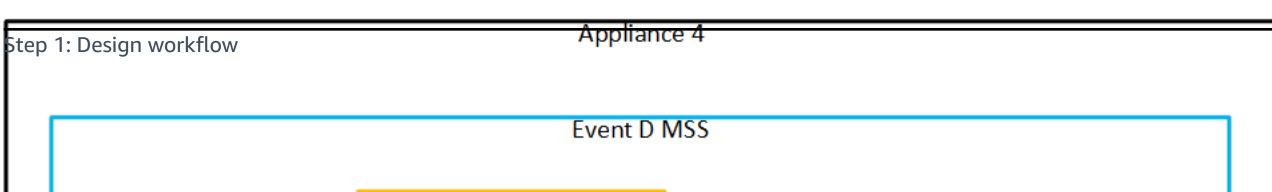
Pay attention to the pools that are implied by this design:

- There are two event pools for five events. One pool contains events A, B, and C. One event contains events D and E.
- There are two output group pools. One pool contains the three HLS output groups, shown in yellow. The other contains the two MS Smooth output groups, shown in striped yellow.
- There are two output pools. One pool contains four HLS outputs, shown in blue. The other contains three MS Smooth outputs, shown in striped blue.
- There are two encode pools. One pool contains four HLS encodes, shown in red. The other contains three MS Smooth outputs, shown in striped red.


Appliance 1
Event A HLS
HLS output group X
inputs
Output A1
Streams
High resolution video


Appliance 2
Event B HLS
HLS output group
inputs
Output B1
Streams
Med-high resolution video


Appliance 3
Event C HLS
HLS output group
Output C1
Streams
Med resolution video
inputs
Streams
Output C2
Low resolution video


Appliance 4
Event D MSS

Step 2: Set up inputs in the events

After you have identified the sources and ensured that they support output locking, you can set them up in the event as inputs.

> **Note**
>
> This section refers to *pools*. For an explanation of pools, see the section called "Output locking pools".

To set up the inputs

1. Create all of the inputs that you have identified. Create the same inputs in each event, and enter them in the same order in each event.
2. Complete all fields that apply to the input type.

   In the **Input – Video Selector** section of the event: Set the following fields as specified in the following table.

Field name	Instruction	Notes
Input Name	In one event, enter a different name for each input. Use that name for the same input in the entire pool of locked events.	This field is optional. However, output locking works best if inputs have names.
Timecode Source	For SDI inputs, choose **Embedded** or **LTC**, depending on the timecode in the input. For other inputs, choose **Embedded**.	If an input doesn't have a timecode, you can't use it in the event.

Step 3: Set up the global controls

This section shows how to set up global controls to enable output locking and epoch locking in an Elemental Live event. You must set up the controls that apply to the entire event—the timecode and the communications mechanism.

> **Note**
>
> This section refers to *pools* and *redundant pairs*. For an explanation of pools, see the section called "Output locking pools". For an explanation of pairs, see the section called "Output locking pairs".

Topics

- Set up the timecode
- Enable output locking

Set up the timecode

In the **Timecode Configuration** section that appears after the **Inputs** section of the event: Set the fields as specified in the following table. This setup configures the timecode for use in all outputs.

Field name	Instruction	Notes
Source	Choose any value, but choose the same value in all events in the pool.	This timecode is the timecode for the outputs. It is inserted in all the outputs in the pool of locked events. The downstream system can use this timecode to determine that a frame from one event corresponds to the frame from another event.
Sync Threshold	Deselect this field.	

Field name	Instruction	Notes
Require Initial Timecode	Select the check box to ensure that the first source does in fact have a timecode.	If you select this check box, the event will fail to start if the first input has no embedded timecode. The event will also fail to start if you stop and restart the event, and the input that is active when you restart has no embedded timecode.

Enable output locking

You must enable output locking.

Make sure that you perform the same setup in all the events in the pool.

To set up to use multicast for standard output locking

To implement standard output locking, the events you are locking together must be able to communicate with each other over multicast or unicast. You can use multicast if you are locking together two events, or if you are locking together several events.

1. In the **Global Processors** section of the event set the **Output Locking** field to **On**.
2. Set the fields as shown in the following table.

Field name	Instruction	Notes
Epoch locking	Deselect this field.	
Multicast	Select this field.	
Address **Port** (optional) **Interface** (optional)	These fields appear only if you have selected **Multicast**. Enter the multicast address of any server. Enter the	The events communicate with each other through this address.

Field name	Instruction	Notes
	identical address, port, and interface across all events that you want to lock together. For more information, see the tooltips on the Elemental Live web interface.	

To set up to use unicast for standard output locking

To implement standard output locking, the events that you're locking together must be able to communicate with each other over multicast or unicast. Unicast is a point-to-point protocol, so you can use unicast only if you're locking together two events.

- In the **Global Processors** section of the event set the **Output Locking** field to **On**.

  Set the fields as shown in the following table.

Field name	Instruction	Notes
Epoch locking	Deselect this field.	
Multicast	Deselect this field.	
Address **Port** (optional) **Interface** (optional)	These fields appear only if you have deselected **Multicast**. Enter the address where the other event is listening . See the tooltips on the Elemental Live web interface for more informati on.	This event in the event pair uses this address to send a message to the other event.

Field name	Instruction	Notes
	If you have more than one locked event on the same appliance, make sure that you assign a unique port to each event. Two different events can't listen on the same port. For an example of two locked events on an appliance, see **Appliance 3**in the diagram in the section called “Example”.	
Receive Port (required) **Receive Interface** (optional)	These fields appear only if you have deselected **Multicast**. Enter the address where this event is listening. See the tooltips on the Elemental Live web interface for more information. If you have more than one locked event on the same appliance, make sure that you assign a unique port to each event. Two different events can't listen on the same port.	The other event uses this address to send messages to this event.

To set up for epoch locking

1. In the **Global Processors** section, set the **Output Locking** field to **On**.
2. Select **Epoch Locking**. The values of other fields are ignored.

Step 4: Set up the output groups and outputs

This section shows how to set up output groups and outputs in an Elemental Live event that implements output locking.

> **Note**
>
> This section refers to *pools*. For an explanation of pools, see the section called "Output locking pools".

After you set up all of the events that you have identified, use the following topics to set up the outputs.

Topics

Setting up an HLS output group

This section shows how to set up an HLS output group to implement Elemental Live output locking.

1. Complete the fields in the **Apple HLS Settings** section of the output group. Set the fields that are listed in the following table as specified in the table.

Field name	Instruction
Custom Group Name	Enter the same name across all events in the pool.
Minimum Segment Length	If you want, enter a value. Output locking works with a minimum length.

2. Set other fields in the **Apple HLS Settings** section to suit your workflow.
3. Set the fields in the outputs to suit your workflow.

Setting up an MS Smooth output group

This section shows how to set up an MS Smooth output group to implement Elemental Live output locking.

1. Go to the **MS Smooth Settings** section of the output group. Set the fields that are listed in the following table as specified in the table.

Field name	Instruction
Custom Group Name	Enter the same name in all events in the pool.
Fragment Length	Enter the same number in all events in the pool.
Use Event ID	If the pooled events are configured to publish to a single publishing point, you can deselect this field. If you don't deselect this field, then when one encoder stops, the publishing point may stop accepting requests from the other events in the pool. If the pooled events publish to different publishing points, you can leave this field selected.
Send EOS	If the pooled events are configured to publish to a single publishing point, you can deselect this field. If you don't deselect this field, then when one encoder stops, the publishing point may stop accepting requests from the other events in the pool. If the pooled events publish to different publishing points, you can leave this field selected.

Field name	Instruction
Send Delay	Complete as desired. For details, see the tooltip on the Elemental Live web interface.

2. Set other fields in the **MS Smooth Settings** section to suit your workflow.
3. Set the fields in the outputs to suit your workflow.

Setting up a UDP/TS output group

This section shows how to set up a UDP/TS output group to implement Elemental Live output locking.

1. Go to the **UDP/TS Output Group** section of the event. Set the fields as follows.
   - **Custom Group Name**: Enter the same name across all events in the pool.
   - Set other fields to suit your workflow.
2. Go to each output. In the **Transport Stream Settings** section, set the fields as specified in the following table.

Field name	Instruction
Segmentation Markers	Output lock requires that a UDP/TS output have segmentation markers. Always choose **EBP Cablelabs**. Other options aren't valid for output locking. This option adds Encoder Boundary Point informati on to the adaptation field in conformance with OpenCable specification OC-SP-EBP-I01-130118.
Segmentation Time	Required. This field ensures that all of the outputs in the pool synchronize continually, not just when the events first start.

Field name	Instruction
Fragment Time	Required.

3. Set other fields in the outputs to suit your workflow.

Step 5: Set up the video encodes

This section shows how to set up video encode parameters in an event that implements Elemental Live output locking.

> **Note**
>
> This section refers to *pools*. For an explanation of pools, see the section called “Output locking pools”.

After you have created the output groups and outputs, follow these guidelines to create the video encodes.

Set up the output encodes (video streams) in the usual way:

1. In each output group, create as many outputs as you planned for that output group. When you add an output, Elemental Live creates a new stream in the **Streams** section.
2. Set the following fields in the **Video** tab as specified.

Field name	Instruction	Notes
Codec	H.264 or H.265	You can use a combination of H.264 and H.265 in the pool of outputs.
Framerate	Enter a value. We strongly recommend not to choose **Follow source**	For information about the rules for frame rate, see the section called “Output encode requirements”. If you ignore these rules, the encodes in the pool of

Field name	Instruction	Notes
		outputs won't be frame accurate with each other.

3. You can set other fields in the **Video** section, the fields in the **Audio** section, and the fields in the **Captions** tab to suit your workflow.

SCTE-35 and SCTE-104 message processing in Elemental Live

You can use Elemental Live to manipulate the SCTE-35 messages in a TS input and to manipulate the SCTE-104 messages in an HD-SDI input. You can also use Elemental Live to remove or include the cueing information conveyed by SCTE messages in the output streams (video, audio, closed captioning, data) and any associated manifests.

You set up SCTE message processing instructions in the Elemental Live event. This guide describes how to perform this set up.

Note that Elemental Live does not support processing of manifests that are present in the input. The information in these manifests is not ingested by Elemental Live and is not included in the output or the output manifest.

About this guide

SCTE messages might convey DPI cueing information for ad avails and for other non-ad-avail messages such as programs and chapters.

This guide covers both Event Signaling and Management (ESAM) and non-ESAM processing of messages.

Assumptions

You should be familiar with the SCTE-35 and SCTE-104 standards and optionally with the SCTE-67 standards and how the input you encode implements these standards. You should be familiar with profiles and with managing Elemental Live events. To use the REST API features, you should be familiar with interacting with Elemental Live through the API.

Topics

- Manifest decoration
- Ad avail blanking and blackout
- Passthrough or removal of SCTE messages
- SCTE-35 message insertion into currently running events
- POIS conditioning
- Setting up using the REST API
- Example manifests for Apple HLS

Eligible messages and streams

Elemental Live can extract SCTE-35 messages and SCTE-104 messages from input sources.

The following table specifies the sources (and their corresponding input types) that Elemental Live supports for each message type.

Message type	Source type	Elemental Live Input type
SCTE-35	HLS	HLS network input
	MPTS over RTP or UDP	Network input
	Transport stream	Secure Reliable Transport (SRT)
	Transport stream over RTP or UDP	Network input
	Transport stream over RTP with SMPTE 2022-7 redundancy	SMPTE 2022-7 network input
SCTE-104	SDI	An SDI or Quadrant interface
	SDI	Interleave 4K (HD-2SI)
	SMPTE 2110-04 over RTP or UDP	SMPTE 2110

> **Note**
>
> SCTE-104 messages are converted to SCTE-35 messages early in the processing, so this guide uses "SCTE-35 messages" to refer to both SCTE-35 messages and SCTE-104 messages.

SCTE-35 and SCTE-104 message processing options

The options

The processing possibilities for SCTE messages include the following.

Automatically convert SCTE-104 messages

SCTE-104 messages in the input are automatically converted to SCTE-35 messages during processing of the input. No setup is required for this processing.

Line up SCTE Messages to use blanking and blackout of output content

The "cue out" and "cue in" instructions in SCTE-35 messages line up with specific content in the video, audio, and closed captions streams. You can set up to blank out this content in the output.

- The content for ad avails is blanked out using the Ad avail *blanking* feature.
- The content for other messages is blanked out using the *blackout* feature.

Your desired behavior must be set up in the event or profile.

SCTE-35 passthrough

You can include SCTE-35 messages in the output data stream in any TS output. Your desired behavior must be set up in the event or profile.

Manifest decoration

You can set up SCTE messages with manifest decoration with these options:

- HLS outputs can be set up so that their manifests include instructions that correspond to the original SCTE-35 message content.
- MS Smooth outputs can be set up to include these instructions in the sparse track.

- RTMP outputs can be set up to include these instructions in the data stream (because RTMP outputs do not have manifests).

Conditioning by a POIS

Optionally, SCTE-35 messages (including those converted from SCTE-104) can be diverted to a POIS server for ESAM conditioning. This conditioning is in addition to all the other processing (manifest decoration, blanking and blackout, and passthrough).

POIS and ESAM conditioning are described in POIS conditioning.

Insert SCTE-35 messages into content during an event

The Elemental Live REST API includes commands to insert SCTE-35 messages into the content while the Elemental Live event is running.

You can insert using the Elemental Live REST API. Elemental Live makes no distinction between these messages and those that were in the original input. So, for example, if manifest decoration is enabled and you insert a message during encoding, that new message is represented in the manifest.

Default behavior of SCTE-35 and SCTE-104 messages

The default handling of SCTE-35 and SCTE-104 messages in Elemental Live includes the following:

- No manifest decoration: You do not convert any SCTE-35 messages to event information in any output manifests or data streams.
- No passthrough: You do not pass through SCTE-35 or SCTE-104 messages in any data stream outputs.
- No blanking: You do not blank out video content for any events: you leave the content as is.

If you desire the default behavior as described , you have the information you need for processing SCTE messages and do not need to read the remainder of this guide.

About timecode configuration and timers

The event or profile includes a timecode configuration field that identifies the source for timecode stamps to be inserted in the output. The source for these stamps might be a timecode embedded in the input or might be a source external to the input (for example, the system clock or a specified time). The Timecode Config field is just after the Inputs section on the event page of the web interface.

Before starting the transcode, the transcoder gets the timecode from the source.

After the initial sampling, Elemental Live calculates the timecode of every frame and attaches it to the output. The timecode stops advancing if there is no output. So, for example, if the input fails at 10:55:03:009, the timecode at that point is 10:55:03:009. If the input restarts 3 seconds later, the timecode of the next frame might be 10:55:03:012. The timecode will *not* be 10:55:**06**:009.

Given the importance of accurate times with SCTE-35 messages, ensure that the **Network Time Protocol (NTP)** is configured on the node.

Scope of processing depending on outputs

The following table summarizes which options apply to which kind of output. Following the table are details for each output type.

Output	Passthrough in TS outputs	Manifest decoration	Blanking
Archive outputs with MPEG-2 as the container	Include all the original SCTE-35 messages. Convert any SCTE-104 messages to SCTE-35 messages (of the same message type) and include in the TS output.	Not applicable	Applicable
Archive outputs with other containers	Not applicable	Not applicable	Applicable
HLS	Include all the original SCTE-35 messages.	Decorate the HLS manifest with one or more of the following types of ad markers:	Applicable

Output	Passthrough in TS outputs	Manifest decoration	Blanking
	Convert any SCTE-104 messages to SCTE-35 messages (of the same message type) and include in the TS output. Note that, with HLS, you either implement both manifest decoration and passthrough or you implement neither.	• Adobe • Elemental • SCTE-35 enhanced.	
DASH	Not applicable	Not applicable	Applicable
MS Smooth	Not applicable	Include information about the SCTE-35 event in the sparse track.	Applicable
SMPTE 2110			
RTMP	Not applicable	Include one or more of the following types of ad markers in the RTMP datastream: • OnAkamaiAdPod • OnCuePoint • OnCuePoint SCTE-35 • OnUserDataEvent	Applicable

Output	Passthrough in TS outputs	Manifest decoration	Blanking
UDP/TS	Include all the original SCTE-35 messages. Convert any SCTE-104 messages to SCTE-35 messages (of the same message type) and include in the TS output.	Not applicable	Applicable

Topics

- Archive output with MPEG-2 container
- Archive output with other containers
- Apple HLS output
- DASH output
- MS Smooth output
- Adobe RTMP output
- SMPTE 2110 output
- UDP/TS output

Archive output with MPEG-2 container

A transport stream (TS) in an MPEG-2 container supports passthrough of the SCTE-35 messages, but it does not support creation of a manifest. Therefore, usable options are:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Enabled	Yes or No	Not applicable	Yes or No	Turns on passthrough

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				of SCTE-35 messages. In this case, you could also insert more SCTE-35 message if desired. You could also implement blanking and blackout.

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Disabled	No	Not applicable	No	Turns off passthrough in order to remove SCTE-35 messages from the video stream. Do not insert extra messages: they simply get stripped out of the output. Do not implement blanking or blackout. Choose this option only if, in a downstream system, you do not want to replace video that was originally marked by cues.

Archive output with other containers

Other archive outputs do not support passthrough of the SCTE-35 messages or manifest decoration. Therefore, the only workable option is the default behavior:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Not applicable	No	Not applicable	No	Removes SCTE-35 messages from the output. The manifest is not decorated. Do not implement blanking or blackout because, without SCTE-35 messages in the video stream and without manifest decoration, it is impossible to find these blanks and blackouts programma tically.

Apple HLS output

Apple HLS output supports both passthrough of the SCTE-35 messages and manifest decoration. In fact, with HLS outputs, passthrough and manifest decoration are either both enabled or both disabled. Therefore, workable options are:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Enabled	Yes or No	Enabled	Yes or No	Turns on passthrough of SCTE-35 messages and manifest decoration. In this case, you could also insert more SCTE-35 message if desired. You could also implement blanking and blackout.
Disabled	No	Disabled	No	Turns off passthrough in order to remove SCTE-35 messages from the video stream. Turns off manifest decoration. Do not insert extra messages: they simply get stripped from the output. Do not implement

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				blanking or blackout. Choose this option only if, in a downstream system, you do not want to replace video that was originally marked by cues.

DASH output

DASH ISO output does not support passthrough of the SCTE-35 messages or manifest decoration. Therefore, the only workable option is the default behavior:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Not applicable	No	Not applicable	No	Removes SCTE-35 messages from the output. The manifest is not decorated. Do not implement blanking or blackout because, without SCTE-35

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				messages in the video stream and without manifest decoration, it is impossible to find these blanks and blackouts programma tically.

MS Smooth output

MSS output does not support passthrough of the SCTE-35 messages but does support instructions in the sparse track. Therefore, the workable options are:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Not applicable	Yes or No	Enabled	Yes or No	Removes SCTE-35 messages from the video stream. But instructions are included in the sparse track. You could insert extra messages: although they are not included

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				in the video stream of the output, they are represent ed by instructi ons in the sparse track. You could also implement blanking and blackout.

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Not applicable	No	Disabled	No	Removes SCTE-35 messages from the output. The sparse track does not include instructions. Do not implement blanking or blackout because, without SCTE-35 messages in the video stream and, without data in the sparse track, it is impossibl e to find these blanks and blackouts programma tically.

Adobe RTMP output

Adobe RTMP output does not support passthrough of the SCTE-35 messages but does support manifest decoration. Therefore, the workable options are:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Not applicable	Yes or No	Enabled	Yes or No	Removes SCTE-35 messages from the video stream. But instructions are included in the manifest. You could insert extra messages: although they are not included in the video stream of the output, they are represented by instructions in the manifest. You could also implement blanking and blackout.
Not applicable	No	Disabled	No	Removes SCTE-35 messages from the output. The manifest is not decorated. Do not implement blanking or blackout

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				because, without SCTE-35 messages in the video stream and without manifest decoration, it is impossible to find these blanks and blackouts programmatically.

SMPTE 2110 output

SMPTE 2110 output supports passthrough of the SCTE-35 messages. Therefore, workable options are:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Enabled	Yes or No	Not applicable	Yes or No	Turns on passthrough of SCTE-35 messages. In this case, you could also insert more SCTE-35 messages if desired.

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				Elemental Live converts all these SCTE-35 messages to SCTE-104 messages in the ancillary data stream in the output. You could also implement blanking and blackout.

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Disabled	No	Not applicable	No	Doesn't include SCTE-104 messages in the output. Do not insert extra messages: they are simply get stripped out of the output. Do not implement blanking or blackout. Choose this option only if, in a downstream system, you do not want to replace video that was originally marked by cues.

UDP/TS output

UDP/TS output supports passthrough of the SCTE-35 messages, but it does not support creation of a manifest. Therefore, workable options are:

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
Enabled	Yes or No	Not applicable	Yes or No	Turns on passthrough of SCTE-35 messages. In this case, you could also insert more SCTE-35 message if desired. You could also implement blanking and blackout.
Disabled	No	Not applicable	No	Turns off passthrough in order to remove SCTE-35 messages from the video stream. Do not insert extra messages: they are simply get stripped out of the output. Do not implement blanking or blackout. Choose this option only if, in a downstrea

SCTE-35 passthrough	Insertion of SCTE-35 messages	Manifest decoration	Blanking and blackout	Effect
				m system, you do not want to replace video that was originally marked by cues.

Blanking and passthrough and manifest decoration

It is important to understand that the logic for blanking ad content works on the video content associated with the "ad avail event" while the logic for passthrough and manifest decoration works on the actual SCTE-35 message.

So you can blank ad avails and not pass through SCTE-35 messages or not blank ad avails and not pass through SCTE-35 messages and decorate the manifest or any combination: the actions are independent.

The only exception to this rule is for HLS outputs: manifest decoration and passthrough are either both enabled or both disabled.

Getting ready: Setting the ad avail mode

Read this section if you want to support any of the following features:

- Manifest decoration for all outputs.
- Ad avail blanking for all outputs. (Note that this section does not apply to the blackout feature.)

If you have several outputs and you want to do manifest decoration or ad avail blanking on some of the outputs and not on others, you have to set up two different profiles or events.

> **Note**
>
> You do not have to read this section if you *are* doing Blackout image insertion but you are not doing ad avail blanking or manifest decoration.

Set the ad avail mode

You must set the Ad Avail mode. The Ad Avail mode applies to all outputs: it cannot be set uniquely for individual outputs. To set up the Ad Avail Mode, do the following.

1. In the Profile or Event screen, click Ad Avail Controls (in the Input section towards the top of the screen):
2. In **Ad Avail Trigger**, choose the desired mode from the drop-down menu. This mode identifies which of all possible "ad avail" events are treated as "ad avails." This distinction comes into play in manifest decoration and ad avail blanking. For more information, see Manifest decoration and Ad avail blanking and blackout.

   Typically, you select the mode to match the type ID that you already know the input is using to indicate "ad avail" events.

   - **Splice Insert Mode** . Select this mode if the input uses splice inserts to indicate ad avails. The input might also contain messages for other events such as chapters or programs.
   - **Time Signal with APOS Mode**. Select this mode if the input contains time signals of segmentation type placement opportunity. The input might also contain messages for other events such as chapters or programs.

The following table specifies how a message (containing a specific combination of message type and segmentation type) is treated depending on the mode that is specified. Each message is treated as either an ad avail, a non-ad avail, or as "other."

Read across the first three columns to identify a combination of mode, the message type and the segmentation type. Then in the last three columns, identify how Elemental Live handles that combination.

Mode	Message type ID	Segmentation type ID	Handled as an ad avail event	Not handled as an ad avail event	Handled as another type of event
Splice Insert Mode	Splice Insert	No segmentation descriptor present	X		

Mode	Message type ID	Segmentation type ID	Handled as an ad avail event	Not handled as an ad avail event	Handled as another type of event
		Provider advertisement	X		
		Distributor advertisement	X		
		Placement opportunity	X		
		Other type			X
	Time Signal	Provider advertisement	X		
		Distributor advertisement	X		
		Placement opportunity	X		
		Other type			X
Time Signal APOS Mode	Splice Insert	Any		X	
	Time Signal	Provider advertisement		X	
		Distributor advertisement		X	

Mode	Message type ID	Segmentation type ID	Handled as an ad avail event	Not handled as an ad avail event	Handled as another type of event
		Placement opportunity	X		
		Other type			X

Manifest decoration

You can choose to interpret SCTE-35 messages from the original input and insert corresponding instructions into the output manifest for the following outputs:

- HLS
- MS Smooth (the instructions are inserted in the sparse track).
- RTMP (the instructions are inserted into the data stream, given that RTMP does not support manifests).

How SCTE-35 events are handled in manifests

Based on your criteria for creating the event, you can insert the following information into the manifest.

Type of instruction	When inserted
Base-64	Information about all SCTE-35 messages in the output is incorporated into the manifest; the entire SCTE-35 message is added in base-64 format.
Cue-out, Cue-in	SCTE-35 messages that are ad avails (see Getting ready: Setting the ad avail mode) result in the insertion of "cue-out, cue-in" instructions.

Type of instruction	When inserted
Blackout	Only applies to the SCTE-35 Enhanced ad marker style (for HLS output; see below). SCTE-35 messages that are *not* ad avails result in the insertion of "blackout start/end" instructions, assuming that blackout is enabled (Ad avail blanking and blackout). If blackout is not enabled, these instructions are not inserted.

Splice insert mode

This table describes which of the three types of instructions from the table in the start of this section are inserted for each message type and segmentation type.

Message type ID	Segmentation type ID	Base 64	Cue-out, cue-in	Blackout
Splice insert	No segmentation descriptor present	X	X	
	Provider advertisement	X	X	
	Distributor advertisement	X	X	
	Placement opportunity	X	X	
	Other type (e.g. Chapter, Program)	X		X

Message type ID	Segmentation type ID	Base 64	Cue-out, cue-in	Blackout
Time signal	Provider advertisement	X	X	
	Distributor advertisement	X	X	
	Placement opportunity	X	X	
	Other type (e.g. Chapter, Program)	X		X

Time signal APOS mode

This table describes which of the three types of instructions from the table in the start of this section are inserted for each message type/segmentation type. Note that many segmentation types are completely ignored: they do not get included in the manifest.

Message type Id	Segmentation type ID	Base 64	Cue-out, cue-in	Blackout
Splice insert	Any	X		
Time signal	Provider advertisement	X		
	Distributor advertisement	X		
	Placement opportunity	X	X	

Message type Id	Segmentation type ID	Base 64	Cue-out, cue-in	Blackout
	Other type (e.g. Chapter, Program)	X		X

Procedure to enable manifest decoration

Apple HLS

Enable manifest decoration at the output group level, which means that the manifests for all outputs in that group include instructions based on the SCTE-35 content.

In the Apple HLS output group section, open the **Advanced** section and complete the following fields:

- **Ad Markers**: Click to select a marker type. You can select more than one type.

See Example manifests for Apple HLS for information about the different of markers.

The manifest for each output includes a separate set of tags for each type that you select.

MS Smooth

With MS Smooth, if you enable manifest decoration, you actually insert instructions in the sparse track.

Enable manifest decoration at the output group level, which means that the sparse tracks for all outputs in that group include instructions based on the SCTE-35 content.

In the MS Smooth output group section, complete the following fields:

- **Enable Sparse Track**: Click to select a marker type. You can select more than one type.
- **Acquisition Point ID**: Enter the address of the certificate, if encryption is enabled on the output.

RTMP

With RTMP, if you enable manifest decoration, you insert instructions in the data stream as RTMP does not support manifests.

Enable manifest decoration at the output group level, which means that the manifests for all outputs in that group include instructions based on the SCTE-35 content.

In the RTMP output group section, complete the following fields:

- **Ad Markers**: Click to select a marker type. You can select more than one type.

The data stream includes a separate set of tags for each type that you select.

Ad avail blanking and blackout

You can turn on one or both of the following features to blank out the content associated with a SCTE-35 event:

- “**Blackout**”: Blank out the content for other types of SCTE-35 messages such as chapters and programs.
- “**Ad avail blanking**”: Blank out the content for a SCTE-35 message that is considered an “ad avail” (according to the mode, Getting ready: Setting the ad avail mode).

In both features, the handling is one of the following.

- Replace the video content associated with the event with an image you specify or with a black image.
- Remove the audio associated with the event.
- Remove the closed captions associated with the event.

Topics

- Blanking is global
- Scope of blackout of SCTE-35 messages
- Scope of ad avail blanking of SCTE-35 messages
- Procedure to enable ad avail blanking
- Procedure to enable blackout

Blanking is global

Both Ad avail blanking and Blackout apply to all outputs. You cannot choose to blank out for some outputs and not blank out for others: it is an all-or-nothing decision.

Compare Blanking to Manifest Decoration and Passthrough

Manifest decoration and passthrough have a smaller scope than blanking : they apply only to outputs that support these features.

Take important note of this fact, because if you do *not* do passthrough and do *not* do manifest decoration in a given output (because these are not supported or because you choose not to) but you do implement blanking, there are no “markers” for where the blanked content occurs.

To identify where this blanking is occurring. look for the IDR I-frames that identify where the SCTE-35 message used to be.

Scope of blackout of SCTE-35 messages

All SCTE-35 messages that are “Other type” are blanked out as follows:

SCTE-35 segmentation type	Blanking
Programs	Always
Chapters	Always
Unscheduled events	Always
Network	See below.

How Network End Blackout Differs from Other Events

Network end blackout is different from the other events that trigger a blackout because:

- With Network, blanking starts when the "Network End" instruction is encountered and ends when the "Network Start" instruction is encountered.
- With other events, blanking starts when the “event start” instruction is encountered and ends when the “event end” instruction is encountered.

Scope of ad avail blanking of SCTE-35 messages

For Ad avail blanking (but not for Blackout), the ad avail mode you set controls which SCTE-35 events result in blanking of the content.

Using Splice Insert Mode

This table describes which message type/segmentation type combination is blanked by Ad avail blanking when the Ad Avail mode (Getting ready: Setting the ad avail mode) is **Splice Insert mode**.

Message type ID	Segmentation type ID	Blanked	Not blanked
Splice insert	No segmentation descriptor present	X	
	Provider advertise ment	X	
	Distributor advertise ment	X	
	Placement opportuni ty	X	
	Other type (e.g. Chapter, Program)		X
Time signal	Provider advertise ment	X	
	Distributor advertise ment	X	
	Placement opportuni ty	X	
	Other type (e.g. Chapter, Program)		X

Using Time Signal APOS Mode

This table describes which message type/segmentation type combination is blanked out by Ad avail blanking when the Ad Avail mode (Getting ready: Setting the ad avail mode) is **Timesignal with APOS mode**.

Message type ID	Segmentation type ID	Blanked	Not blanked
Splice insert	Any		X
Time signal	Provider advertisement		X
	Distributor advertisement		X
	Placement opportunity	X	
	Other type (e.g. Chapter, Program)		X

Ad avail blanking and restriction flags: restrictions in the input

SCTE-35 messages of type **time_signal** always contain segmentation descriptors.

SCTE-35 messages of type **splice_insert** may or may not include segmentation descriptors.

If the input has SCTE-35 messages that *do* include segmentation descriptors, these segmentation descriptors always include two types of flags. These flags provide additional information as guidance for blanking in specific situations:

- **web_delivery_allowed_flag**.
  - "True" means that there is no restriction on including the ad avail event's content in a stream intended for web delivery: you do not need to blank out content in streams intended for web delivery.
  - "False" means there is a restriction: you should blank out the content .

- **no_regional_blackout_flag.**
  - "True" means that there is no restriction on including the ad avail event's video in a stream intended for regional markets: you do not need to blank out content in streams intended for regional markets.
  - "False" means there is a restriction: you should blank out the content.

If neither flag is present (usually the case with **splice_inserts**), then both are considered to be false: blanking should occur.

If both flags are present (which is usually the case; it is unusual to have only one flag present), then a “false” for one flag takes precedence over a “true” for the other flag: blanking should occur.

Typically, in any given message in the input only one of these flags would ever be set to "false", so only one restriction would ever be in place. There would typically never be *both* a regional delivery restriction and a web delivery restriction. This is because, if content is considered restricted for regional delivery, then it would not also be considered restricted for web delivery (where the concept of a region does not relate).

To summarize, this table shows the blanking logic that applies to each ad avail event that is encountered:

	Content of correspon ding SCTE-35 Message **Web delivery allowed?**	**Content of correspon ding SCTE-35 Message** **Regional delivery allowed?**	**Result**	**Comment**
S1	Flag is not present.	Flag is not present.	Blanking will occur.	This combinati on can occur only in a message type splice_insert (where the segmentation

	Content of correspon ding SCTE-35 Message Web delivery allowed?	Content of correspon ding SCTE-35 Message Regional delivery allowed?	Result	Comment
				descriptor is optional).
S2	True	True	Blanking will not occur.	
S3	True	False	Blanking will occur.	
S4	False	True	Blanking will occur.	

Ad avail blanking and restriction flags: Elemental Live handling of restrictions

You can modify this default blanking behavior by instructing Elemental Live to ignore a restriction flag that is set to "false", so that blanking will not occur for this ad avail event. In other words, to use this logic: “Even if the message indicates to blank content because a regional blackout is in place, do not follow this instruction. Ignore the fact that a regional blackout is in place and do not blank content.”

You modify the behavior by setting one or the other of the **Ignore** flags to:

- **Cleared**(default). The logic as above applies: so continue to observe the restriction.
- **Selected**. Ignore the restriction and do not blank video for the ad avail event.

> **Warning**
>
> Never set both fields to **Ignore**!

To summarize, the Ignore flags make a difference only to the scenarios in this table:

	Content of correspon ding SCTE-35 Message Web delivery allowed?	Content of correspon ding SCTE-35 Message Regional delivery allowed?	Restriction field in Elemental Live	Result
S3	True	False	Ignore “regional delivery” restriction	Blanking will *not* occur.
S4	False	True	Ignore “web delivery” restriction	Blanking will *not* occur.

Ad avail blanking and restriction flags: restriction flags with “splice insert”

If you selected Splice Insert as the Ad Avail mode, then you can assume that the SCTE-35 ad avail message not include the two restriction flags described above. Every SCTE-35 ad avail message should result in an ad avail.

Therefore, if you know that input contains splice inserts (not time signals), you should leave both Elemental Live fields cleared.

Procedure to enable ad avail blanking

To enable ad avail blanking

1. In the Profile or Event screen, click Advanced Avail Controls (in the Input section towards the top of the screen):
2. Select or clear the two restriction fields:

   - **Cleared** (default): Observe the restriction and blank the content for the ad avail event.
   - **Selected**: Ignore the restriction and do *not* blank the content for the ad avail event.

3. Go down to the Global Processors section and complete the following fields:

   - **Ad Avail Blanking**: Click to turn on. The **Blanking Image** field appears.
   - **Blanking Image**: Specify a `.bmp` or `.png` file to use for the blanking. If you leave this field blank, a plain black image is inserted.

Procedure to enable blackout

To enable blackout

1. In the Profile or Event screen, go down to the Global Processors section and complete the following fields:

   - **Blackout Image Insertion**: Click to turn on. The Blanking Image field appears.
   - **Blanking Image**: Specify a `.bmp` or `.png` file to use for the blanking. If you leave this field empty, a plain black image is inserted.

2. If you want to enable network end blackout (in other words, blank content when network transmission has ended and remove blanking only when network transmission resumes), complete these fields:

   - **Enable Network End Blackout**: Selected.
   - **Network ID**: The **Entertainment Identifier Registry** (**EIDR**) **ID** of the network in the format 10.nnnn/xxxx-xxxx-xxxx-xxxx-xxxx-c (case- insensitive). Only network end events with this ID trigger blackout.
   - **Network End Blackout Image**: Specify a `.bmp` or `.png` file to use for the blanking. If you leave this field blank, a plain black image is inserted.

Passthrough or removal of SCTE messages

SCTE-35 messages from the input can be passed through (included) in the data stream for the following outputs.

- Archive outputs with MPEG-2 as the container: You specify whether to pass through at the output level.
- HLS: You specify whether to pass through at the output group level: passthrough or removal applies globally to all outputs in the output group.

- UDP/TS: You specify whether to pass through at the output level: for each individual output in the output group.

SCTE-104 messages are handled for each output as follows:

- If you choose to pass through the SCTE-35, then all SCTE-104 messages are converted to SCTE-35 messages (of the same message type) and included in the data stream.
- If you choose to remove the SCTE-35 messages, the SCTE-104 messages are also removed.

Topics

Archive procedure

You enable or disable passthrough at the output level: only in outputs that have an MPEG-2 TS container.

1. In the Profile or Event screen, go to the Output Groups section at the bottom of the screen and display the tab for Archive Output Group.
2. In the output that has the MPEG-2 TS container, open the PID Control section. Complete the following fields:
   - SCTE-35: Click to select.
   - SCTE-35 PID: Enter the ID of the PID where you want the SCTE-35 messages to go.

Result

All SCTE-35 messages from the input are included in the data stream of this output.

Apple HLS passthrough procedure

Passthrough is enabled or disabled individually for each output, which means it can be applied differently for different outputs in the same group.

1. If you have not already set up for manifest decoration, do so now; see Procedure to enable manifest decoration.
2. In the Profile or Event screen, go to the Output Groups section at the bottom of the screen and display the tab for **Apple HLS Output Group**.
   a. In each output, open the PID Control section. You will note that the SCTE-35 field is automatically selected (because you set up for manifest decoration) and you cannot clear it .
   b. Complete the following field:
      - SCTE-35 PID field: Enter the ID of the PID where you want the SCTE-35 messages to go.

Result

All SCTE-35 messages from the input are included in the data stream of the relevant output.

UDP/TS procedure

Passthrough is enabled or disabled individually for each output, which means it can be applied differently for different outputs in the same group.

To enable passthrough

1. In the Profile or Event screen, go to the Output Groups section at the bottom of the screen and display the tab for UDP/TS Output Group.
2. In the output where you want to pass through SCTE-35 messages, open the PID Control section. Complete the following fields:
   - SCTE-35: Click to select.
   - SCTE-35 PID: Enter the ID of the PID where you want the SCTE-35 messages to go.

Result

All SCTE-35 messages from the input are included in the data stream of the relevant output.

SCTE-35 message insertion into currently running events

You can use the Elemental Live REST API to insert SCTE-35 messages into an Elemental Live event that is currently running. The API supports insertion of a SCTE-35 message of type **splice_insert** or **time signal** (Working with time signals).

Topics

- Working with splice inserts
- Working with time signals

Working with splice inserts

Splice inserts inserted by the REST API are always of type “ad avail.”

You can:

- Insert a SCTE-35 ad avail of type splice_insert. See Insert a new splice insert message.
- Get the timecode of the content that is currently being processed. This data can be useful to determine the start time you need to enter in the command. See Get current time.
- Insert an end time in an ad avail. See Insert an end time in an existing ad avail.
- Cancel a SCTE-35 ad avail that is pending (the start time has not yet been reached). See Cancel a pending ad avail.

Effect of these commands

These commands modify the data stream as it is being encoded.

Their precise effect on other parts of the content depends on how the event is set up, as described in earlier chapters of this guide. So it depends on the Ad Avail mode, whether manifest decoration is enabled, blanking and blackout are enabled, and whether passthrough is enabled.

Topics

- Insert a new splice insert message
- Get current time
- Insert an end time in an existing ad avail
- Cancel a pending ad avail

Insert a new splice insert message

Inserts a SCTE-35 message of type splice_insert in the stream either immediately or at a specified time. The command always includes a start time. It can also include a duration (which implies an end time).

The command does not support inclusion of a segmentation descriptor, which means that the message is always considered to be an “ad avail.”

HTTP URL

```
POST <IP address of Live node>/live_events/<ID of event>/cue_point
```

Body of HTTP

The XML body contains one cue_point elements containing the following tags:

Tag	Sub-tag	Type	Value
event_id		integer	Specify an ID for this SCTE-35 request to allow for canceling of the insertion later (Cancel a pending ad avail). Or leave blank, in which case an ID is generated and returned in the response.
splice_time			Include this in order to specify the insertion point relative to the stream timecode. Include either splice_time or splice_offset, not both. See Specifying Time with splice_time Tag for details.

Tag	Sub-tag	Type	Value
			Specify the time by including the hours, minutes, seconds, and frames tags.
	hours	integer	The start time of the ad avail. All fields are required. Enter the time in 24-hour format. To insert the ad avail immediately (taking into account that there is a small delay while the request is processed), enter 0 in all fields.
	minutes	integer	
	seconds	integer	
	frames	integer	The frame within the specified seconds at which to insert the ad avail. If blank, the start time is the first frame in the specified second.

Tag	Sub-tag	Type	Value
splice_offset		integer	The start time of the ad avail. Include either splice_time or splice_offset, not both. Include in order to specify the start time for the ad avail as the specified milliseco nds after the request is received. See Specifying Time with splice_offset Tag. Specify the milliseco nds. The number cannot be negative.

Tag	Sub-tag	Type	Value
duration		integer	Optional. You can include a duration so that a start time is included and an end time is implied by the length of time for the duration. Or you can omit the duration so that only a start time is included. If you omit the duration, you must enter a separate command for the end time.

Specifying time with "splice_time" tag

Use the splice_offset tag to specify the start time as a specific clock time, for example, at 10:20:33. The time you specify must match the timecode format in the event, as specified by the **Timecode Config Source** field in the event or profile. For example, if you specify 10:20:33 and the event uses system clock, the message is inserted at 10:20:33 UTC. If the event uses local time, the message is inserted at 10:20:33 for the time zone of the node.

To verify the timecode format in the event, submit a GET live_events request and (in the response) read the value in the **timecodeconfig** tag.

Splice Time requires either knowing in advance to insert an ad avail at a given time, or obtaining the current time and inserting an offset so that the time is not missed. It is easier to use **Splice Time** when you know start times in advance. You can obtain the timecode of the content currently being encoded; see Get current time.

Specifying Time with "splice_offset" Tag

You can use the **splice_offset** tag to specify time as a number of milliseconds into the future from the moment at which the command is performed. This offset can be 0, which means to insert immediately.

Time of insertion of the message and time for the ad avail

The time the message is inserted and the start time for the ad avail are typically not identical. For example, you insert a message that says “insert an ad avail at 10:35:15:0”. The message is actually inserted in the content close to the moment that you enter the request. The downstream system won't act on the instruction (it won't insert the ad avail) until the specified time.

So you can insert the message in advance of its targeted start time. In fact, it is a good idea to include some offset. But take care when calculating offset.

- **Too little offset**: If you do not add enough offset, the specified time might have already passed. The ad avail might still be inserted in the video (at the current frame) but will be inserted too late for Elemental Live to act upon it.

  For example, the command says “insert an ad avail at 10:35:15.0” but, if that time has passed, the message is inserted at 10:35:40.0 (for example). Elemental Live inserts the message if its request time is less than 1 hour after the targeted time. Otherwise, it discards the message. Note that the message is inserted in the content but it has a start time that has already passed.
- **Too much offset**: The maximum offset is 6 hours. Within that range, the ad avail will be inserted at a timepoint (for example 10:35:20), and the command is “insert ad avail at 10:36:00.0, that is, in 40 seconds from now.”

Splice offset and clock changes

When you use splice offsets in a splice insert, you must restart the event whenever the clocks change to or from Standard Time. If you don’t restart the event, your splice inserts will permanently either be inserted at the wrong time or not be inserted at all.

These rules apply only if the event uses the local system clock, because this is the only type of clock that is affected by the clock change.

If you don't plan for any splice inserts in the first few minutes after the clocks change, you can simply restart the event as soon as the clocks change in the region and time zone that the Elemental Live appliance is configured for.

If your workflow does include plans for splice inserts just after the clocks change, follow these steps:

- Determine the date and time that the clocks change,in the region and time zone that the Elemental Live appliance is configured for.
- Identify all the splice insert commands that you plan to enter *before* the clocks change, and where the splice insert has an intended start time that is *after* the clocks change. For example, you plan to enter a command at 1:15 a.m. for a splice insert with a start time that is after the clocks change. Don't enter any of these commands.
- As soon as the clocks change (so at approximately 2:00.01 a.m.), restart the event. Restart each event where the problem scenario applies.
- After the event restarts, you can insert the splice insert commands that you previously didn't enter. It's possible that the intended start time of one or more splice inserts has already passed. You might want to reschedule the start time for the splice inserts, or you might want to skip these splice inserts.
- You can then continue entering splice insert commands in the usual way.

Response

The body of the response is XML content consisting of one **response** element containing the following tags:

Tag	Sub-tag	Sub-sub-tag	Type	Description
event_id			integer	The event ID of this SCTE-35 request.
splice_time	hours		integer	If splice_time was specified , the hour, minutes, seconds and frame at which
	minutes		integer	
	seconds		integer	
	frames		integer	

Tag	Sub-tag	Sub-sub-tag	Type	Description
				to insert the ad avail. If splice_offset was specified, all tags specify “0.”
splice_offset			integer	If splice_offset was specified , the time at which to insert the ad avail. If splice_time was specified , this tag has a null value.
message			string	A description of the action taken.
errors				Included only in an error response.
	error	code		An error code.
	error	message	string	A human-rea dable error message.

A success response does not include the <errors> element. A failure response contains only the <event_id> and <errors> elements.

Example Message

The following shows an example message:

```
<message>Inserted event [32] at event time[08:02:38], PTS[00:02:20.982]. Avail
 time[08:02:38 0f] PTS[08:02:38.023], duration[01:00:00.000]. Current NTP
 [15:18:05.712]</message>
```

Data	Description
Inserted event [n]	The event ID for the ad avail request.
event time	The requested start time of the ad avail, as per the original request.
PTS (first occurrence)	The time at which the SCTE-35 message insertion request was received by Elemental Live, in a clock representation of the presentation timestamp (PTS). This PTS is a "timer", not a clock time.
Avail Time	The requested start time of the ad avail, including the frame. This time is in the timecode specified in the event or profile. For more information, see About timecode configuration and timers. This time is in the timecode specified in the event or profile. If the timecode configuration source is Clock time, Local time, and Specified time, this time is a "clock time."
PTS (second occurrence)	The requested start time of the ad avail (with the frame converted to milliseconds).
duration	The duration of the ad avail, if specified, in 24-hour format.
Current NTP	The network time protocol (NTP) when the SCTE-35 message insertion request was received by Elemental Live.

Splice insert examples

Splice time

Insert a message into the event with the ID 3. Insert the message at 10 hours, 32 minutes, and 10 seconds, and give it a duration of 30 seconds. (The implied end time will be 10 hours ,32 minutes, and 40 seconds.)

```
POST 10.4.136.95/live_events/3/cue_point
----------------------------------------
<cue_point>
<splice_time>
<hours>10</hours>
<minutes>32</minutes>
 <seconds>10</seconds>
 <frames>0</frames>
</splice_time>
<duration>30</duration>
</cue_point>
```

The following shows a success response where splice_offset was used in the request. The SCTE-35 request has an ID of 8.

```
<response value="cue_point">
  <event_id>8</event_id>
  <splice_time>
    <hours>0</hours>
    <minutes>0</minutes>
    <seconds>0</seconds>
    <frames>0</frames>
  </splice_time>
  <splice_offset>8000</splice_offset>
  <message> Inserted at PTS[1234]. Avail time[00:00:05.000] PTS[2345], duration[30].
  </message>
</response>
```

The following shows a failure response:

```
<response value="cue_point">
  <event_id>8</event_id>
  <errors>
```

```
    <error>
      <code>1040</code>
      <message>Preroll time must be positive integer</message>
    </error>
  <errors>
</response>
```

Splice offset

Insert a message into the event with the ID 3. Insert the message 8000 milliseconds from the moment the command is performed. The message has a duration of 30 seconds.

```
POST 10.4.136.95/live_events/3/cue_point
----------------------------------------
<cue_point>
<splice_offset>8000</splice_offset>
<duration>30</duration>
</cue_point>
```

Get current time

You can obtain the timecode of the frame that is currently being processed in the specified event. Refer to the following tables.

HTTP URL

```
POST <IP address of Live node>/live_events/<ID of event>/cue_point
```

Body of HTTP

The XML body contains one `cue_point` element containing the following tag:

Tag	Type	Value
get_current_time	Integer	Always 1

Response

The body of the response contains one **response** element containing the following tags:

Tag	Sub-tag	Type	Value
tag		integer	A unique ID that could be used in the event_id in a new request to insert a splice_insert. See the explanation below this table.
splice_time	hours	integer	The time (and frame) associated with the event at the moment that the API request was processed. Time is in 24-hour format. For general informati on about timecodes in events, see [About timecode configura tion and timers]().
	minutes	integer	
	seconds	integer	
	frames	integer	
message		string	A description of the time: the presentat ion timestamp (PTS) of the time and the time in NTP.
splice_offset			Always 0. See the explanation below this table.
value		string	Always “cue_point.”

The response is provided in this format so that you could take the entire response, clean it up a bit (for example, changing the “tag” tag to “event_id”), and use it as the body of a request to insert a spliceinsert (see [Insert a new splice insert message]().

Get current time example

The following shows a request for the current timecode in the event that has the ID 15:

```
POST 10.4.136.95/live_events/15/cue_point
-----------------------------------
<cue_point>
  <get_current_time>1</get_current_time>
</cue_point>
```

The following shows an example response for the request:

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <tag>1</tag>
  <splice_time>
    <hours>0</hours>
    <minutes>0</minutes>
    <seconds>2</seconds>
    <frames>23</frames>
  </splice_time>
  <message>PTS[00:00:02.969]. Current NTP [16:21:23.573].</message>
  <splice_offset>0</splice_offset>
  <value>cue_point</value>
</response>
```

Insert an end time in an existing ad avail

You can insert an end time in the following situations:

- To add an end time to a message that has not yet started if an end time was not initially included.
- To cut short an ad avail that is in progress. The command can be used in this way if no end time was initially included or if it was included but you want to end early.

HTTP URL

```
PUT <IP address of Live node>/live_events/<ID of event>/cue_point
```

Body of HTTP

The XML body contains one `cue_point` element containing the following tags:

Tag	Sub-tag	Type	Value
event_id		integer	The event ID of the original POST cue_point (the request that did not include a duration).
return_offset	integer		The number of milliseconds to wait before inserting the ad avail.

Insert an End Time in an Existing Ad Avail Example

Insert an end time immediately into the event with the ID 4. The original SCTE-35 ad avail has an ID of 38.

```
PUT 10.4.136.96/live_events/4/cue_point
----------------------------------------
<cue_point>
  <event_id>38</event_id>
  <return_offset>0</return_offset>
</cue_point>
```

Cancel a pending ad avail

If you inserted an ad avail the start time has not yet passed, you can cancel the insertion. No SCTE-35 message will be inserted.

HTTP URL

```
POST <IP address of Live node>/live_events/<ID of event>/cue_point
```

Body of HTTP

The XML body contains one `cue_point` element containing the following tag:

Tag	Sub-tag	Type	Value
cancel_event_id		integer	The event ID of the original SCTE-35 request.

Response

The response includes a message that identifies the event ID of the original request.

Cancel a Pending Ad Avail Example

The following shows a request to cancel the insertion of a pending ad avail in the event with the ID 4. The original SCTE-35 ad avail has an ID of 38:

```
POST 10.4.136.96/live_events/4/cue_point
----------------------------------------
<cue_point>
  <cancel_event_id>38</cancel_event_id>
</cue_point>
```

The following shows an example response for the request:

```
<response>
  <tag>13</tag>
  <event_id>5</event_id>
  <message>Canceled eventID [5]</message>
  <value>cue_point</value>
</response>
```

Working with time signals

Time signals inserted by the REST API can be one of the “ad avail” types, or some other type, depending on what you specify in the segmentation descriptor in the request.

Effect of creating a time signal

Insertion of a time signal modifies the data stream as it is being encoded.

The effect of time signals on other parts of the content depends on how the event is set up, as described in earlier chapters of this guide. So it depends on the Ad Avail mode, whether manifest decoration is enabled, blanking and blackout are enabled, and whether passthrough is enabled.

Insert a new time signal message

Inserts a SCTE-35 message of type time_signal in the stream either immediately or at a specified time. The command always includes a start time (in the time tag) and a duration (included in the segmentation descriptor).

HTTP URL

```
POST <IP address of Live node>/live_events/<ID of event>/time_signal
```

Body of HTTP

The XML body contains one **time_signal** element containing the following tags:

Tag	Sub-tag	Type	Value
time	hours	integer	The start time. All fields are required. Enter the time in 24-hour format. To insert the message immediately (taking into account that there will be a small delay while the request is processed), enter 0 in all fields.
	minutes		
	seconds		
	frames		
descriptors		string	Optional. A hexadecimal string containing time signal descriptor data in the form of

Tag	Sub-tag	Type	Value
			a binary blob. See the SCTE-35 specifica tion for details on the contents and format. This descriptor includes the time signal duration, segmentation type ID, and other key information.

Response

The body of the response is XML content consisting of one **response** element containing the following tags:

Tag	Sub-tag	Sub-sub-tag	Type	Description
tag			integer	A unique ID
signal_time	hours		integer	The start time, repeated from the request.
	minutes		integer	
	seconds		integer	
	frames		integer	
message			string	A description of the action taken.
errors				Included only in an error response.
	error	code		An error code.

Tag	Sub-tag	Sub-sub-tag	Type	Description
	error	message	string	A human-readable error message.

A success response does not include the <errors> element. A failure response contains only the <tag> and <errors> elements.

Example Message

The following shows an example message:

```
<message>Inserted event [32] at event time[08:02:38], PTS[00:02:20.982]. Avail
 time[08:02:38 0f] PTS[08:02:38.023], duration[01:00:00.000]. Current NTP
 [15:18:05.712]</message>
```

Data	Description
Inserted event [n]	The event ID for the ad avail request.
event time	The requested start time of the ad avail, as per the original request.
PTS (first occurrence)	The time at which the SCTE-35 message insertion request was received by Elemental Live, in a clock representation of the presentation timestamp (PTS). This PTS is a “timer”, not a clock time.
Avail time	The requested start time of the ad avail, including the frame. This time is in the timecode specified in the event or profile. This time is in the timecode specified in the event or profile. If timecode configuration source is Clock time, Local time, and Specified time, this time is a “clock time.”

Data	Description
PTS (second occurrence)	The requested start time of the ad avail (with the frame converted to milliseconds).
duration	The duration of the ad avail, if specified, in 24-hour format.
Current NTP	The network time protocol (NTP) when the SCTE-35 message insertion request was received by Elemental Live.

Insert a new time signal message example

The following shows a request to insert a SCTE-35 message immediately into the event that has ID 3.

```
POST 10.4.136.95/live_events/3/time_signal
----------------------------------------
<time_signal>
  <time>
    <hours>0</hours>
    <minutes>0</minutes>
    <seconds>0</seconds>
    <frames>0</frames>
   </time>
  <descriptors>021B43554549000000027FBF030C54564E41313030303030313300000
  </descriptors>
 </time_signal>
```

The following shows an example success response for the request:

```
<response value="time_signal">
  <message> Inserted time signal at event time[1234], PTS[1234]. Signal time[1234]
 PTS[1234].
  </message>
  <tag>1</tag>
  <signal_time>
    <hours>0</hours>
    <minutes>0</minutes>
    <seconds>0</seconds>
```

```
  </signal_time>
</response>
```

The following shows an example failure response for the request:

```
<response value="time_signal">
  <tag>1</tag>
  <errors>
    <error>
     <code>1040</code>
     <message>Invalid time signal message</message>
    </error>
  </errors>
</response>
```

POIS conditioning

Elemental Live events can be configured to communicate with a POIS server. During processing of the input, each time a SCTE-35 message is encountered, Elemental Live sends the message contents to the server. The POIS responds with SCTE-35 content that might be identical to the original, slightly different from, or completely different from the original.

Elemental Live also supports handling of “out-of-band” SCTE-35 messages from the POIS – messages that are not a response to a message originally sent by Elemental Live. If such a message is received, Elemental Live accepts and processes it.

POIS conditioning and other SCTE 35 features

This section describes how POIS conditioning interacts with other SCTE 35 features that you can set up.

Ad avail mode

When POIS conditioning is enabled, the ad avail mode is always set to *splice insert*. For information about how this value affects the behavior of manifest decoration and ad avail blanking see [Getting ready: Setting the ad avail mode](#).

SCTE-35 messages inserted by REST API

All these messages are sent to the POIS along with messages that are already in the input.

Manifest decoration

The effect of POIS conditioning on manifest decoration is as follows:

- **HLS Outputs –** A message received from the POIS might include instructions to decorate an HLS manifest (not other types of manifests). This information is used to decorate the HLS manifest.

  How Elemental Live processes the decoration information depends on how the event or event has been configured:

  - If the event or event does not have manifest decoration enabled for HLS outputs, then the information is ignored.
  - If the event does have it enabled, then the decoration information is inserted in the manifest.
  - The information is inserted into the manifest "as is." The style of the information might not match the styles (ad marker styles) specified in the event. Elemental Live doesn't check for format inconsistencies between these instructions and the ad marker style.
- **Other Outputs –** Decoration of other manifest types is according to the information in the SCTE-35 message and how the Elemental Live event is set up for manifest decoration and which ad avail mode is enabled. The POIS conditioning has no effect on these manifest types.

Blanking and blackout

The effect of POIS conditioning on blanking and blackout is as follows:

- **Extra Blackout Instructions –** A message received from the POIS might include explicit "blank the content corresponding to this SCTE-35 message" instructions for any SCTE-35 message.
- **Blanking Image –** A POIS response might include reference to a blanking .png or .bmp file. Elemental Live uses this file for any blanking/blackout if it can find the file at `/data/server/esam/` on the Elemental Live node.

  If Elemental Live cannot find the file, it uses a black slate.
- **Restriction Flags –** In ESAM mode, the Override restriction flags in the Elemental Live event (**Ignore Web Delivery Allowed** flag and **Ignore No Regional Blackout** flag) are always cleared (not selected). See Ad avail blanking and blackout.
- **Passthrough or Removal –** Without POIS conditoin, if passthrough is enabled in the Elemental Live event, then the rule is that all SCTE-35 message are passed through.

  But when POIS conditioning is enabled, the POIS can override this rule: if the POIS instruction is to remove a given SCTE-35 message, then that message is removed and is not passed through, even though passthrough is enabled in Elemental Live.

Procedure to enable POIS conditioning

1. In the **Profile** or **Event** screen, select **Advanced Avail Controls** (in the Input section towards the top of the screen):
2. In **Ad Avail Trigger**, choose **ESAM**. More fields appear.
3. Complete the fields as follows:
   - Complete the first 6 fields to identify the endpoints on the POIS.
   - For **Response Signal Preroll**, change the value as desired to set the distance (in milliseconds) between the time that the SCTE-35 message is inserted and the start time of that ad avail.

Setting up using the REST API

This topic lists the parameters found on the Elemental Live event or profile and specifies the location of those parameters in the XML for an event or profile. This topic does not cover control of SCTE-35 message via the REST API at runtime; that information is in SCTE-35 message insertion into currently running events.

Set the Ad Avail Mode

Field	XML Tag
Advanced Avail Controls > Ad Avail Trigger	ad_trigger

Manifest Decoration

Field	XML Tag
Output Group > Apple HLS > Advanced > Ad Markers	output_group/apple_live_group_settings/ad_markers
Output Group > MS Smooth > Enable Sparse Track	output_group/ms_smooth_group_settings/enable_sparse_track
Output Group > MS Smooth > Acquisition Point ID	output_group/ms_smooth_group_settings/acquisition_point_id

Field	XML Tag
Output Group > RTMP> Ad Markers	output_group/rtmp_group_settings/ad_markers

Ad Avail Blanking and Blackout

Field	XML Tag
Advanced Avail Controls > Ignore no_regional_blackout_flag	ignore_no_regional_blackout_flag
Advanced Avail Controls > Ignore web_delivery_allowed_flag	ignore_web_delivery_allowed_flag
Processors > Global Processors > Ad Avail Blanking > On/Off	avail_blanking/enabled/
Processors > Global Processors > Ad Avail Blanking > Browse	avail_blanking/avail_blanking_image/certificate_file
Processors > Global Processors > Ad Avail Blanking > Browse	avail_blanking/avail_blanking_image/interface
Processors > Global Processors > Ad Avail Blanking > Credentials icon > Password	avail_blanking/avail_blanking_image/password
Processors > Global Processors > Ad Avail Blanking > Browse	avail_blanking/avail_blanking_image/uri
Processors > Global Processors > Ad Avail Blanking > > Credentials icon > Username	avail_blanking/avail_blanking_image/username
Processors > Global Processors > Blackout Image Insertion > On/Off	blackout_slate/enabled/
Processors > Global Processors > Blackout Image Insertion > Enable Network End Blackout > Network ID	blackout_slate/network_id

Field	XML Tag
Processors > Global Processors > Blackout Image Insertion > Browse	blackout_slate/blackout_slate_image/certificate_file
Processors > Global Processors > Blackout Image Insertion > Browse	blackout_slate/blackout_slate_image/interface
Processors > Global Processors > Blackout Image Insertion > Browse	blackout_slate/blackout_slate_image/password
Processors > Global Processors > Blackout Image Insertion > Browse	blackout_slate/blackout_slate_image/uri
Processors > Global Processors > Blackout Image Insertion > Browse	blackout_slate/blackout_slate_image/username
Processors > Global Processors > Blackout Image Insertion > Enable Network End Blackout > Network End Blackout Image > Browse	blackout_slate/network_end_blackout_image/certificate_file
Processors > Global Processors > Blackout Image Insertion > Enable Network End Blackout > Network End Blackout Image > Browse	blackout_slate/network_end_blackout_image/interface
Processors > Global Processors > Blackout Image Insertion > Enable Network End Blackout > Network End Blackout Image > Credentials > Password	blackout_slate/network_end_blackout_image/password
Processors > Global Processors > Blackout Image Insertion > Enable Network End Blackout > Network End Blackout Image > Browse	blackout_slate/network_end_blackout_image/uri

Field	XML Tag
Processors > Global Processors > Blackout Image Insertion > Enable Network End Blackout > Network End Blackout Image > Credentials > Username	blackout_slate/network_end_blackout_image/username

Passthrough or Removal

Field	XML Tag
Archive Output Group > Output > MPEG-2 TS > PID Control > SCTE-35	output_group/output/scte35_passthrough
Archive Output Group > Output > MPEG-2 TS > PID Control > SCTE-35 PID	output_group/output/m2ts_settings/scte35_pid
Apple HLS Output Group > Output > PID Control > SCTE-35	output_group/output/scte35_passthrough
Apple HLS Output Group > Output > PID Control > SCTE-35 PID	output_group/output/m3u8_settings/scte35_pid
UDP/TS Output Group > Output > SCTE-35	output_group/output/scte35_passthrough
UDP/TS Output Group > Output > SCTE-35 PID	output_group/output/ts_settings/scte35_pid

POIS conditioning

Field	XML Tag
Advanced Avail Controls > Ad Avail Trigger > Acquisition Point Identifier	esam/acquisition_point_id/
Advanced Avail Controls > Ad Avail Trigger > Asset URI Identifier	esam/asset_uri_id/

Field	XML Tag
Advanced Avail Controls > Ad Avail Trigger > Signal Conditioner Endpoint	esam/scc_uri/certificate_file
Advanced Avail Controls > Ad Avail Trigger > Signal Conditioner Endpoint	esam/scc_uri/interface
Advanced Avail Controls > Ad Avail Trigger > Signal Conditioner Endpoint	esam/scc_uri/password
Advanced Avail Controls > Ad Avail Trigger > Signal Conditioner Endpoint	esam/scc_uri/uri
Advanced Avail Controls > Ad Avail Trigger > Signal Conditioner Endpoint	esam/scc_uri/username
Advanced Avail Controls > Ad Avail Trigger > Alternate Signal Conditioner Endpoint	esam/alternate_scc_uri/certificate_file
Advanced Avail Controls > Ad Avail Trigger > Alternate Signal Conditioner Endpoint	esam/alternate_scc_uri/interface
Advanced Avail Controls > Ad Avail Trigger > Alternate Signal Conditioner Endpoint	esam/alternate_scc_uri/password
Advanced Avail Controls > Ad Avail Trigger > Alternate Signal Conditioner Endpoint	esam/alternate_scc_uri/uri
Advanced Avail Controls > Ad Avail Trigger > Alternate Signal Conditioner Endpoint	esam/alternate_scc_uri/username
Advanced Avail Controls > Ad Avail Trigger > Manifest Conditioner Endpoint	esam/mcc_uri/certificate_file
Advanced Avail Controls > Ad Avail Trigger > Manifest Conditioner Endpoint	esam/mcc_uri/interface
Advanced Avail Controls > Ad Avail Trigger > Manifest Conditioner Endpoint	esam/mcc_uri/password

Field	XML Tag
Advanced Avail Controls > Ad Avail Trigger > Manifest Conditioner Endpoint	esam/mcc_uri/uri
Advanced Avail Controls > Ad Avail Trigger > Manifest Conditioner Endpoint	esam/mcc_uri/username
Advanced Avail Controls > Ad Avail Trigger > Alternate Manifest Conditioner Endpoint	esam/alternate_mcc_uri/certificate_file
Advanced Avail Controls > Ad Avail Trigger > Alternate Manifest Conditioner Endpoint	esam/alternate_mcc_uri/interface
Advanced Avail Controls > Ad Avail Trigger > Alternate Manifest Conditioner Endpoint	esam/alternate_mcc_uri/password
Advanced Avail Controls > Ad Avail Trigger > Alternate Manifest Conditioner Endpoint	esam/alternate_mcc_uri/uri
Advanced Avail Controls > Ad Avail Trigger > Alternate Manifest Conditioner Endpoint	esam/alternate_mcc_uri/username
Advanced Avail Controls > Ad Avail Trigger > Response Signal Preroll	esam/response_signal_preroll/

Example manifests for Apple HLS

This section lists example manifests for the Apple HLS manifest styles that AWS Elemental supports.

Example manifest for Adobe ad marker

This does not insert any CUE-OUT CONT (continuation tags) to indicate to a player joining mid-break that there is a current avail. This does not insert a CUE-IN tag at the end of the avail.

Structure

Segment	Tag	Tag Count
Segment in which the ad avail starts.	1 CUE: DURATION tag	1

Tag contents

- CUE:DURATION containing:
  - duration: Duration in fractional seconds.
  - id: An identifier, unique among all ad avails CUE tags.
  - type: SpliceOut
  - time: The PTS time for the ad avail, in fractional seconds.

Example

The following is the tag for an ad avail lasting 414.171 PTS.

```
#EXT-X-CUE:DURATION="201.467",ID="0",TYPE="SpliceOut",TIME="414.171"
```

Example manifest for AWS Elemental ad marker

Structure

Segment	Tag	Tag Count
Segment in which the ad avail starts	CUE-OUT	1
Each succeeding segment	CUE-OUT-CONT	0-n
Segment in which ad avail ends	CUE-IN	1

Tag contents

- CUE-OUT contains Duration.

- CUE-OUT-CONT contains Elapsed time and Duration.
- CUE-IN has no content.

Example

```
#EXT-X-CUE-OUT:30.000
.
.
.
# EXT-X-CUE-OUT-CONT: 8.308/30
.
.
.
# EXT-X-CUE-OUT-CONT: 20.391/30
.
.
.
# EXT-X-CUE-IN
```

Example manifest for SCTE-35 enhanced ad marker

Structure

Segment	Tag	Tag Count
Segment in which the ad avail starts	OATCLS-SCTE35	1
Segment in which the ad avail starts	ASSET	1
Segment in which the ad avail starts	CUE-OUT	1
Each succeeding segment	CUE-OUT-CONT	0-n
Segment in which ad avail ends	CUE-IN	1

Tag contents

- OATCLS-SCTE35 containing the base-64-encoded raw bytes of the original SCTE-35 ad avail message.
- ASSET containing the CAID or UPID as specified in the original SCTE35 message.
- 1 CUE-OUT per ad avail.
- CUE-OUT-CONT containing:
  - The elapsed time of the avail.
  - The duration declared in the original SCTE35 message.
  - SCTE35 containing the base-64-encoded raw bytes of the original SCTE-35 ad avail message.

    These lines repeat until the ad avail ends.
- CUE-IN to indicate the end of the avail.

Example

```
#EXT-OATCLS-SCTE35:/DA0AAAAAAAAAAAABQb+ADAQ6QAeAhxDVUVJQAAAO3/PAAEUrEoICAAAAAAg
+2UBNAAANvrtoQ==
#EXT-X-ASSET:CAID=0x0000000020FB6501
#EXT-X-CUE-OUT:201.467
.
.
.
#EXT-X-CUE-OUT-CONT:ElapsedTime=5.939,Duration=201.467,SCTE35=/DA0AAAA+…AAg
+2UBNAAANvrtoQ==
.
.
.
#EXT-X-CUE-IN
```

SCTE-35 ad marker EXT-X-DATERANGE

This section is an addendum to the section called “SCTE-35 and SCTE-104 message processing”. Elemental Live supports the EXT-X-DATERANGE ad marker style in the manifest created for an HLS output.

To select this marker, do the following:

- In the event, go to **Output Group** > **Apple HLS** > **Advanced** > **Ad Markers** and select **EXT-X-DATERANGE**.

The structure and contents of the line are as described in HTTP Live Streaming, draft-pantos-http-live-streaming-23 (Version 23).

This marker style applies when the SCTE-35 messages are time_signal or splice_insert, and when the messages are passed through from the input or inserted at runtime using the REST API.

Example

This example shows an ad avail with an ID of 999 and a planned duration of 30 seconds. The first `EXT-X-DATERANGE`, for the start of the ad avail, includes a `START-DATE`. The second `EXT-X-DATERANGE` includes an `END-DATE` and a time that is 30.03 seconds later.

> **Note**
>
> The `DURATION`, `PLANNED-DURATION` and `END-DATE` tags are optional. If they are not present in the input or not specified in the REST API command, then these tags are omitted from the manifest.

```
#EXT-X-DATERANGE:ID="999",START-DATE="2018-08-22T21:54:00.079Z",PLANNED-
DURATION=30.000,
SCTE35-
OUT=0xFC30250000000000000FFF01405000003E77FEFFE0011FB9EFE002932E00001010100004D192A59
.
.
.
#EXT-X-DATERANGE:ID="999",END-DATE="2018-08-22T21:54:30.109Z",DURATION=30.030,
SCTE35-
IN=0xFC0000425100FFF01405000003000000000E77FEFFE0011FB9EFE0029004D1932E0000100101002A22
```

Working with SMPTE 2022-6

Elemental Live supports sources that are compliant with the SMPTE 2022-6 standard. The Elemental Live implementation of SMPTE 2022-6 provides an effective way to handle uncompressed video content. SMPTE 2022-6 uses standard IP networking to receive content, which means it uses a cheaper and more readily available network infrastructure than the traditional SDI protocol.

Elemental Live supports redundant inputs using SMPTE 20227, and non-redundant inputs.

With SMPTE 2022-6, the video, audio, and ancillary data are muxed into one feed. Compare this design to SMPTE 2110, where the content is each in a separate essence.

To work with SMPTE ST 2022-6 in Elemental Live, see Ingesting SMPTE 2022-6 content.

Working with SMPTE 2110

Elemental Live supports both inputs and outputs that are compliant with the SMPTE 2110 suite of standards.

The Elemental Live implementation of SMPTE 2110 provides an effective way to handle uncompressed and lightly compressed video content. SMPTE 2110 uses standard IP networking to send and receive content, which means it uses a cheaper and more readily available network infrastructure than the traditional SDI protocol.

This section provides general information about the capabilities that SMPTE 2110 offers. For detailed information about setting up SMPTE 2110 inputs, see the section called “SMPTE 2110 inputs”. For detailed information about setting up SMPTE 2110 outputs, see the section called “SMPTE 2110 output group”.

Separate streams

SMPTE 2110 separates the key media—video, audio, and ancillary data—into separate streams, or *essences*. This architecture cuts down on the transmission costs compared to a transport stream, for example. This architecture also means that there is less wasted processing (unnecessary demuxing) when ingesting the content. Elemental Live only receives the specific video, audio, and ancillary data required for a specific event.

Uncompressed and lightly compressed content

Elemental Live supports both video that is uncompressed, and video that is lightly compressed using the JPEG XS codec. The JPEG XS codec reduces the bitrate of the video content, typically without visible loss of video quality after multiple transcodes.

SMPTE 2110 and SDP files

The SMPTE 2110 specification relies on SDP files to describe the contents of the SMPTE 2110 streams. There is one file for each individual SMPTE 2110 stream. For more information about SDP files, see the section called "About SDP files".

Support for NMOS

Elemental Live supports NMOS IS-04 and IS-05 with both SMPTE 2110 inputs and outputs.

You can use NMOS to manage SMPTE 2110 streams. You can't use NMOS to manage other types of streams. For more information about NMOS, see the section called "Support for NMOS".

Conductor Live and SMPTE 2110 inputs with NMOS

You can use Conductor Live to set up SMPTE 2110 inputs either with or without NMOS. If you're using NMOS, the recommended procedure is to create a profile that includes a parameter for a device input. Then when you create the channel, you set the parameter to either an SDI device (for an SDI input) or to a SMPTE 2110 Receiver Group.

Conductor Live and SMPTE 2110 outputs

You can't use Conductor Live to produce SMPTE 2110 outputs that use NMOS. You can use AWS Elemental Conductor Live if you're not including NMOS.

Support for SMPTE 2022-7 – seamless protection switching

Elemental Live supports seamless protection switching (conforming with SMPTE 2022-7) for both SMPTE 2110 inputs and SMPTE 2110 outputs. Elemental Live uses SMPTE 2022-7 to implement resiliency via redundant streams.

For more information about SMPTE 2022-7, see the section called "Support for SMPTE 2022-7".

Topics

- Requirements for the appliance and network
- Supported content
- About SDP files
- Support for SMPTE 2022-7 – seamless protection switching
- Support for NMOS IS-04 stream discovery
- Setup: Remove bonded interfaces
- Setup: Reserve cores for SMPTE 2110
- Setup: Enable precision time protocol (PTP)

Requirements for the appliance and network

A SMPTE 2110 input or output requires an Elemental Live appliance with a high-speed network interface card (NIC). Therefore, to set up a SMPTE 2110 input or output, you must create the event on one of the following appliances.

Appliance	Network interface card (NIC)	Scope of support for SMPTE 2110	Support for NMOS
L800 series Elemental Live appliance	25 GbE NIC	Inputs and outputs	Supported
A bare-metal appliance	25 GbE Mellanox NIC. You must make sure that the NIC is licensed for use with the RiverMax SDK.	Inputs and outputs	Supported

Supported content

The following table describes the content that Elemental Live supports in SMPTE 2110 sources (inputs) and SMPTE 2110 outputs.

- For inputs, Elemental Live supports ingest of one instance of each type of stream—one video stream, zero or one audio stream, and zero or one ancillary data stream.
- For outputs, Elemental Live supports one video stream, zero or more audio streams, and zero or one ancillary data stream.

For detailed instructions for setting up a SMPTE 2110 input or output, see the section called “SMPTE 2110 inputs” and the section called “SMPTE 2110 output group”.

> **Note**
>
> Elemental Live can't ingest more than one audio stream in one SMPTE 2110 input.

Type	Direction	Details	Applicable standard
Video	Input and output	Uncompressed Resolutions – SD, HD, and 4K Scan types – Progressive and interlaced Sampling – 4:2:2 Bit format – 10-bit	SMPTE 2110-20
	Input and output	Lightly compressed with JPEG XS Resolutions – SD and HD resolution Scan types – Progressive and interlaced Sampling – 4:2:2 Bit format – 8-bit, 10-bit, 12-bit	SMPTE 2110-22
Audio	Input and output	PCM audio Uncompressed Sample rates: 44.1kHz and 48.0 kHz	SMPTE 2110-30
	Input and output	Dolby Digital (AC3)	SMPTE 2110-31

Type	Direction	Details	Applicable standard
		Coding modes – 1.0, 1+1, 2.0, 3.2 (with LFE)	
		Dolby Digital Plus (EAC3) Coding modes – 1.0, 2.0, 3.2	
	Output	Dolby Digital passthrough You can pass through Dolby Digital (AC3) from any input (SMPTE 2110 or another type) to a SMPTE 2110 output	
Ancillary data – Captions (optional)	Input and output	EIA-608 embedded captions CEA-708 embedded captions	SMPTE 2110-40
Ancillary data – Ad avail messages (optional)	Input	SCTE 104 messages. Elemental Live will automatically convert these messages to SCTE 35 messages during ingest.	SMPTE 2110-40

Type	Direction	Details	Applicable standard
	Output	SCTE 104 messages. If the source content has SCTE 35 messages, you can configure Elemental Live to convert them to SCTE 104 and include the SCTE 104 messages in the output.	
Timecode	Input	The SMPTE 2110 source must be Precision Time Protocol (PTP) locked. If it isn't locked, the video, audio, and ancillary data might not get synchroni zed properly during processing, resulting in unsynchronized media in all the outputs in the event.	SMPTE 2110-21

Type	Direction	Details	Applicable standard
	Output	You must enable PTP in Elemental Live so that the SMPTE 2110 outputs include RTP packet timestamp s. This timestamp synchronizes the video, audio, and ancillary data. It ensures that the output is PTP-locked.	

About SDP files

The SMPTE 2110 specification relies on SDP files to describe the contents of the SMPTE 2110 streams. There is one file for each type of stream. The file for each stream contains the following information:

- The source IP address for the file. This is an address that the sender controls.
- The unicast or multicast address that the receiver can connect to, in order to obtain the content.
- Descriptions of the media in the stream.

SDP files for content into Elemental Live

For SMPTE 2110 inputs, Elemental Live is the receiver and the upstream system is the sender.

- Typically, the sender (the upstream system) is responsible for creating the *source SDP file*.

  If the sender hasn't created the file, or if your organization is the originator of the stream and the file wasn't created by an automated system, then you can create the file. For more information, see the section called "About SDP files".
- If the source SMPTE 2110 streams aren't managed by NMOS, the source SDP files must be stored on a server that Elemental Live can access using HTTP. You and the sender should agree on a suitable location.

- If the source SMPTE 2110 streams are being managed by NMOS, you should have already set up the stream in the NMOS registry.

SDP files for output from Elemental Live

For SMPTE 2110 outputs, Elemental Live is the sender and the downstream system is the receiver.

- Elemental Live always generates the applicable *output SDP files* and places them on a directory on the appliance.
- If the SMPTE 2110 output group doesn't use NMOS, you must make the output SDP files available to the receiver. The files must be on an HTTP server that the receiver can access. For information about obtaining the SDP files, see the section called “Step 4: Download the SDP file”.
- If the SMPTE 2110 output group does use NMOS, Elemental Live handles delivery of the output SDP files available to the NMOS registry. There is nothing you need to do with the files.

The following topics describe the format of each type of SDP file, using examples.

> **Note**
>
> You are responsible for understanding the SMPTE 2110 specification. We provide the examples below only as a courtesy. Refer to the specification for complete information about the SDP file.

Topics

Format of an SDP file for video

```
v=0
o=- 3826217993 3826217993 IN IP4 10.xx.xxx.198
s=AWS Elemental SMPTE 2110 Output: [LiveEvent: 13] [OutputGroup: smpte_2110]
```

```
   [EssenceType_ID: 2110-20_video_198]
t=0 0
m=video 50000 RTP/AVP 96
c=IN IP4 239.x.x.x/64
b=AS:2568807
a=source-filter: incl IN IP4 239.x.x.x 10.xx.xxx.2
a=rtpmap:96 raw/90000
a=fmtp:96 sampling=YCbCr-4:2:2; width=1920; height=1080;
   exactframerate=60; depth=10; TCS=SDR; colorimetry=BT709;
   interlaced; PM=2110GPM; SSN=ST2110-20:2017; TP=2110TPN; PAR=1:1;
a=mediaclk:direct=0
a=ts-refclk:localmac=1c-34-da-5a-be-34
```

Following is information about the data in this example:

- `o=` This line identifies the source IP for the stream
- `m=video`. This identifies the file as a video description. `50000` is the port of the stream. This line must occur before the `b` and `a` lines..
- `c=` identifies the destination IP address. This is a unicast or multicast address.
- `b=AS:2568807` is the bandwidth in kilobytes, and applies only for JPEG XS.
- `a=source-filter` identifies a filter. This line is optional.

  In a file for an input to Elemental Live:

  - If the line is included, Elemental Live will listen only to the source IP (10.xx.xxx.2) for packets.
  - If the line isn't included, Elemental Live will listen to any packets on the destination IP address (239.x.x.x).

  In a file for an output: Elemental Live always includes this line in any SDP file that it creates.
- `Width` and `height` together specify the resolution.
- `Exactframerate` specifies the frames per second (fps).
- `Raw` is the video compression—`raw` for uncompressed video, `jxsv` for JPEG XS.
- `90000` is the frequency.

Format of an SDP file for audio

```
v=0
o=- 1443716955 1443716955 IN IP4 10.xx.xxx.236
```

```
s=st2110 0-1-0
t=0 0
m=audio 20000 RTP/AVP 97
c=IN IP4 239.x.x.x/64
a=source-filter: incl IN IP4 239.x.x.x 10.xx.xxx.236
a=rtpmap:97 L24/48000/2
a=mediaclk:direct=0 rate=48000
a=framecount:48
a=ptime:1
a=ts-refclk:ptp=IEEE1588-2008:04-5c-6c-ff-fe-0a-53-70:127
```

Following is information about the data in this example:

- `o=` This line identifies the source IP for the stream
- `m=audio`. This identifies the file as an audio description. `20000` is the port of the stream. This line must occur before the `a` lines.
- `c=` identifies the destination IP address. This is a unicast or multicast address.
- `a=source-filter` identifies a filter. This line is optional.

  In a file for an input to Elemental Live:

  - If the line is included, Elemental Live will listen only to the source IP (10.xx.xxx.236) for packets.
  - If the line isn't included, Elemental Live will listen to any packets on the destination IP address (239.x.x.x).

  In a file for an output: Elemental Live always includes this line in any SDP file that it creates.
- **a=rtpmap** provides information about the audio format:
  - An example for PCM:

    ```
    a=rtpmap:97 L24/48000/2
    ```

    `L24` is the number of bits per PCM audio sample. `48000` is the sample rate. 2 is the number of channels, in this example, a single stereo track.
  - An example for a Dolby Digital codec:

    ```
    a=rtpmap:96 AM824/48000/5 a=fmtp:96 channel-order=SMPTE2110.(AES3,AES3)
    ```

    `48000` is the sample rate. 6 is the number of AM824 subframe pairs, and must be an even number.

Format of an SDP file for ancillary data

```
v=0
o=- 1443716955 1443716955 IN IP4 10.xx.xxx.236
s=st2110 0-9-0
t=0 0
m=video 20000 RTP/AVP 100
c=IN IP4 239.x.x.xx/64
a=source-filter: incl IN IP4 239.x.x.xx 10.xx.xxx.236
a=rtpmap:100 smpte291/90000
a=fmtp:100 VPID_Code=133;
a=mediaclk:direct=0 rate=90000
a=ts-refclk:ptp=IEEE1588-2008:04-5c-6c-ff-fe-0a-53-70:127
```

Following is information about the data in this example:

- `o=` This line identifies the source IP for the stream
- `m=` The combination of `m=video` and `smpte291` (in the `a=rtpmap:` line) identifies this file as an ancillary data file. `20000` is the port of the stream. This line must occur before the `a` lines.
- `c=` identifies the destination IP address. This is a unicast or multicast address.
- `a=source-filter` identifies a filter. This line is optional.

  In a file for an input to Elemental Live:

  - If the line is included, Elemental Live will listen only to the source IP (10.xx.xxx.236) for packets.
  - If the line isn't included, Elemental Live will listen to any packets on the destination IP address (239.x.x.x).

  In a file for an output: Elemental Live always includes this line in any SDP file that it creates.

SDP file with 2022-7 information

If the source implements seamless protection switching (SMPTE 2022-7), it includes extra data in each applicable file, to provide information about the primary and secondary streams.

Keep in mind that it's possible that only some of the streams implement seamless protection switching, and some don't implement it.

Following is an example for the tags that identify video, as indicated by the line `m=video`. The information for the audio and ancillary streams follow the same pattern.

```
v=0
o=- 456221445 0 IN IP4 203.x.xxx.252
s=AJA Lily10G2-SDI 2110
t=0 0
a=group:DUP 1 2
m=video 20000 RTP/AVP 96
c=IN IP4 10.24.34.0/24
a=source-filter:incl IN IP4 192.x.x.1 198.xx.xxx.252
a=rtpmap:96 raw/90000
a=fmtp:96 sampling=YCbCr-4:2:2; width=1920; height=1080;
   exactframerate=30000/1001; depth=10; TCS=SDR; colorimetry=BT709;
   PM=2110GPM; SSN=ST2110-20:2017; TP=2110TPN; interlace=1;
a=ts-refclk:ptp=IEEE1588-2008:00-90-56-FF-FE-08-0F-45
a=mediaclk:direct=0
a=mid:1
m=video 20000 RTP/AVP 96
c=IN 10.24.34.0/24
a=source-filter:incl IN IP4 192.x.x.3 1198.xx.xxx.253
a=rtpmap:96 raw/90000
a=fmtp:96 sampling=YCbCr-4:2:2; width=1920; height=1080;
   exactframerate=30000/1001; depth=10; TCS=SDR; colorimetry=BT709;
   PM=2110GPM; SSN=ST2110-20:2017; TP=2110TPN; interlace=1;
a=ts-refclk:ptp=IEEE1588-2008:00-90-56-FF-FE-08-0F-45
a=mediaclk:direct=0
a=mid:2
```

Portions marked in bold are unique to SMPTE 2022-7.

- `a=group:DUP 1 2` indicates that seamless protection is being used, and identify which streams are the primary (1) and secondary (2).
- `a=mid:1` indicates that the lines that follow apply to the primary stream. The name value (1, in this example) must be identical to the first entry in the `a=group` tag.
- `a=mid:2` indicates that the lines that follow apply to the secondary stream.

Support for SMPTE 2022-7 – seamless protection switching

Elemental Live supports seamless protection switching (conforming with SMPTE 2022-7) for both SMPTE 2110 inputs and SMPTE 2110 outputs. The Elemental Live implementation of SMPTE 2022-7 provides protection against packet loss, interface failure, and network loss (because the two interfaces use different network paths).

- For inputs, if the source implements SMPTE 2022-7, you can configure some or all of the SMPTE 2110 streams in your event to accept inputs over two interfaces. Elemental Live will then perform seamless protection switching at the packet level, to ensure uninterrupted ingest of the content.

  For those streams, AWS Elemental Live receives two identical packet streams. When there is a problem with the first stream, AWS Elemental Live immediately uses the second stream to reconstruct the data, with no effect on the content.
- For outputs, you set up for SMPTE 2022-7 in the streams (Elemental Live outputs) in the SMPTE 2110 output group. Elemental Live will include two identical packet streams in each applicable stream.

Note that in both the inputs and the outputs, seamless protection switching might be implemented in some streams but not others. For example, it might be implemented in the video streams but not the audio or ancillary data streams. Compare this to NMOS, where all the streams in an input or output either use NMOS or don't use NMOS.

Support for NMOS IS-04 stream discovery

Elemental Live supports NMOS IS-04 with both SMPTE 2110 inputs and outputs.

> **Note**
>
> Currently, Elemental Live supports management of SMPTE 2110 streams using NMOS. Elemental Live doesn't support management of other types of streams.

SMPTE 2110 with NMOS

An NMOS solution includes an NMOS controller and an optional NMOS registry. If your solution includes a registry, you configure Elemental Live to communicate with that registry. If your solution doesn't include a registry, you must configure your NMOS controller to query Elemental Live

- Information about the SMPTE 2110 streams, including a unique ID for each stream.
- Information about the available *senders* and *receivers*. For a stream that Elemental Live outputs, Elemental Live is the sender. For a stream that Elemental Live ingests, it is the receiver.

SMPTE 2110 without NMOS

If you don't set up an NMOS solution, you still use SDP files:

- For a SMPTE 2110 input, you must identify the server where the SDP files are stored. This can be any HTTP server. When you configure the input, you specify which SDP files contain information about the SMPTE 2110 stream.
- For a SMPTE 2110 output, Elemental Live automatically creates the applicable SDP files. You must make these files accessible to the downstream system.

Setup: Remove bonded interfaces

Read this section if you plan to implement redundant inputs and/or outputs with SMPTE 2110.

SMPTE 2110 inputs and outputs support resiliency, but they do so using SMPTE 2022-7. You can’t implement input resiliency by bonding the two interfaces on the appliance. This resiliency implementation has the following impact:

- If you currently have a bonded interface on the appliance (on a Mellanox card or on a 25Gb card), you must remove the bond.
- In addition, if you have non-SMPTE 2110 (or non-SMPTE 2022-6) events set up on the appliance that use this bonded interface, you must modify the event configuration. You might be able to bond other interfaces on other cards in the appliance, and then use that bonded interface in those events. If you can’t do that, you must change the event configuration to not use a bonded interface.

If you don't plan to implement redundant SMPTE 2110 inputs or outputs (or SMPTE 2022-6 inputs), you can retain the bonded interface on the appliance.

Setup: Reserve cores for SMPTE 2110

Before you can run an event using SMPTE 2110, you must reserve cores on the appliance NIC. When you reserve these cores, AWS Elemental Live uses them only for processing SMPTE 2110 and/or SMPTE 2022-6.

The setting to reserve cores applies to the appliance. Therefore you need to reserve the cores only once, not every time you set up a new event that uses SMPTE 2110 or SMPTE 2022-6.

> **Important**
>
> Make sure you reserve the cores only when you plan to create events that have SMPTE 2110 inputs or outputs and/or SMPTE 2022-6 inputs.
> After you reserve the cores, AWS Elemental Live uses these cores only for processing SMPTE 2110 and/or SMPTE 2022-6. Other processing won't be able to use these cores.

To reserve cores

1. Find out which Ethernet interfaces on the appliance apply to your NIC. For example, eth4 and eth5.
2. Stop all events that are currently running on the appliance.
3. In the Elemental Live web interface, choose **Settings**. (Don't choose **Input Devices** or **Routers** from the submenu).
4. On the **Settings** page, choose the **Network** tab, then choose the **Network Devices** tab.
5. Choose the **edit** icon (pencil) next to the Ethernet interface that you identified, and select the **SMPTE 2110 and SMPTE 2022-6 Enabled** check box.
6. Repeat for the other Ethernet interface, if applicable.
7. Choose **Save**.
8. Stop and restart the service for your changes to take effect. You can do this in the web interface or in the command line interface (CLI).

    - In the web interface, go to the **Settings** tab, select **Stop Service** and then **Start Service**.

      OR
    - In the CLI, run `sudo service elemental_se restart`

Setup: Enable precision time protocol (PTP)

If you set up SMPTE 2110 outputs in events, you must enable Precision Time Protocol (PTP). You don't need to enable it if you only set up SMPTE 2110 inputs.

When PTP is enabled, you can view the PTP logs in `/var/log/messages`.

To enable PTP

You must enable PTP on the Elemental Live appliance. Enabling PTP automatically disables the NTP clock. And disabling PTP automatically enables the NTP clock.

1. In the Elemental Live web interface, choose **Settings**. (Don't choose **Input Devices** or **Routers** from the submenu).
2. On the **Settings** page, choose the **Network** tab, then choose **Hostname, DNS, & Timing Server**.
3. Select the **Enable PTP** check box, and then choose **Save**.

Working with PTP

PTP provides the timing for packet pacing in Elemental Live's SMPTE 2110 output. In line with the SMPTE 2110 specification, PTP is used for synchronizing the streams in SMPTE 2110 outputs—it ensures that the video, audio, and ancillary data streams remain synchronized downstream of Elemental Live.

Elemental Live is a follower that needs to sync to a boundary clock.

Elemental Live uses the Type N – Narrow sender profile of PTP. In the output video SDP file, the line `TP=2110TPN` identifies the profile.

The default configuration file

When you enable PTP, Elemental Live uses PTP as defined in the configuration file. This file is in `/etc/ptp4l.conf`.

The contents of the default PTP configuration file are shown below. You can modify the values and add more parameters to suit your network requirements. However, note that AWS Elemental won't be able to troubleshoot customizations that you make to the file.

Default PTP configuration file

```
[global]
domainNumber 127
priority1 128
priority2 128
use_syslog 1
logging_level 6
tx_timestamp_timeout 30
hybrid_e2e 0
dscp_event 46
dscp_general 46

[eth6]
logAnnounceInterval 0
announceReceiptTimeout 3
logSyncInterval -3
logMinDelayReqInterval -3
delay_mechanism E2E
network_transport UDPv4
```

Working with SRT

Elemental Live supports both inputs and outputs that use the SRT (secure reliable transport) protocol.

Elemental Live can ingest a transport stream (TS) that is sent from an SRT caller. In this scenario, the upstream system initiates the handshake that precedes transmission. AWS Elemental Live is the SRT listener that accepts or rejects the handshake. The transport stream source can be encrypted with AES.

To work with SRT inputs, see the section called “SRT Inputs”. To work with SRT outputs, see the section called “TS output using SRT”.

Implementing a trick-play track

Trick-play is used in digital video players to mimic some capabilities of analog players, including fast-forward and rewind capabilities. These capabilities often include a trick-play *track*—a visual cue for the person using the video player. In AWS Elemental Live, you can include track assets in an HLS output group. The downstream system for that output group can use these assets to implement the visual cue in their trick-play implementation.

Elemental Live provides two methods for including these assets:

- An I-frame-only manifest that conforms with the HLS specification.
- A trick-play track that conforms with the Image Media Playlist specification, version 0.4.

Choosing an implementation of trick-play track

You can follow one or both trick-play methods in the same output group.

Before you follow either method, contact the downstream system for the output group to find out how they implement trick-play. Find out the following:

- Can the downstream system support a trick-play track? If so, which trick-play specification does it follow?
- Is the supported implementation required or optional? Both of these implementations introduce specific lines into the HLS manifest. If the lines are absent, will the downstream system fail to handle the output from Elemental Live?

  It is likely that the downstream system considers both of these implementations to be optional.
- If you choose the I-frame-only manifest method, confirm that the downstream system supports the method according to the HLS specification. If the downstream system has a variation, it's possible that the downstream system won't be able to handle the output from Elemental Live. Elemental Live doesn't support customizations of the method.
- If you choose the image media playlist method, confirm that the downstream system supports the method according to the Image Media Playlist specification. If the downstream system has a variation, it's possible that the downstream system won't be able to handle the output from Elemental Live. Elemental Live doesn't support customizations of the implementation.

Topics

Trick-play track via I-frames

In an HLS output group, you can support a trick-play track by providing an I-frame-only manifest.

How the method works

When you create the HLS output group, you create one or more video outputs, in the usual way. In each video output, you enable the field to create an I-frame-only manifest that conforms to the HLS specification.

Elemental Live produces two child manifests for each encode (stream)—one manifest for handling the video in the usual way, and the I-frame-only manifest. The I-frame-only manifest lets the downstream player identify specific video frames to request, to construct the trick-play track. So this trick-play track method doesn't produce additional encodes in the output group.

Each I-frame-only manifest contains the following:

- One `#EXT-X-I-FRAMES-ONLY` tag, to indicate that the manifest is I-frame-only.
- Many `#EXT-X-BYTERANGE` entries. Each entry identifies the position of an I-frame position.

Setting up

You set up the trick-play track once for the entire HLS output group.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To set up an I-frame-only manifest

Include these steps when you create the HLS output group.

1. In the **HLS Output Group**, in **HLS Outputs**, choose **Add Output** to add an output. Or display an existing output.
2. In the output, choose **Advanced** to open that section, then select **Add I-frame Only Manifest**.
3. Set up the remaining fields in the output group as you normally would. Set up the video, audio, and captions outputs and encodes as you normally would.

Trick-play track via the Image Media Playlist specification

In an HLS output group, you can support a trick-play track by providing an asset that follows the Image Media Playlist specification, version 0.4. The Elemental Live implementation follows the time-based method of the specification. The specification is located here:

https://github.com/image-media-playlist/spec/blob/master/image_media_playlist_v0_4.pdf

Roku is one example of a platform that implements this specification.

How the method works

When you create the output group, you create standard outputs in the usual way for the video, audio, and captions encodes.

You also create one or more outputs that each contains one encode (stream). This encode is the asset that the downstream player can use to implement the trick-play track. The encode consists of a series of JPEG files, with one file for every video segment. This means that the capture follows the segmentation of the video encode, which means it provides the best experience on the video player.

Elemental Live creates a main manifest and child manifests in the usual way. The main manifest includes an `EXT-X-IMAGE-STREAM-INF` tag for the frame capture encode. The child manifest for the frame capture encode contains `EXT-X-IMAGES-ONLY` tags. The contents and format of these tags comply with the Image Media Playlist specification.

Setting up

You set up the trick-play track in the output group by creating an additional output that contains a video encode that consists of frame captures.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

To set up the frame capture encode in an HLS output group

To create a frame capture encode in an HLS output group, you create an output, and set up the video stream to use the **Frame Capture to JPEG** video codec.

1. In the **HLS Output Group**, in **HLS Outputs**, choose **Add Output** to add another output.
2. For that output, go to the Video section in the corresponding Stream section.
3. Choose **Video** and set up the video fields, including:
   - **Video Codec** – Choose **Frame Capture to JPEG**.
   - **Resolution** – Contact your downstream system to obtain the correct width and height. If you guess at the values, the experience on the downstream player might not be optimal.
   - **Trick Play** – Select this field. Many of the remaining fields disappear.
4. Choose **Audio 1** and choose **X** to remove the audio stream. The output must have only one encode (a video encode).

The output is part of the ABR stack and has the same destination as the other encodes in the HLS output group.

Virtual input switching

With the virtual input switching feature, you can set up a POIS to control switching from one AWS Elemental Live input to another. You can also switch inputs using dynamic input switching using the REST API. For more information, see the section called "Dynamic input switching".

Topics

About virtual input switching

Supported version of the specification

Elemental Live supports communications with a POIS according to this specification: OpenCable Specifications Alternate Content Real-time Event Signaling and Management API, OC-SP-ESAM-API-I03-131025.

The two virtual input switching features

There are two virtual input switching features. These features work in different ways and support different use cases. But they both require a POIS.

- Virtual input switching using asynchronous ESAM messages

  This type of input switching is based only on decisions from the POIS. We also refer to this switching as asynchronous input switching. This feature is available in Elemental Live version 2.23.0 and later.
- Virtual input switching using SCTE-35 messages

  This type of input switching relies on switch-related SCTE-35 messages that are in the input. We also refer to this switching as SCTE-35-triggered input switching. This feature is available in Elemental Live version 2.20.2 and later.

Virtual input switching requires the virtual input switcher license (the Virtual Input Switcher License add-on pack). Contact your AWS sales manager.

Choosing the switching feature to use

Typical use cases for the virtual input switching features are the following:

- SCTE-35-triggered input switching alone

  When your event consists of live MPTS programs, you can implement SCTE-35-triggered input switching. Upstream of Elemental Live, a system inserts SCTE-35 messages into the programs, and Elemental Live reacts to them during ingest.

  This scenario works well when you have well-orchestrated, predictable event.
- Asynchronous input switching alone

  When it's not feasible to implement SCTE-35-triggered input switching, you can implement asynchronous input switching. For example, if your event doesn't use sources that support SCTE-35-triggered input switching.

  With asynchronous input switching, the POIS sends SCTE-35 messages to Elemental Live. These messages contain instructions that Elemental Live reacts to.

Asynchronous input switching doesn't mean that the switching is unplanned. You can design a schedule and automate your POIS to send the switch requests.

- Both types of switching

  You might implement the SCTE-35-triggered switch scenario described previously, and also implement asynchronous input switching. To handle planned cutaways to file content, you can implement asynchronous input switching to handle unplanned cutaways to a file content. You can also implement synchronous input switching when you need to quickly drop the live stream and show a "please standby" file clip.

How virtual input switching works

Topics

Virtual input switching using asynchronous ESAM messages

Input types

Asynchronous input switching works only with the following sources:

- MPTS programs, from an MPTS input. MPTS programs are typically live sources.
- File inputs—any type of input that Elemental Live considers to be a file input. See the section called "Supported upstream systems"

If your event doesn't include these types of inputs, you can't set up for asynchronous input switching.

Setup

You set up the inputs in the Elemental Live event. You also set up the POIS with information about the same set of inputs. In other words, the inputs must be set up in both Elemental Live and the POIS.

You start the event. At any time, the POIS can send an asynchronous ESAM request to Elemental Live.

Principle of operation

The POIS sends a request to Elemental Live to request a switch away from the currently active input and to another input. The request can be seen as a switch away to another input. In addition, the originator of any potential action is the POIS.

Compare this principle of operation to SCTE-35-based triggers, where Elemental Live probes each non-active input for requests to switch to that non-active input. The request can be seen as a switch to this input. In addition, the originator of any potential action is Elemental Live.

Types of requests

The POIS can send one or more of the following requests:

- Prepare input. This request makes Elemental Live start processing the input before you switch to it. This preparation reduces the delay that always occurs between the switch request and the switch happening.
- Switch input. The POIS sends this request to instruct Elemental Live to switch, either immediately, or at a specific time.
- Handles SCTE-35 messages for ad avails. The POIS can request that Elemental Live insert an ad avail message. For information about the handling of these types of SCTE-35 messages, see the section called “POIS conditioning”.
- SCTE-35 messages that instruct Elemental Live whether to include the SCTE-35 in the output or not. The POIS can choose to instruct Elemental Live to include or to remove any of the three types of SCTE-35 messages.

Timing of input switching requests

The prepare input request should be sent at least 15 seconds before the switch request.

If the POIS sends both a prepare request and a switch request, the switch will occur in less than 4 seconds. After the switch occurs, Elemental Live stops preparing the input.

If the POIS sends a switch request but no prepare request, the input switch might take as long as 12 seconds.

Result in outputs

If the POIS instructs Elemental Live to include an SCTE-35 message in the output, that message becomes part of the stream. This rule applies to all types of messages – prepare input, switch input, and message relating to ad avails.

For a prepare input or switch input, Elemental Live still performs the action (prepare or switch), even if the POIS instructs Elemental Live not to include the message itself in the output.

Elemental Live includes these messages in output types that support SCTE-35 messages. For example, Elemental Live includes the messages in an HLS output.

For detailed information about how Elemental Live handles ad avail SCTE-35 messages, see the section called "POIS conditioning".

No queuing of requests

There is no way to create a queue of requests. Elemental Live can handle only one request. The POIS might send a request to switch to input A at 2:00:00.00. Then, before Elemental Live has performed that request, the POIS might send a request to switch to input B.

Even if the second request has a start time that is later than the first request, the second request will replace the first request in the request list.

Example

Here's an example of an input switching scenario that illustrates the different input switching capabilities.

The time is 2:00:00. You start the event and Elemental Live starts ingesting the first input, which is a file input.

Asynchronous request to a file input

At 2:01:00, the POIS sends an asynchronous request to switch to input B. Input B is the first in a series of 20-second ads.

Asynchronous request to switch to a live input

At 2:02:07, while these files are being processed, the POIS sends an asynchronous request to switch to input Y at 2:04:00.00. Elemental Live switches to input Y at the specified start time. Assume that input Y is a live TS input.

Canceled asynchronous request

This example shows cancellation of an asynchronous request.

At 2:51:44, the POIS sends a request to switch to input F at 2:52:00.

At 2:51:47, the POIS sends a request to switch to input Y (the input that is currently active) at 2:49:60.00. This request effectively cancels the switch to input F. Elemental Live can only have one request in its buffer, and the most recently received request always replaces any request already waiting.

Multiple requests in an unexpected order

At 2:53:48, the POIS sends a request to switch to input D at 2:57:00.

At 2:54:00, the POIS sends a request to switch to input E at 2:55:30.

Elemental Live ignores the request to switch to input D. Elemental Live can only have one request in its buffer, and the most recently received request always replaces any request already waiting.

Therefore at 2:55:30, Elemental Live switches to input E.

Virtual input switching using SCTE-35 messages

This type of input switching works with SCTE-35 messages that are in a transport stream source. When Elemental Live encounters an SCTE 35 message in the source, it sends the message to the POIS. The POIS might decide to send a request to switch input. Note that there is no sense of a particular type of SCTE 35 message that can trigger an input switch request. The POIS can request a switch in response to any type of SCTE 35 message that it receives.

Input types

SCTE-35-triggered input switching works only with the following sources:

- MPTS programs, from one MPTS input. MPTS programs are typically live sources.

Principle of operation

Elemental Live probes each non-active MPTS program for requests to switch to that non-active input. The request can be seen as a switch to this input. In addition, the originator of any potential action is Elemental Live.

Compare this principle of operation to asynchronous input switching, where the POIS sends a request to Elemental Live to request a switch away from the currently active input and to another input. The request can be seen as a switch away to another input. In addition, the originator of any potential action is the POIS.

Setup

In the event, you set up the inputs:

- You identify all the MPTS programs that you want to use, and set up each program as an individual input.

You also set up the POIS with information about the same set of inputs. In other words, the inputs must be set up in both Elemental Live and the POIS.

You start the event. As Elemental Live ingests the first input, it encounters SCTE-35 messages. It sends these messages to the POIS. It sends all SCTE-35 messages, not only switching-related messages. For example, it also sends ad avail messages.

For any switching-related message, the POIS might decide to send a request to Elemental Live to switch to another input at a specific time. The POIS can send this instruction at any time. There is no timeout between receiving the information and sending an instruction. The POIS might also decide not to send an instruction.

Types of requests

The following are the types of requests:

- Prepare input. This request makes Elemental Live start processing the input, even before the input is active. This preparation reduces the delay that always occurs between the switch request and the switch happening.
- Switch input. The POIS sends this request to instruct Elemental Live to switch, either immediately, or at a specific time.

  The prepare input request should be sent at least 15 seconds before the switch request.

  If the POIS sends a prepare input and a switch input, the switch will occur in less than 4 seconds.

  If the POIS doesn't send a prepare input, the input switch can take as long as 12 seconds.

- SCTE-35 messages for ad avail handling. The POIS can request that Elemental Live insert an ad avail message, or delete or modify an ad avail message that is in the source stream (and that Elemental Live therefore sent to the POIS). For information about the handling of these types of SCTE-35 messages, see the section called "POIS conditioning".
- SCTE-35 messages that instruct Elemental Live whether to include the SCTE-35 in the output or not. The POIS can choose to instruct Elemental Live to include or to remove any of the three types of SCTE-35 messages.

Timing of input switching requests

The prepare input request should be sent at least 15 seconds before the switch request.

If the POIS sends a prepare input and a switch input, the switch will occur in less than 4 seconds. After the switch occurs, Elemental Live stops preparing the input.

If the POIS doesn't send a prepare input, the input switch might take as long as 12 seconds.

Result in outputs

If the POIS instructs Elemental Live to include an SCTE-35 message in the output, that message becomes part of the stream. This rule applies to all types of messages – prepare input, switch input, and messages relating to ad avails.

For a prepare input or switch input, Elemental Live still performs the action (prepare or switch), even if the POIS instructs Elemental Live not to include the message itself in the output.

Elemental Live includes these messages in output types that support SCTE-35 messages. For example, Elemental Live includes the messages in an HLS output.

For detailed information about how Elemental Live handles ad avail SCTE-35 messages, see the section called "POIS conditioning".

Example

Here's an example of an input switching scenario that illustrates the different input switching capabilities.

The time is 2:00:00. You start the event and Elemental Live starts ingesting the first input. At the same time, Elemental Live reads all the other inputs,

SCTE-35-triggered request

At 2:17:33, while ingesting input Y, Elemental Live encounters an SCTE-35 message. It sends the message to the POIS. Shortly after, the POIS sends a request to switch to input Z at 2:30:00.

At 2:30:00, Elemental Live switches to input Z (another live input).

SCTE-35 message that the POIS ignores

At 2:37:00, Elemental Live encounters an SCTE-35 message. It sends the message to the POIS, but the POIS doesn't send an input switch request. There is no rule that dictates that the POIS must switch on receiving a message.

Multiple SCTE-35-triggered requests

At 2:40:02, Elemental Live encounters another SCTE-35 message, and this time the POIS does respond by sending two requests. One request is to switch at 2:45:13 to input B (one of the ads that played previously). The second request is to switch at 2:45:43 back to input Z.

At 2:45:13, Elemental Live switches to input B.

At 2:45:43, Elemental Live switches back to input Z.

Canceled asynchronous request

At 2:51:44, the POIS sends a request to switch to input F at 2:52:00.

At 2:51:47, the POIS sends a request to cancel the switch to input F.

This example shows cancellation of an asynchronous request. The POIS can cancel either an asynchronous request of a SCTE-35-triggered request.

Multiple requests in an unexpected order

At 2:53:48, the POIS sends a request to switch to input D at 2:57:00.

At 2:54:00, the POIS sends a request to switch to input E at 2:55:30.

At 2:55:30, Elemental Live switches to input E. Then at 2:57:00, Elemental Live switches to input D.

Note that Elemental Live doesn't ignore the request to switch to input D, even though it received the request for input E in the meantime.

Setting up for virtual input switching

With virtual input switching, the Elemental Live event and the POIS must be set up with identical information.

The setup steps are the same for asynchronous input switching and SCTE-35-triggered input switching.

Topics

Overview

The POIS must be set up with the following information:

- A tag for the Elemental Live input. Each input must be unique in the Elemental Live event.
- The acquisition point ID and zone for the Elemental Live event.

In Elemental Live, you set up the event with the following information:

- Acquisition point ID. This ID is how the POIS identifies the Elemental Live node. The POIS provides you with this ID.
- Zone. This ID is how the POIS identifies the specific event. The POIS provides you with this ID.
- These two fields set up Elemental Live and the POIS to have a common identifier for the event.

In the event, you set up the inputs with the following information:

- Input ID. This is the standard ID that Elemental Live generates and assigns when you save an event.
- The URL for the input.
- Label. This tag has the same value as the label (in Elemental Live). In this way, both Elemental Live and the POIS have a common identifier for each input.

- A virtual tag. This identifies the input as one that the POIS knows about. Typically, you set up all the inputs in an event as virtual.

Number of nodes and number of POIS

- Each Elemental Live node can communicate with only one POIS. All the events on the Elemental Live node must be configured for the same POIS.
- One POIS can communicate with several Elemental Live nodes. The POIS uses the POIS AcquisitionPointID and ZoneID parameter to uniquely identify each Elemental Live node.

Number and type of inputs

With asynchronous input switching, the inputs can be any of the following:

- Programs in one MPTS. You can set up a maximum of 11 programs.
- Any type of input that Elemental Live considers to be a file input. See the section called "Supported upstream systems".

  There is no limit to the number of inputs of this type.

With SCTE-35-triggered input switching, the inputs can be any of the following:

- Programs in one MPTS. You can set up a maximum of 11 programs.

Step 1: Coordinate with the POIS operator

Talk to the POIS operator, and agree on the values for this data:

- The label for each input. This value must be identical, and it is case sensitive.
- The acquisition ID for the Elemental Live node. The POIS must have a different acquisition ID for each Elemental Live node that it works with.
- The zone ID for the specific event.

  The combination of acquisition ID and zone ID must be unique in the Elemental Live event.
- The URL for the signal conditioner endpoint on the POIS. This endpoint handles the input switching communications with Elemental Live.
- The URL for the alternate signal conditioner endpoint on the POIS, if this exists.

- The preroll that is the number of seconds between when Elemental Live receives the request from the POIS and when Elemental Live inserts any SCTE-35 messages in the content. This preroll isn't required if the messages that the POIS sends include a start time for the input switch,

Make a note of all of this data. You will need it to set up on Elemental Live.

Step 2: Set up the event

These steps show you how to configure the event with information about the POIS server.

The information in this section assumes that you are familiar with the general steps for creating an event.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

1. On the web interface, open the **Ad Avail Controls** section of the event.
2. Complete the fields as follows:
   - **Ad Avail Trigger**: ESAM.
   - **Acquisition Point Identifier**: The value that you obtained from the POIS operator.
   - **Zone Identity**: The value that you obtained from the POIS operator.
   - **Signal Conditioner Endpoint**: The URL that you obtained from the POIS operator.
   - **Alternate Signal Conditioner Endpoint**: The URL that you obtained from the POIS operator, if any.
   - **Response Signal Preroll**: The value you obtained from the POIS operator, if any.
   - Other fields: The other fields in this section aren't used for virtual input switching.
3. Complete this field only if you want to enable asynchronous input switching:
   - **Unsolicited ESAM Server**.

Step 3: Set up the inputs in the event

To set up an MPTS program

You must set up each program in the MPTS as an individual input.

You can use inputs from an MPTS with both asynchronous input switching and SCTE-35-triggered input switching.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

1. On the web interface, got to the **Inputs** section of the event.
2. Complete the **Input** section as follows:
    - **Input Type**: Choose Network Input.
    - **Input Name**: Enter the label for the input, obtained from the POIS operator.
    - **Network Location**: The URL of the MPTS. For example, udp://192.20.08.18:5001
3. Open the **Advanced** section of this input and complete the following fields (on the first line of fields)
    - **Virtual Input**: Choose this field to set up the input as a virtual input. Elemental Live notifies the POIS only about inputs that are configured as virtual inputs.
    - **Virtual Input SCTE 35 PID**: Enter the PID in the MPTS that holds the SCTE-35 data stream. You complete this field only if you are using this program with SCTE-35-triggered input switching. This is the PID that Elemental Live probes for SCTE-35 messages to send to the POIS.
4. Go to the **Video Selector** section of this input and complete the following fields:
    - **Program**, in **Video Selector**: Enter the program number of the individual program in the mpts.
5. Set up the audio and captions for this MPTS in the usual way.
6. Set up more inputs, for other programs in the MPTS. For all the inputs, these rules apply:
    - The URLs must be identical, because the programs must all be in the same MPTS.

- The Input names must each be different.
- The Program (in the Video Selector) must each be different.

To set up a file input

You can use file inputs with asynchronous input switching. You can't use them with SCTE-35-triggered input switching.

> **Note**
>
> The information in this section assumes that you are familiar with the general steps for creating an event.

1. On the web interface, got to the **Inputs** section of the event.
2. Complete the **Input** section as follows:
   - **Input Type**: Choose File Input or HLS File Input.
   - **Input Name**: Enter the label for the input, obtained from the POIS operator.
   - **File Location**: The path and file name for the source. For example, ftp://vod_files/mlaw.wav
3. Open the **Advanced** section of this input and complete the fields as follows:
   - **Virtual Input**: Choose this field to set up the input as a virtual input. Elemental Live notifies the POIS only about inputs that are configured as virtual inputs.
   - **Virtual Input SCTE 35 PID**: Ignore this field. It is used only with SCTE-35-triggered input switching.

Step 4: Start the event

When you start the event, Elemental Live ingests inputs, starting with the first input listed in the event. While the event is running, Elemental Live responds to POIS requests to switch to a different input. The role of Elemental Live is reactive. The POIS controls input switching.

Reference Information

This chapter provides reference information about AWS Elemental Live.

Topics

Reference: Supported captions

This section provides reference information about the captions formats that Elemental Live supports.

Topics

Supported caption formats

Caption	Supported in input	Supported in output	Description
Ancillary data	√		• From MXF input, data that is compliant with “SMPTE 291M: Ancillary Data Package and Space Formatting” and that is contained in ancillary data. • From QuickTime input or for

Caption	Supported in input	Supported in output	Description
			QuickTime output, data that is compliant with EIA-608 (also known as CEA-608) or CEA-708 (also known as EIA-708) and that is contained in ancillary data.
Ancillary+Embedded		√	For QuickTime output only, the output combines captions in ancillary data and embedded captions. The ancillary captions are compliant with EIA-608 (also known as CEA-608) or CEA-708 (also known as EIA-708). The embedded captions are described later in this table.
ARIB	√	√	Captions that are compliant with the ARIB STD-B37 Version 2.4.

Caption	Supported in input	Supported in output	Description
Burn-in	N/A	√	From input: It is technically impossibl e for Elemental Live to read burn-in captions. Therefore , from an input viewpoint, they cannot be considered to be captions. For output: Burn-in captions are captions that are converted into text and then overlaid on top of the picture directly in the video stream.
DVB-Sub	√	√	Captions that are compliant with ETSI EN 300 743.
EBU-TT-D		√	Captions that are compliant with EBU Tech 3380, EBU-TT-D Subtitling Distribut ion Format, 2018.

Caption	Supported in input	Supported in output	Description
Embedded	√	√	Captions that are compliant with the EIA-608 standard (also known as CEA-608 or SMPTE-259M or “line 21 captions”) or the CEA-708 standard (also known as EIA-708).
Embedded+SCTE-20	√	√	Captions that have both embedded and SCTE-20 in the video. The embedded captions are inserted before the SCTE-20 captions.
RTMP CaptionInfo		√	Captions that are compliant with the Adobe onCaptionInfo format.
RTMP CuePoint		√	Captions that are in the cuePoint format.
SCC	√	√	Captions that are in the Scenarist format, file extension .scc.

Caption	Supported in input	Supported in output	Description
SCTE-20	√		Captions that are compliant with the standard “SCTE 20 2012 Methods for Carriage of CEA-608 Closed Captions and Non-Real Time Sampled Video.”
SCTE-20+Embedded		√	Captions that are compliant with SCTE-43. The SCTE-20 captions are inserted in the video before the Embedded captions.
SCTE-27	√		Captions that are compliant with the standard “SCTE-27 (2011), Subtitlin g Methods for Broadcast Cable.”
SMI	√	√	Captions in the Microsoft SAMI format.
SMPTE-TT		√	Captions that are compliant with the standard “SMPTE ST 2052-1:2010.”
SRT	√	√	Captions in the SRT format.

Caption	Supported in input	Supported in output	Description
STL	√		Captions in the EBU STL format. Spruce STL format is not supported.

Caption	Supported in input	Supported in output	Description
Teletext	√	√	From SDI input. Captions in: • OP42 teletext format. SMPTE 2031 field is unchecked in source. • OP47 teletext format wrapped in a SMPTE-2031 envelope. SMPTE 2031 field is checked in source. • OP47 teletext format, also known as SMPTE RDD-08 (compliant with ITU-R BT.1120-7). SMPTE 2031 field is unchecked in source. From TS input: Captions in the EBU Teletext format. From MXF file input: OP47 teletext format, also known as SMPTE RDD-08 (compliant with ITU-R BT.1120-7).

Caption	Supported in input	Supported in output	Description
			SMPTE 2031 field is unchecked in source. For output: Captions in the EBU Teletext format.
TTML	√	√	Caption files that are compliant with the standard “Timed Text Markup Language 1 (TTML1) (Second Edition).”
WebVTT		√	Captions that are compliant with “webvtt: The Web Video Text Tracks Format” (http://dev.w3.org/html5/webvtt/).

Rules for extracting captions from sources

To use captions in a source, Elemental Live must be able to extract the captions. The rules are as follows:

- Elemental Live can always extract sidecar captions from the source, so long as Elemental Live supports the captions format.
- Elemental Live can always support captions from a streaming source, so long as Elemental Live supports the captions format and the input type.

- Elemental Live can't necessarily extract captions from a file source. Even if Elemental Live supports the captions format, it can extract the captions only from specific container types. See the table that follows.

Container in file input	Elemental Live can extract captions from the container?
Adobe Flash	
Audio Video Interleave (AVI)	
HLS	Yes
Matroska	
MP4	Yes
MPEG Transport Stream (TS)	Yes
MPEG-1 System Stream	
MXF	Yes
No container	Yes
QuickTime	Yes
WAV	Yes

Rules for converting source captions to output captions

This section helps you to ensure that when creating an event or profile, you select a format that is valid for your output captions.

Various constraints exist for the caption formats that you can include in your content:

- For a given input container, Elemental Live can parse certain caption formats.

- For a given input caption format, Elemental Live can create one or more output captions. However, a given output caption can appear only in specific output containers.

In summary – starting with the output container you want to produce:

- A given output container supports a given set of output caption formats. This container constrains your choice of output captions.
- Furthermore, to produce each output caption format, you must use one of the compatible input caption formats .And you must select an input which can appear in the input container you have selected. So both which original input container and format you choose constrains your choice of output caption formats.

You must determine if it's possible for you to include the captions format that you want in your output. For example, assume that you want to include WebVTT captions in an HLS output. Assume that your captions source is in an MP4 container.You can implement this use case only if the MP4 container holds 608 embedded captions. You can't implement if, for example, the MP4 container holds ancillary captions.

Topics

Supported source captions and output captions in a GPP output container

To read this table, find the type of container and captions from your input. The supported captions formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in
	SCTE-20	Burn-in
MP4 Container	Embedded	Burn-in
	SCTE-20	Burn-in
MXF Container	Embedded	Burn-in
	Ancillary Data	Burn-in
	Teletext	Burn-in
QuickTime Container	Embedded	Burn-in
	Ancillary Data	Burn-in
Raw Container	SRT	Burn-in
	SMI	Burn-in
	TTML	Burn-in
	STL	Burn-in
	SCC	Burn-in
RTMP Container	Embedded	Burn-in
SDI Container	Embedded	Burn-in
	Teletext	Burn-in
	ARIB	None

Source input container	Source caption format	Supported output captions
MPEG2-TS Container	Embedded	Burn-in
	SCTE-20	Burn-in
	Teletext	Burn-in
	ARIB	None
	DVB-Sub	Burn-in
	SCTE-27	Burn-in

Supported source captions and output captions in a DASH output container

To read this table, find the type of container and captions from your input. The supported captions formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCTE-20	Burn-in, SMPTE-TT, TTML, EBU-TT-D
MP4 Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCTE-20	Burn-in, SMPTE-TT, TTML, EBU-TT-D
MXF Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Ancillary Data	Burn-in, SMPTE-TT, TTML, EBU-TT-D

Source input container	Source caption format	Supported output captions
	Teletext	Burn-in, SMPTE-TT, TTML, EBU-TT-D
QuickTime Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Ancillary Data	Burn-in, SMPTE-TT, TTML, EBU-TT-D
Raw Container	SRT	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SMI	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	TTML	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	STL	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCC	Burn-in, SMPTE-TT, TTML, EBU-TT-D
RTMP Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
SDI Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Teletext	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	ARIB	None
MPEG2-TS Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D

Source input container	Source caption format	Supported output captions
	SCTE-20	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Teletext	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	ARIB	None
	DVB-Sub	Burn-in, SMPTE-TT
	SCTE-27	Burn-in, SMPTE-TT

Supported source captions and output captions in an HLS output container

To read this table, find the type of container and captions from your input. The supported captions formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, Embedded, Web-VTT
	SCTE-20	Burn-in, Embedded, Web-VTT
MP4 Container	Embedded	Burn-in, Embedded, Web-VTT
	SCTE-20	Burn-in, Embedded, Web-VTT
MXF Container	Embedded	Burn-in, Embedded, Web-VTT
	Ancillary Data	Burn-in, Embedded, Web-VTT
	Teletext	None
QuickTime Container	Embedded	Burn-in, Embedded, Web-VTT
	Ancillary Data	Burn-in, Embedded, Web-VTT
Raw Container	SRT	Burn-in, Web-VTT

Source input container	Source caption format	Supported output captions
	SMI	Burn-in, Web-VTT
	TTML	Burn-in, Web-VTT
	STL	Burn-in, Web-VTT
	SCC	Burn-in, Embedded, Web-VTT
RTMP Container	Embedded	Burn-in, Embedded, Web-VTT
SDI Container	Embedded	Burn-in, Embedded, Web-VTT
	Teletext	Burn-in, Web-VTT
	ARIB	None
MPEG2-TS Container	Embedded	Burn-in, Embedded, Web-VTT
	SCTE-20	Burn-in, Embedded, Web-VTT
	Teletext	Burn-in, Web-VTT
	ARIB	None
	DVB-Sub	Burn-in, Web-VTT
	SCTE-27	Burn-in, Web-VTT

Supported source captions and output captions in an fMP4 output container

To read this table, find the type of container and captions from your input. The supported captions formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Web-VTT
	SCTE-20	Web-VTT

Source input container	Source caption format	Supported output captions
MP4 Container	Embedded	Web-VTT
	SCTE-20	Web-VTT
MXF Container	Embedded	Web-VTT
	Ancillary Data	Web-VTT
QuickTime Container	Embedded	Web-VTT
	Ancillary Data	Web-VTT
Raw Container	SRT	Web-VTT
	SMI	Web-VTT
	TTML	Web-VTT
	STL	Web-VTT
	SCC	Web-VTT
RTMP Container	Embedded	Web-VTT
SDI Container	Embedded	Web-VTT
	Teletext	Web-VTT
MPEG2-TS Container	Embedded	Web-VTT
	SCTE-20	Web-VTT
	Teletext	Web-VTT

Supported source captions and output captions in an MP4 output container

To read this table, find the type of container and captions from your input. The supported captions formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
MP4 Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
MXF Container	Embedded	Burn-in, Embedded
	Ancillary Data	Burn-in, Embedded
	Teletext	Burn-in, Embedded
QuickTime Container	Embedded	Burn-in, Embedded
	Ancillary Data	Burn-in, Embedded
Raw Container	SRT	Burn-in
	SMI	Burn-in
	TTML	Burn-in
	STL	Burn-in
	SCC	Burn-in, Embedded
RTMP Container	Embedded	Burn-in, Embedded
SDI Container	Embedded	Burn-in, Embedded
	Teletext	Burn-in
MPEG2-TS Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
	Teletext	Burn-in

Source input container	Source caption format	Supported output captions
	DVB-Sub	Burn-in
	SCTE-27	Burn-in

Supported source captions and output captions in MPEG2-TS or MPEG2-UDP

The table provides information about captions in an MPEG2-TS file output container or MPEG2-UDP streaming output container.

To read this table, find the type of container and captions from your input. The supported caption formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	SCTE-20	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
MP4 Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	SCTE-20	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
MXF Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded

Source input container	Source caption format	Supported output captions
	Ancillary Data	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	Teletext	Burn-in, DVB-Sub, Teletext
QuickTime Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	Ancillary Data	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
Raw Container	SRT	Burn-in, DVB-Sub
	SMI	Burn-in, DVB-Sub
	TTML	Burn-in, DVB-Sub
	STL	Burn-in, DVB-Sub
	SCC	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
RTMP Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
SDI Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	Teletext	Burn-in, DVB-Sub, Teletext
	ARIB	ARIB

Source input container	Source caption format	Supported output captions
MPEG2-TS Container	Embedded	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	SCTE-20	Burn-in, DVB-Sub, Embedded, Embedded+SCTE-20, SCTE-20+Embedded
	Teletext	Burn-in, DVB-Sub, Teletext
	ARIB	ARIB
	DVB-Sub	Burn-in, DVB-Sub
	SCTE-27	Burn-in, DVB-Sub

Supported source captions and output captions in an MSS output container

To read this table, find the type of container and captions from your input. The supported caption formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCTE-20	Burn-in, SMPTE-TT, TTML, EBU-TT-D
MP4 Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCTE-20	Burn-in, SMPTE-TT, TTML, EBU-TT-D
MXF Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D

Source input container	Source caption format	Supported output captions
	Ancillary Data	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Teletext	Burn-in, SMPTE-TT, TTML, EBU-TT-D
QuickTime Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Ancillary Data	Burn-in, SMPTE-TT, TTML, EBU-TT-D
Raw Container	SRT	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SMI	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	TTML	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	STL	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCC	Burn-in, SMPTE-TT, TTML, EBU-TT-D
RTMP Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
SDI Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Teletext	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	ARIB	None

Source input container	Source caption format	Supported output captions
MPEG2-TS Container	Embedded	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	SCTE-20	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	Teletext	Burn-in, SMPTE-TT, TTML, EBU-TT-D
	ARIB	None
	DVB-Sub	SMPTE-TT
	SCTE-27	SMPTE-TT

Supported source captions and output captions in an MXF output container

To read this table, find the type of container and captions from your input. The supported caption formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, Embedded
SCTE-20	Burn-in	Embedded
MP4 Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
MXF Container	Embedded	Burn-in, Embedded
	Ancillary Data	Burn-in, Embedded
	Teletext	Burn-in
QuickTime Container	Embedded	Burn-in, Embedded

Source input container	Source caption format	Supported output captions
Ancillary Data	Burn-in	Embedded
Raw Container	SRT	Burn-in
	SMI	Burn-in
	TTML	Burn-in
	STL	Burn-in
	SCC	Burn-in, Embedded
RTMP Container	Embedded	Burn-in, Embedded
SDI Container	Embedded	Burn-in, Embedded
	Teletext	Burn-in
	ARIB	None
MPEG2-TS Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
	Teletext	Burn-in
	ARIB	None
	DVB-Sub	Burn-in
	SCTE-27	Burn-in

Supported source captions and output captions in a QuickTime output container

To read this table, find the type of container and captions from your input. The supported caption formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
	SCTE-20	Burn-in, Embedded, Embedded+Ancillary
MP4 Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
	SCTE-20	Burn-in, Embedded, Embedded+Ancillary
MXF Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
	Ancillary Data	Burn-in, Embedded, Embedded+Ancillary
	Teletext	Burn-in
QuickTime Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
	Ancillary Data	Burn-in, Embedded, Embedded+Ancillary
Raw Container	SRT	Burn-in
	SMI	Burn-in
	TTML	Burn-in
	STL	Burn-in
	SCC	Burn-in, Embedded, Embedded+Ancillary

Source input container	Source caption format	Supported output captions
RTMP Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
SDI Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
	Teletext	Burn-in
	ARIB	None
MPEG2-TS Container	Embedded	Burn-in, Embedded, Embedded+Ancillary
	SCTE-20	Burn-in, Embedded, Embedded+Ancillary
	Teletext	Burn-in
	ARIB	None
	DVB-Sub	Burn-in
	SCTE-27	Burn-in

Supported source captions and output captions in a raw output container

This table describes the caption formats that can be included in a *raw output container that contains video*. For information about support when the captions are in a raw container on their own (independent of video), see [Supported source captions and output captions in a captions-only output container](#).

To read this table, find the type of container and captions from your input. The supported caption formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, Embedded

Source input container	Source caption format	Supported output captions
	SCTE-20	Burn-in, Embedded
MP4 Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
MXF Container	Embedded	Burn-in, Embedded
	Ancillary Data	Burn-in, Embedded
	Teletext	Burn-in
QuickTime Container	Embedded	Burn-in, Embedded
	Ancillary Data	Burn-in, Embedded
Raw Container	SRT	Burn-in
	SMI	Burn-in
	TTML	Burn-in
	STL	Burn-in
	SCC	Burn-in, Embedded
RTMP Container	Embedded	Burn-in, Embedded
SDI Container	Embedded	Burn-in, Embedded
	Teletext	Burn-in
	ARIB	None
MPEG2-TS Container	Embedded	Burn-in, Embedded
	SCTE-20	Burn-in, Embedded
	Teletext	Burn-in

Source input container	Source caption format	Supported output captions
	ARIB	None
	DVB-Sub	Burn-in
	SCTE-27	Burn-in

Supported source captions and output captions in an RTMP output container

To read this table, find the type of container and captions from your input. The supported caption formats for this *output* container are then shown in the last column.

Source input container	Source caption format	Supported output captions
HLS Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	SCTE-20	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
MP4 Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	SCTE-20	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
MXF Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	Ancillary Data	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	Teletext	Burn-in, RTMP CuePoint
QuickTime Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint

Source input container	Source caption format	Supported output captions
	Ancillary Data	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
Raw Container	SRT	Burn-in, RTMP CuePoint
	SMI	Burn-in, RTMP CuePoint
	TTML	Burn-in, RTMP CuePoint
	STL	Burn-in, RTMP CuePoint
	SCC	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
RTMP Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
SDI Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	Teletext	Burn-in, RTMP CuePoint
	ARIB	
MPEG2-TS Container	Embedded	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	SCTE-20	Burn-in, Embedded, RTMP CaptionInfo, RTMP CuePoint
	Teletext	Burn-in, RTMP CuePoint
	ARIB	None
	DVB-Sub	Burn-in
	SCTE-27	Burn-in

Supported source captions and output captions in a captions-only output container

This table describes the caption formats that can be included on their own in an output. With this option, *the container is always a raw container that contains only the captions* (video would be in another container that may be a raw container or may be some other type).

If you have one of the source caption formats listed in the first column – regardless of the source container – you can convert it to an external captions file and include it in a raw container that contains only that captions file.

Source input container	Source caption format	Supported output captions
Any container	SRT	SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	SMI	SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	TTML	SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	SMPTE-TT	SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	STL	SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	Embedded	SCC, SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	SCC	SCC, SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	SCTE-20	SCC, SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	SCTE-20+Embedded	SCC, SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT

Source input container	Source caption format	Supported output captions
	Embedded+SCTE-20	SCC, SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	Ancillary Data	SCC, SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	Teletext	SMI, SMPTE-TT, SRT, TTML, EBU-TT-D, Web-VTT
	DVBSub	SMPTE-TT
	SCTE-27	SMPTE-TT

Reference: Supported DRM solutions

This chapter lists the Digital Rights Management (DRM) solutions that are available for the types of outputs supported by AWS Elemental Live. The tables are organized alphabetically by output type and the DRM technology provider name.

Topics

DASH output

Encryption mode: Always AES CTR (AES-128)

Key rotation: Always Static

Support client players: Consult with the DRM solution provider (for SPEKE) or the DRM Technology provider for supported players.

Description	DRM technology provider	Key provider (DRM implementer)	Version of server API from DRM implementer
The customer uses a SPEKE-compliant DRM solution for protecting DASH output using Widevine/CENC and PlayReady/CENC technology.	CENC Widevine and PlayReady	SPEKE	SPEKE v1.0
The customer uses a static PlayReady key for protecting DASH output using PlayReady/CENC technology.	PlayReady	PlayReady	Not applicable
The customer uses a static Widevine key for protecting DASH output using Widevine/CENC technology.	CENC/Widevine	Generic	Not applicable
The customer uses the Piksel DRM solution for protecting DASH output using the Widevine/CENC standard. The end user will play the content on a player	CENC/Widevine	Piksel	GetEncryptInfo v1.0

Description	DRM technology provider	Key provider (DRM implementer)	Version of server API from DRM implementer
that is Widevine/ CENC compliant and Piksel-approved.			

HLS output with Apple FairPlay

Encryption mode: Always AES CBC (Sample AES)

Supported client players: Consult with the DRM solution provider (for SPEKE) or the key provider (DRM implementer) for supported players. For a self-generated key, the iOS players must be able to retrieve the key. For a generic key, the iOS player must be able to retrieve the key from the manifest.

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses a SPEKE-compliant DRM solution for protecting HLS fMP4 output using Apple Fairplay DRM technology.	SPEKE	SPEKE v1.0	Static, Rotating
The customer uses a SPEKE-compliant DRM solution for protecting HLS output using Apple Fairplay DRM technology.	SPEKE	SPEKE v1.0	Static, Rotating

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the 1Mainstream DRM solution for protecting HLS output using the Apple Fairplay DRM technology. The end user plays the content on a 1Mainstream-approved player.	1Mainstream	Version 1.1	Static
Elemental Live generates a key that it uses to encrypt the content. Elemental Live also puts that key at a customer-specified location; the client player retrieves the key from that location and decrypts the content. The end user plays the content on an iOS player. Strictly speaking, this is an encryption solution, not a DRM solution.	Self-Generated	Not applicable; key generated by Elemental Live	Static, Rotating

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
Each customer has a key server. Elemental Live places the keys inside the manifest that is delivered with the content; the client player obtains the key from the manifest. The end user plays the content on an iOS player.	Generic Key Provider	Not applicable.	Static, Rotating

HLS output with PlayReady

Encryption mode: Always AES CTR (AES-128)

Supported client players: Consult with the key provider (DRM implementer) for supported players.

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Conax DRM solution for protectin g HLS output using the PlayReady DRM technology. The end user plays the content on a Conax-approved player.	Conax	GetPlayReady KeyForHLS v1	Static

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Irdeto DRM. (ActiveCloak for Media) solution for protecting HLS output using the PlayReady DRM technology. The end user plays the content on an Irdeto-approved player.	Irdeto ActiveCloak for Media	Irdeto does not currently have API versioning.	Static
The customer uses the InsideSecure DRM solution for protectin g HLS output using the PlayReady DRM technology. The end user plays the content on an Inside-Secure-approved player.	InsideSecure	keyprovisioning v1.0	Static, Rotating

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the InsideSecure feature of the thePlatform DRM solution for protecting HLS output using the PlayReady DRM technology. The end user plays the content on a thePlatform-approved player.	InsideSecure on thePlatform	Not applicable; static key generated by Elemental Live	Static
The customer uses the Irdeto feature of the thePlatform DRM solution for protecting HLS output using the PlayReady DRM technology. The end user will play the content on a thePlatform-approved player.	Irdeto on thePlatform	Not applicable; static key generated by Elemental Live	Static

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Microsoft feature of the thePlatfo rm DRM solution for protecting HLS output using the PlayReady DRM technology. The end user plays the content on a thePlatform-approv ed player.	Microsoft client on thePlatform	Not applicable; static key generated by Elemental Live	Static
The customer uses the Piksel DRM solution for protectin g HLS output using the PlayReady DRM technology. The end user will play the content on a Piksel-approved player.	Piksel	GetEncryptInfo v1.0	Static

HLS output with SecureMedia

Encryption mode: Always AES CTR (AES-128)

Supported client players: Consult with the key provider (DRM implementer) for supported players.

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Arris SecureMedia DRM solution for protecting HLS output using the SecureMedia DRM technology. The end user plays the content on a SecureMedia-approved player.	SecureMedia	No versioning information is available from Arris.	Static, Rotating

HLS output with Verimatrix

Encryption mode: Always AES CTR (AES-128)

Supported client players: Consult with the key provider (DRM implementer) for supported players.

Description	DRM technology provider	Key provider (DRM implementer)	Version of server API from DRM implementer	Client player	Encryption mode	Key rotation
The customer uses the Verimatrix VCAS DRM solution for protectin	Verimatrix Content Authority System (VCAS)	Verimatrix	VCAS for Internet TV 4.2 Integration Guide	Verimatrix-approved player	AES CBC (AES-128)	Static, Rotating

Description	DRM technology provider	Key provider (DRM implementer)	Version of server API from DRM implementer	Client player	Encryption mode	Key rotation
g HLS output. This solution uses the Verimatrix-proprietary DRM technology. The end user plays the content on a Verimatrix-approved player.						

Microsoft smooth output with PlayReady

Encryption mode: Always AES CTR (AES-128)

Supported client players: Consult with the key provider (DRM implementer) for supported players.

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Conax DRM solution for protecting MSS output using	Conax	GetPlayReady KeyFor MSSmooth Streaming v1	Static

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
the PlayReady DRM technology. The end user plays the content on a Conax-approved player.			
The customer uses the InsideSecure DRM solution for protecting MSS output using the PlayReady DRM technology. The end user plays the content on an InsideSecure-approved player.	InsideSecure	keyprovisioning v1.0	Rotating
The customer uses the Irdeto feature of the thePlatform DRM solution for protecting MSS output using the PlayReady DRM technology. The end user plays the content on a thePlatform-approved player.	Irdeto on the Platform	Not applicable; static key generated by Elemental Live.	Static

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Piksel DRM solution for protecting MSS output using the PlayReady DRM technology. The end user plays the content on a Piksel-approved player.	Piksel	GetEncryptInfo v1.0	Static
Elemental Live lets you enter a key or generate a key that Elemental Live uses to encrypt the content. Elemental Live also puts that key at a customer-specified location; the client player retrieves the key from that location and decrypts the content. The end user plays the content on a Microsoft-Silverlight-approved player. Strictly speaking, an encryption solution, not a DRM solution.	Self-Generated or Static	Not applicable.	Static

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Seachange DRM solution for protecting MSS output using the PlayReady DRM technology. The end user plays the content on a Seachange-approved player.	Seachange	Acquire Packaging Data v1.0	Static

UDP/TS outputs with DVB Simulcrypt Standard

Encryption mode: Always AES CBC as described in ATIS-0800006

Supported client players: Consult with the key provider (DRM implementer) for supported players.

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
The customer uses the Verimatrix MultiCAS/DVB DRM solution for protecting UDP/TS output in compliance with the DVB Simulcrypt standard. The end user plays the content on a	Verimatrix	ECMG interface as described in ETSI TS 101 197	Rotating

Description	Key provider (DRM implementer)	Version of server API from DRM implementer	Key rotation
Verimatrix-approved player.			

Reference: add-on packages for features

Some features of Elemental Live require that you purchase an add-on package from AWS Elemental. This section includes tables that list the features and the audio and video codecs that require an add-on package.

Topics

Purchasing and licensing an add-on package

To purchase an add-on package, contact your AWS Elemental sales person.

Then to license (enable) the add-on package, follow the procedure that applies to your situation:

- You might want to deploy a newly purchased Elemental Live appliance. There is nothing for you to do because the software is already installed, along with the license (which covers the core features and all add-on packages). Your appliance is ready to use.
- You might be performing the initial deployment of Elemental Live on qualified hardware or a VM. AWS Elemental sends you an email that includes a link to the license.

  You must generate and activate the license (which covers the core features and all add-on packages) as part of the procedure to install the software. See Set up licenses (for qualified hardware) or Set up licenses (for a VM) in Elemental Live Install Guide.
- You might want to implement an add-on pack in an existing deployment, and upgrade the software at the same time.

Elemental Live provides you with a new license that you must install. See Upgrade the license in Elemental Live Upgrade Guide.

- You might want to implement an add-on pack in an existing deployment, without needing to upgrade the software.

  Elemental Live provides you with a new license that you must install. See Licensing an add-on package without upgrading in the Elemental Live Configuration Guide.

Elemental Live features that require an add-on package

Some Elemental Live features require that you purchase an add-on package. The following table lists these features in alphabetical order, and identifies the add-on package that applies to each.

Feature	Description	Add-on Package Name	Change on Web Interface
Audio outputs, normalization	Implement audio normalization and monitoring for any audio output. Audio normalization and monitoring are compliant with ITU-R BS.1770. Audio logging is compliant with the CALM ACT (ATSC A/85).	Audio Normalization Package	The **Audio Normalization** option is activated, so that you can enable it. To find the option in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. then choose **Advanced**.The field for this option appears.
Color space	Implement Dolby Vision in video outputs, from suitable inputs.	Dolby Vision Live Metadata Generation Package	The **Color Space Conversion** field on the web interface includes two options:

Feature	Description	Add-on Package Name	Change on Web Interface
			Dolby Vision Profile 5, and Dolby Vision Profile 8.1. This field is in the **Color Corrector** section of the **Preprocessors** section in the **Video** section of the **Streams** section of the **Output** section of an event. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Video**. choose **Advanced**, then choose **Preprocessors**.The field appears.
Motion graphics	Insert motion graphics into video outputs.	Motion Graphics Overlay Package	You can enable the **Motion Image Inserter** option in the event. This option is in the **Global Processors** section of an event.

Feature	Description	Add-on Package Name	Change on Web Interface
Nielsen watermark	Implement Nielsen watermark encoding in audio outputs, using NAES VI.	Nielsen Watermark Encoding Package	You can enable the **Nielsen Watermarking** option in the event. This option is in the **Advanced** section of the **Audio** section of the **Streams** section of the **Output** of an event. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. then choose **Advanced**.The field for this option appears.

Feature	Description	Add-on Package Name	Change on Web Interface
Statmux	Create statmux outputs (statistical MPTS outputs).	Statmux Rate Control Package	The **Rate Control Mode** field in the event includes the **Statmux** option. This field is in the **Advanced** section of the **Video** section of the **Streams** section of the **Output** of an event. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Video**. then choose **Advanced**.The field appears on the left, halfway down the section.

Feature	Description	Add-on Package Name	Change on Web Interface
Virtual input switching	Implement the virtual input switching feature via SCTE-35 messages, and the virtual input switching feature via asynchronous ESAM messages that the POIS sends.	Virtual Input Switcher Package	The following fields appear: • In the **Ad Avail Controls** section of an event, the **Acquisition Point Identifier** field, the **Zone Identity** field, and the **Unsolicited ESAM Server** field appear. • In the **Inputs** section, the **Virtual Input** field and **Virtual Input SCTE 35 PID** field appear.

Add-on packages for video codecs

Some video codecs require an add-on package. This table lists video codecs in alphabetical order. Each row identifies whether Elemental Live requires an add-on package in order to decode a video source that uses this codec or encode an output with this codec.

Codec	Action	Add-on Package	Change on web interface
AVC (H.264)	Decode	No add-on package required	

Codec	Action	Add-on Package	Change on web interface
	Encode specific variations of AVC (H.264). See the section called “Add-on packages for AVC (H.264) variations”.	Advanced Broadcast Contribution Encoder Package	When the **Video Codec** field is set to **MPEG-4 AVC** (H.264), then the **Profile** field for an event includes options for these variations. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Video**. The field is on the first line.
HEVC (H.265)	Decode specific variations of HEVC (H.265). See the section called “Add-on packages for HEVC (H.265) variations”.	HEVC Main Profile Encode and Decode Package	There is no change in the input section of an event, but Elemental Livewill ingest these variation s from a supported input.

Codec	Action	Add-on Package	Change on web interface
	Encode specific variations of HEVC (H.265). See the section called “Add-on packages for HEVC (H.265) variations”.	HEVC Main Profile Encode and Decode Package	When the **Video Codec** field is set to **HEVC** (H.265), then the **Profile** field for an event includes options for these variations. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Video**. The field is on the first line.
	Encode specific variations of HEVC (H.265). See the section called “Add-on packages for HEVC (H.265) variations”.	Advanced Broadcast Contribution Encoder Package	When the **Video Codec** field is set to **HEVC** (H.265), then the **Profile** field for an event includes options for these variations. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Video**. The field is on the first line.

Codec	Action	Add-on Package	Change on web interface
JPEG XS	Decode	JPEG XS Encode and Decode Package	There is no change in the input section of an event, but Elemental Live will ingest JPEG XS video sources in a supported input.
	Encode	JPEG-XS Encode and Decode Package	The **JPEG-XS** codec appears as an option in the **Video Codec** field in supported outputs of an event. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Video**. The field is on the first line.
Other video codecs	Decode	No add-on package required	
	Encode	No add-on package required	

Add-on packages for AVC (H.264) variations

The AVC (H.264) codec has *variations* based on the profile, chroma sampling, and bit depth that apply. Some variations require an add-on package to encode. The following table lists the variations, and identifies the add-on packages that Elemental Live requires to encode each variation.

Profile	Chroma Sampling	Bit Depth	Option in Profile field on Web Interface	Add-on Package to Decode	Add-on Package to Encode
Baseline	4:2:0	8-bit	**Baseline**	None	None
Main	4:2:0	8-bit	**Main**	None	None
High	4:2:0	8-bit	**High**	None	None
High	4:2:0	10-bit	**High 10-bit**	None	Advanced Broadcast Contribut ion Encoder Package
High	4:2:2	8-bit	**High 4:2:2**	None	Advanced Broadcast Contribut ion Encoder Package
High	4:2:2	10-bit	**High 4:2:2 10-bit**	None	Advanced Broadcast Contribut ion Encoder Package

Add-on packages for HEVC (H.265) variations

The HEVC (H.265) codec has *variations* based on the profile, chroma sampling, bit depth, and tier that apply. All variations require an add-on package to decode and/or to encode. The following table lists the variations, and identifies the add-on package that Elemental Live requires to decode a source that uses this codec or encode an output with this codec.

Profile	Chroma Sampling	Bit Depth	Tier	Option in Profile field on Web Interface	Add-on Package to Decode	Add-on Package to Encode
Main	4:2:0	8-bit	Main	**Main/ Main**	HEVC Main Profile Encode and Decode Package	HEVC Main Profile Encode and Decode Package
Main	4:2:0	8-bit	High	**Main/High**	HEVC Main Profile Encode and Decode Package	HEVC Main Profile Encode and Decode Package
Main	4:2:0	10-bit	Main	**Main10/ Main**	HEVC Main Profile Encode and Decode Package	HEVC Main Profile Encode and Decode Package
Main	4:2:0	10-bit	High	**Main10/Hi gh**	HEVC Main Profile Encode and Decode Package	HEVC Main Profile Encode and Decode Package
Main	4:2:2	8-bit	Main	**Main 4:2:2 8-bit/Main**	HEVC Main Profile Encode	Advanced Broadcast Contribut

Profile	Chroma Sampling	Bit Depth	Tier	Option in Profile field on Web Interface	Add-on Package to Decode	Add-on Package to Encode
					and Decode Package	ion Encoder Package
Main	4:2:2	8-bit	High	**Main 4:2:2 8-bit/High**	HEVC Main Profile Encode and Decode Package	Advanced Broadcast Contribution Encoder Package
Main	4:2:2	10-bit	Main	**Main 4:2:2 10-bit/Main**	HEVC Main Profile Encode and Decode Package	Advanced Broadcast Contribution Encoder Package
Main	4:2:2	10-bit	High	**Main 4:2:2 10-bit/High**	HEVC Main Profile Encode and Decode Package	Advanced Broadcast Contribution Encoder Package

Add-on packages for audio codecs

Some audio codecs require an add-on package. This table lists audio codecs in alphabetical order. Each row identifies whether Elemental Live requires an add-on package to decode an audio source that uses this codec, to pass through the audio source to the output, or to encode an output with this codec.

Codec	Action	Add-on Package	Change on Web Interface
Dolby Digital	Decode or passthrou gh	No add-on package required	
	Encode	Advanced Audio Package	The codec appears as an option in the **Audio Codec** field in the relevant outputs. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. The field appears on the first line.
Dolby Digital Plus	Decode	Audio Decoder Package	There is no change in the input section of an event, but Elemental Live will ingest a **Dolby Digital Plus** audio source.
	Passthrough	No add-on package required	
	Encode	Advanced Audio Package	The codec appears as an option in the **Audio Codec** field in the relevant outputs.

Codec	Action	Add-on Package	Change on Web Interface
			To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. The field appears on the first line.
Dolby E	Decode	Audio Decoder Package	The **Dolby E Program Selection** field appears for an event. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. The field appears on the first line.
	Passthrough	No add-on package required	
	Encode	No add-on package required	
Dolby Digital Plus with Atmos	Decode	Decode isn't supported	

Codec	Action	Add-on Package	Change on Web Interface
	Passthrough	No add-on package required	
	Encode	Advanced Audio Package	The codec appears as an option in the **Audio Codec** field in the relevant outputs. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. The field appears on the first line.
DTS Express	Decode	Decode isn't supported	
	Passthrough	No add-on package required	

Codec	Action	Add-on Package	Change on Web Interface
	Encode	Advanced Audio Package	The codec appears as an option in the **Audio Codec** field in the relevant outputs. To find the field in the event, go to an **Output Groups** tab. In that tab, go to the **Outputs** section, then to the **Streams** section. Choose **Audio**. The field appears on the first line.
Other audio codecs	Decode	No add-on package required	
	Passthrough	Passthrough isn't supported	
	Encode	No add-on package required	

Document history for user guide

The following table describes the main changes to the documentation for AWS Elemental Live. For notification about updates to this documentation, you can subscribe to an RSS feed.

Change	Description	Date
New way to work with SMPTE 2110 with NMOS	The guide now includes information about working with SMPTE 2110 using NMOS. The implementation of NMOS was completel y changed in 2024. The guide now describes the new implementation.	July 1, 2025
SMPTE 2110 with NMOS and AWS Elemental Conductor Live	The user guide now includes a note to clarify a constrain t with AWS Elemental Conductor Live. You can't use AWS Elemental Conductor Live to set up SMPTE 2110 inputs that use NMOS or to produce SMPTE 2110 outputs that use NMOS.	April 10, 2023
Delivering low-latency HLS output to MediaPackage	Starting with version 2.25.0, Elemental Live supports delivery of an HLS output to an AWS Elemental MediaPack age channel that uses MediaPackage v2. The HLS output delivered to MediaPackage v2 is typically (but not necessarily) part of	April 10, 2023

	a glass-to-glass low latency workflow.	
Working with SRT input	We fixed an error where we referred to the upstream system as the SRT caller. With an SRT input, the upstream system is always the sender and the SRT listener. Elemental Live is always the receiver and the SRT caller.	November 10, 2022
RTMP inputs	The guide has been revised to clarify that AWS Elemental Live supports both RTMP push and RTMP pull inputs.	September 23, 2022
Revision to output locking section	The section on implement ing output locking and epoch locking has been completely revised.	July 15, 2022
SWF files deprecated	We have removed information about using SWF with motion graphic overlays. Adobe no longer supports Flash Player. Adobe has blocked Flash content from running in Flash Player. We strongly recommend that you not implement SWF in motion graphic overlays.	July 5, 2022
Updates to video quality information	The information in this section has been updated to incorporate information about the latest improveme nts to video quality.	June 6, 2022

Performance expectations for appliances	The guide now includes information about performan ce expectations for Elemental Live appliances, including ways that you can measure and optimize it.	June 6, 2022
Update to list of supported captions	This list has been revised to remove CFF captions. The CFF fomat hasn't been supported since Elemental Live removed support for the Ultraviolet output container.	May 27, 2022
Inserting splice inserts on the day the clocks change	This section now includes information about using the REST API to insert a splice insert, specifically on the day that the clocks change to or from Standard Time.	May 27, 2022
DTS Express audio codec	Revised license requirements for this codec because decode isn't supported.	April 28, 2022
Ultra-low latency	The user guide now includes information about the existing chunked transfer feature. This feature contribut es to ultra-low latency in outputs.	February 11, 2022

Interleave 4K inputs	This section (which was formerly called *Interleave 2SI inputs*) has been revised to include the relevant L8xx appliances that can support this input.	January 10, 2022
Eligible messages in input sources	There is an update to the section on the sources and input types that are supported for SCTE-35 and SCTE-104 messages. The table is now up to date with version 2.23.2.	January 7, 2022
Dolby Atmos audio	This section has been revised to fix a mistake. The source can include up to 16 channels, not 8 channels, as previously specified.	January 7, 2022
Add-on packages	The guide now includes information about add-on packages that you can purchase.	December 15, 2021
Nielsen watermarks to ID3	Elemental Live now supports the ability to convert Nielsen watermarks to ID3 metadata.	December 6, 2021
Nielsen watermark insertion	The guide now includes information about inserting Nielsen watermarks into the output.	December 6, 2021

Virtual input switching	The guide includes informati on about virtual input switching using SCTE-35 messages (introduced in version 2.20.2) and virtual input switching through synchronous ESAM messages (introduced in version 2.23.0).	November 10, 2021
Support for RTSP inputs	Starting with version 2.22.4, Elemental Live supports RTSP inputs. The guide now includes information about the uses of this input and about the supported video and audio codecs.	October 7, 2021
Revision to codecs for supported outputs	The tables for supported input codecs have been revised to specify which input types support Dolby Digital Plus with Atmos.	October 7, 2021
Revision to codecs for supported outputs	The tables for supported input codecs have been revised to specify which input types support Dolby Digital Plus with Atmos.	October 7, 2021
Revision to codecs for supported outputs	The tables for supported output codecs have been revised to specify which output types support Dolby Digital Plus with Atmos.	October 7, 2021

Enhanced support for Dolby Digital Plus with Atmos	The guide includes information about audio output encoded with Dolby Digital Plus with Atmos. Previously, AWS Elemental Live supported this output only as passthrough. Starting with version 2.20.4, there is also support for converting an audio source to Dolby Digital Plus with Atmos.	October 7, 2021
Improved support for Dolby Vision color space	Starting with AWS Elemental Live version 2.20.3, there is a new implementation of Dolby Vision color space. The color space section of the guide has been revised to include Dolby Vision.	October 5, 2021
Color space	The section on handling color space has been completely revised.	October 5, 2021
Motion graphic using HTML5	The user guide now includes information about using an HTML5 asset in a motion graphic overlay. This feature was introduced in Elemental Live version 2.21.0. In addition, the existing information about the other methods for inserting motion graphic overlays has been redesigned, but there are no changes to the functionality.	September 23, 2021

SMPTE 2110 SDP files	The guide has been updated to clarify that the source filter lines in SDP files are optional.	September 9, 2021
Supported outputs	We have completely revised the information about types of output media and output codecs that Elemental Live supports. We have also moved the information to a section in the Outputs chapter.	August 18, 2021
Supported inputs	We have completely revised the information about input types and input codecs that Elemental Live supports. We have also moved the information to a section in the Inputs chapter.	August 18, 2021
Supported inputs	We have completely revised the information about input types and input codecs that Elemental Live supports. We have also moved the information to a section in the Inputs chapter.	August 18, 2021
Delivering a TS using the SRT protocol	Starting with version 2.22.2, Elemental Live supports delivering a TS output using the SRT protocol.	August 11, 2021
Working with SRT input	The guide now includes information about working with SRT inputs.	July 23, 2021

Revision to SMPTE 2022-6 input section	The information about setting up a SMPTE 2022-6 input has been reorganized.	July 23, 2021
Revisions to SCTE-35 section	The SCTE-35 section has been revised to fix some errors. In particular, references to RTSP and HDS outputs have been removed because these output types are no longer supported. Also, screenshots have been removed.	July 19, 2021
Trick-play in an HLS output group	Starting with version 2.22.1, Elemental Live supports inclusion of a trick-play track in an HLS output group. This track follows the Image Media Playlist specification, version 0.4. The guide has been updated to include this feature, and to include the existing feature for implementing trick-play using an I-frame only manifest.	July 7, 2021
Ingesting SMPTE 2038	The section on ingesting SMPTE 2038 has been completely revised to better describe the behavior. In addition, new information about the passing through custom data has been added.	July 7, 2021

Revision for bonded interface s in SMPTE 2110 and SMPTE 2022-6	The user guide has been revised to clarify that if you implement redundant inputs (with SMPTE 2110 or SMPTE 2022-6) or redundant outputs (with SMPTE 2110), you can't have a bonded interface on the Mellanox card or the 25Gb card.	June 28, 2021
WebVTT output captions using OCR	Starting with version 2.22.0, Elemental Live supports the ability to produce WebVTT output captions by convertin g object-based source captions that are in DVB-Sub or SCTE-27 format. Elemental Live uses OCR (optical character recogniti on) to perform this conversio n. The section on supported captions and the section on working with captions have been revised.	June 18, 2021
Color space	The section on handling color space has been completel y revised. In addition, the section includes information for working with Dolby.	June 15, 2021
SMPTE 2110	The user guide now includes sections about the existing support for SMPTE 2110 inputs and SMPTE 2110 outputs.	June 5, 2021

Version 2.25 release	First release of the 2.25 software version.	February 5, 2021