Adobe® Primetime Platform 1.2
Release Notes

Summary

Welcome to the Adobe Primetime 1.2 release! These release notes provide information about the following Primetime platform components:

- Adobe Primetime Live Packager 1.2
- Adobe Primetime HTTP Origin Server 1.2
- Adobe Primetime Offline Packager 1.2

Table of Contents

Adobe Primetime Live Packager

Welcome to the Adobe Primetime - Live Packager 1.2 release!

Primetime Live Packager is a scalable server technology that can ingest multiple pre-encoded H264 feeds through M2TS, multicast and output can be AES-encrypted or DRM-protected streams with ad signals to HDS and HLS formats.
Primetime Live packager also supports ingestion of RTMP streams from encoders. In addition to TS streams over multicast, it can now act as an RTMP server, to which RTMP streams can be published.
The Primetime Live Packager provides a packaging and ad signaling solution independent of the encoder to develop services that use Adobe Primetime.

Detailed configuration and feature details of Primetime Live Packager can be found at
http://help.adobe.com/en_US/primetime/#Platform-concept-Primetime_Live_Packager_Getting_Started_Guide

Minimum system requirements

Network requirements

- ✓ Multicast between Encoder and Live Packager
- ✓ Encoder with RTMP output

Supported operating systems

- ✓ Linux CentOS 5.8 64 bit

Hardware requirements

- ✓ 3.2GHz Intel® Pentium® 4 processor (dual Intel Xeon® or faster recommended)
- ✓ 64-bit operating systems: 4GB of RAM (8GB recommended)
- ✓ 1Gb Ethernet card recommended (multiple network cards and 10Gb also supported)
- ✓ Disk
  - (Disk-SAS) : Minimum 10GB with 7.5K RPM
  - (Disk-SSD) : 400MBps read/write
  - (NAS) : 1 GB dedicated link

Software requirements

- Oracle Java JRE 1.6 (Recommend: Sun/Oracle Hotspot JVM)
  The JDK is required for JConsole access to the JMX APIs

Installing the Live Packager

- Download the Java SE and JDK software from the Oracle site and follow the installation instructions.
- Extract the Adobe Primetime - Live Packager 1.2 archive file to your disk.

To start the Live Packager

To start the packager, execute the following command from the packager's root directory:

$*packager_start.sh*

To stop the Live Packager

To stop the packager, execute the following command from the packager's root directory:

$*packager_stop.sh*

To restart the Live Packager

To restart the packager, stop and start the packager.

Note: When the packager starts, it attempts to initialize the manifest information from the fragment target in the temp directory. If the manifest information is found at the fragment target, it implies that the packager has been restarted. In case of restart, the packager waits until the next fragment boundary of incoming stream and then starts packaging. The packager inserts a gap entry in the bootstrap to indicate that there are missing fragments.

Configuring the Live Packager

After extracting the Adobe Primetime - Live Packager 1.2 archive, change directories to the packager directory, and run the packager_start.sh script. The sample configuration listens on the multicast address 239.235.0.3:14000, and runs the local origin server on port 8080. The output is configured to be written to the packager/webroot/_default_/_default_/ directory.

Refer the Primetime Live Packager Getting Started document for the configuration details.

Uninstalling the Live Packager

To uninstall the packager, stop the packager and remove the packager directory from the Primetime Platform directory.

New features:

Enhanced SCTE 35 Cues support

Primetime Live Packager has the capability to parse in-stream SCTE 35 cues and then add them as cue info in the generated manifest. The Primetime Live Packager captures SCTE 35 cues from the broadcast feed and passes them along so that they may be used for ad decisioning / insertion / blackouts by downstream processors. These cues are added in f4m for HDS output and m3u8 for HLS output.

In PT Packagers 1.2, parsing of SCTE 35 cues has been enhanced to support a wider part of SCTE 35 spec. The cue formats specified in Digital program insertion specifications are supported. For more details about DPI cue formats please refer to Primetime Live Packager getting started guide.

Query Parameters in http push

Target URL for HTTP Push in packager's configuration now supports query parameters. For example, the http push target url can look like

```
<HttpPush>
 <UseSecurityToken>true</UseSecurityToken>
       <SecurityTokenKey>4ff4756ed68239d34d482dbc88819abc</SecurityTokenKey>
      <TargetURL>http://host:port/module_path?p1=val1&amp;p2=val2</TargetURL>
</HttpPush>
```

Some example fragment/manifest URIs for this http push target:

http://host:port/module_path/livestream.f4m?p1=val1&p2=val2
http://host:port/module_path/livestreamSeg1-Frag9?p1=val1&p2=val2
http://host:port/module_path/livestream.2347823.ts?p1=val1&p2=val2
http://host:port/module_path/livestream.m3u8?p1=val1&p2=val2

HLS Manifest Custom Tag for Synchronization

HLS Manifest (M3U8) will contain a custom tag **#EXT-X-MEDIA-TIME**. It will be set to the stream time at start of the following TS segment. This time will be same as the Presentation Time of the first sample in the segment in fractional seconds.

#EXT-X-MEDIA-TIME will occur in M3U8 at:

- First segment in the M3U8 sliding window
- Each segment following the #EXT-X-DISCONTINUITY tag

This tag fills the gap in HLS spec that it is not possible to derive timestamps just from M3U8 and without parsing the TS segments. It can be used to synchronize across different renditions during bitrate switch-over or failover. Synchronization is needed for example to insert pre-roll ads in a live stream having multiple renditions.

Security Token for Output Targets:

Now Security Token will not be enabled by default for Http or Manifest Push. Token will be added only if the *UseSecurityToken* tag is set as true. If *UseSecurityToken* is set as true then a valid AES 128 Key must be provided in the *SecurityTokenKey* tag. No default key will be used for token generation. If *UseSecurityToken* is set as true and *SecurityTokenKey* is not provided then *ModuleConfigException* will be thrown.

DRM/FAXS Impact

PT Packager 1.2 will no longer create an expired leaf license for chained license policy. All the leaf license properties like validity, start date and end date, will be determined from the policy. To create an expired leaf license to enforce the license validity from root license, the end data should be specified as such in the policy.

Configuration Changes:

Following configurations have been added / modified to stream.xml

Configuration	Details
Stream/OutputPipeline/ID	A unique ID for a given Output Pipeline. It can be any alphanumeric String. For backward compatibility, it is not a mandatory config parameter. However if more than one Output Pipeline of same Output Type is configured then ID is mandatory and each Output Pipeline needs to define different ID.
Stream/OutputPipeline/CueMode	The Cue Info Mode used for cue information to be added to m3u8/f4m. Allowed values are PT_1_0, DPI-Simple and DPI-Scte35. It's not a mandatory configuration. By default value "DPIScte35" is used.
Stream/OutputPipeline/Output/HttpPush or ManifestPush/UseSecurityToken	When active, adds an encrypted security token to the outgoing HTTP request. By default it value is false.
Stream/OutputPipeline/Output/HttpPush or ManifestPush/SecurityTokenKey	The AES-128 bit key to be used for encrypting the PUT token. This value shall be a string of 32 hexadecimal digits. If UseSecurityToken is set as true then this param must be provided.
~~Stream/OutputPipeline/Output/HttpPush/SerializePush~~	It is *not supported in 1.2* and exists only in Packager version 1.1.

Stream/OutputPipeline/Output/HttpPush /MaxConcurrentPushCount	Maximum number of fragments that can be dispatched in parallel. The default value is 1, implying that the fragments will be pushed by the packager to remote origin serially by default. For pushing large fragments of high bitrate over high latency network concurrently, it can be set to a value more than 1.

stream.xml example:

```
<Stream>
  <FragmentDuration>2000</FragmentDuration>
  <DiskManagementDuration>3.0</DiskManagementDuration>
<Input>
    <TSMulticast>
    <MulticastAddress>239.235.0.3</MulticastAddress>
    <MulticastPort>14000</MulticastPort>
  </TSMulticast>
  </Input>
  <OutputPipeline>
    <OutputType>HDS</OutputType>
        <ID>HDS_AV</ID>
        <CueMode>PT_1_0</CueMode>
    <Output>
      <LocalOrigin>
        <ContentPath>webroot/_default_/_default_</ContentPath>
      </LocalOrigin>
    </Output>
  </OutputPipeline>
   <OutputPipeline>
    <OutputType>HDS</OutputType>
        <ID>HDS_Audio</ID>
        <IsAudioOnly>true</IsAudioOnly>
    <Output>
      <LocalOrigin>
        <ContentPath>webroot/_default_/audio</ContentPath>
      </LocalOrigin>
              <HttpPush>
              <UseSecurityToken>true</UseSecurityToken>
                <SecurityTokenKey>4ff4756ed68239d34d482dbc88819abc</SecurityTokenKey>
              <TargetURL>http://localhost:8090/_default_/_default_</TargetURL>
          </HttpPush>
    </Output>
  </OutputPipeline>
</Stream>
```

JMX Operations:

- ✓ Reset() API of ModuleLifecycleManagerMBean is now applicable for Stream Container Modules as well . Invoking reset() API on a Stream Container Module resets all the child modules (streams) under that container.

- ✓ Refresh() API: When a container is refreshed through refresh() API call, and if some stream module fails to load, the response status code 8 would be returned along with stack trace which details the reason of failure.

Adobe Primetime HTTP Origin Server

Welcome to the Adobe Primetime - HTTP Origin Server 1.2 release!

Primetime HTTP Origin Server is a pre-configured HTTP server that connects to a local storage and cache for live, linear, or VOD fragments. The Origin server only supports segmentless fragments, or fragments not bound inside the F4F segment file. This HTTP server has full support for HLS (Apple M3U8) file format index lookup.

Detailed configuration and feature details of Primetime Http Origin Server can be found at
http://help.adobe.com/en_US/primetime/#Platform-concept-Origin_Server_Getting_Started_Guide

Minimum system requirements

The Adobe Primetime – HTTP Origin Server 1.2 system requirements are:

Supported operating systems

- ✓ Linux CentOS 5.8 64 bit

Hardware requirements

- ✓ 3.2GHz Intel® Pentium® 4 processor (dual Intel Xeon® or faster recommended)
- ✓ 64-bit operating systems: 4GB of RAM (8GB recommended)
- ✓ 1Gb Ethernet card recommended (multiple network cards and 10Gb also supported)
- ✓ Disk
  - (Disk-SAS) : Minimum 10GB with 7.5K RPM
  - (Disk-SSD) : 400MBps read/write
  - (NAS) : 1 GB dedicated link

Software requirements

- ✓ Oracle Java JRE 1.6 (Recommend: Sun/Oracle Hotspot JVM)
  The JDK is required for JConsole access to the JMX APIs.

Installing the HTTP Origin Server

- Download the Java JRE and JDK software from the Oracle site and follow the installation instructions.
- Extract the Adobe Primetime - HTTP Origin Server 1.2 archive file to your disk.

To start the HTTP Origin Server

To start the HTTP Origin Server, execute the following command from the command line in the Origin Server's root directory:

$./origin_start.sh

To stop the HTTP Origin Server

To stop the HTTP Origin Server, execute the following command in the Origin Server's root directory:

$./origin_stop.sh

To restart the HTTP Origin Server

To restart the HTTP Origin Server, stop and start the HTTP Origin Server.

Configuring the HTTP Origin Server

Refer the Primetime HTTP Origin Server Getting Started document for the configuration details.

Uninstalling the HTTP Origin Server

To uninstall the HTTP Origin Server, stop the HTTP Origin Server and remove the HTTP Origin Server's httporigin directory in the Primetime directory.

New Features:

MP4 JIT Packaging

Origin Server 1.2 supports generating HDS or HLS VOD stream just-in-time from an mp4 file which has been copied to Origin Server's disk or is located on the network. For a VOD module on Origin Server, following MP4 JIT transformations will be available:

MP4 file location	MP4 file URI
local disk	*http://origin-server/container_name/module_name/mp4_relative_path/filename.mp4* *OR* *http://origin-server/container_name/module_name/mp4_relative_path/filename.m3u8*
on network	*/mp4_relative_path/filename.mp4.f4m or /mp4_relative_path/filename.mp4.m3u8*

Access Plugin for remote mp4

For mp4 files which are fetched from a remote http server, an http access plugin will have to be provided which will allow modification of the remote mp4 http request so that headers can be added/removed and url can be re-written. The specifics of the plugin API are covered in the Exposed API section. PT Origin 1.2 comes with two default implementations of http access plugin.

One is com.adobe.fms.util.httpfileaccess.SimpleHttpAccess which just resolves the mp4 url against the base URL of remote http server on which mp4 assets are hosted.
The other one is com.adobe.fms.util.httpfileaccess.S3HttpAccess which resolves the mp4 request against the configured S3 bucket and also computes the request token for accessing the mp4 and adds the required headers to the mp4 http request.

The name of the plugin to be used and the corresponding configuration needs to be specified in the vod.xml under the "Config/MP4" element. Fully qualified class name of the plugin should be specified in the element Config/MP4/HttpAccessPlugin/Class and any configuration for the plugin should be specified under Config/MP4/HttpAccessPlugin/Init.

Example config for SimpleHttpAccess

```
<MP4>
    <!-- The duration of the output fragments in milliseconds. -->
    <FragmentDuration>4000</FragmentDuration>
    <!-- The TargetDuration should be greater than the maximum fragment duration in milliseconds. -->
    <TargetDuration>6000</TargetDuration>

    <HttpAccessPlugin>
      <Class>com.adobe.fms.util.httpfileaccess.SimpleHttpAccess</Class>
      <Init>
        <BaseURL>http://server_host:port/root_path</BaseURL>
      </Init>
    </HttpAccessPlugin>
</MP4>
```

WebVTT Support

The Primetime Origin shall support JIT re-packaging of WebVTT files for HLS delivery. Settings for Fragment Duration, Target Duration, Disk path/Network Path shall be derived from the MP4 JIT transformation settings. The URI for JIT re-packaged WebVTT file shall be of the form *http://origin-server/container_name/module_name/mp4_relative_path/filename.vtt.m3u8*. The following is a sample master (m3u8) playlist

Sample Master Playlist

```
#EXTM3U
#EXT-X-MEDIA:TYPE=SUBTITLES,GROUP-ID="subs",NAME="English",DEFAULT=YES,URI="http://origin-server:8090/_default_/_vod_/orig/english_en.vtt.m3u8",LANGUAGE="en"

#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=150000,SUBTITLES="subs"
http://origin-server:8090/_default_/_vod_/movie.mp4.m3u8
```

JIT HLS Fault Tolerance Support

Primetime Http Origin Server supports JIT HLS fault tolerance using server-side redundancy and healing in addition to HLS client-side failover. Fault Tolerance can be enabled under Config/FaultTolerance in stream.xml. 'ConstantLagFromLiveEdge' is the new config added in Origin Server version 1.2.
For more details about the fault tolerance supports and configuration details please refer to Primetime Http Origin Server getting started guide.

HLS Manifest Tags for Synchronization

#EXT-X-MEDIA-TIME:

Just-in-time generated M3U8 at origin will contain a custom tag, called #EXT-X-MEDIA-TIME, for synchronization across renditions

#EXT-X-PROGRAM-DATE-TIME:

- SMPTE timecodes in the HDS manifest will be used to generate #EXT-X-PROGRAM-DATE-TIME tags in JIT created M3U8, in the following way: Value of attribute **@smpte** of <smpteTimecode> element in HDS manifest, will be used to derive the TIME part.
- If there are multiple <smpteTimecode> elements whose @timestamp lies in a single fragment, then only the first time-code will be utilized.
- If <smpteTimecode> is not already aligned with the fragment-start-timestamp, then the value of smpte attribute will be adjusted so that the resulting time is aligned with start of the fragment on which #EXT-X-PROGRAM-DATE-TIME tag is applied.
- If @timezone is not specified in <smpteTimecode>, then GMT is assumed in #EXT-X-PROGRAM-DATE-TIME
- If date is not provided, then the current system/wall-clock date is taken.

JIT Audio Only

PT Origin 1.2 will support JIT audio only conversion for live and vod streams for both HDS and HLS. For any published live/vod stream, the corresponding audio only stream will be available at

Original URL	JIT Audio Only URL
http://host:port/container/module/rel_path/streamName.f4m	http://host:port/container/module/rel_path/audio_only/streamName.f4m
http://host:port/container/module/rel_path/streamName.m3u8	http://host:port/container/module/rel_path/audio_only/streamName.m3u8

For HLS, this feature is available for both the HLS streams directly published to Origin and also those which are JIT generated at Origin.

HLS JIT Key Frame Only

For HLS, trick play assets can be JIT created on PT Origin 1.2 while accessed using a URL containing kfonly. For any published live/vod stream, the corresponding trick play stream will be available at:

Original URL	JIT Trick Play URL
http://host:port/container/module/rel_path/streamName.m3u8	*http://host:port/container/module/rel_path/kfonly/streamName.m3u8*

Section 3.4.12. of HLS Spec explains the requirements for the trick play m3u8. The trick play m3u8 is generated as per this specification.
HLS Trick Play requires minimum HLS version 4. All the JIT Trick Play HLS Streams will contain EXT-X-VERSION as 4 or higher (even if the corresponding original stream is at a lower version).

So for the above example, it is possible that,

http://host:port/container/module/rel_path/streamName.m3u8

has #EXT-X-VERSION:3 whereas

http://host:port/container/module/rel_path/kfonly/streamName.m3u8

will have #EXT-X-VERSION:4

For HLS, this feature is available for both the HLS streams directly published to Origin and also those which are JIT generated at Origin.
Please note that for this feature to work, the segment URIs inside the m3u8 should be relative URIs. If m3u8 contains absolute segment URI then JIT Trick Play will not work.

Serving PUT requests of HDS fragments

Origin server will respond to PUT request of HDS fragment from packager or encoder with 409 (CONFLICT) under the following situations:

- HDS manifest, containing END_OF_PRESENTATION discontinuity, has been pushed earlier marking end of the stream.
- The fragment, being pushed to the origin, contains backward timestamp and it is older than the start of the current bootstrap sliding window.

Enhanced SCTE 35 Cues support

PT Origin 1.2 supports parsing of cues generated by PT Packager 1.2. The cue formats specified in DPI spec are supported.
The Primetime Origin server processes SCTE 35 cues from the source HDS Manifest (F4M) and adds them to Just-in-time generated HLS manifest (M3U8). The format of cue info to be added in the m3u8 is determined using the option *Config/*CueMode, available for both Live and Vod origin modules.

HTTP Listener for Containers

Stream Containers on Origin will now support HTTP GET / PUT / POST /DELETE for set level f4m and m3u8. HTTP requests for files of any other type will return in 415 error(unsupported media type).

Example URI:
To publish a set level manifest named eventXYZ.m3u8 to stream container "*default*", the URI will be http://host:port/_default_/eventXYZ.m3u8

Modifications to Container Config

Security token and TTL settings can now be specified in container.xml which will be applicable for set level manifests published to container. Separate security tokens can be specified for PUT/POST and DELETE requests. It is possible to enable security token for PUT/POST and DELETE requests independent of each other i.e. say security token can be enabled only for DELETE and disabled for PUT/POST or vice versa.

These settings on container are NOT inherited by stream/vod module. To apply security or TTL settings on a stream/vod module, corresponding config must be applied in stream.xml/vod.xml

HTTP Put/Delete Listener for VOD Modules

VOD Modules on Origin 1.2 will now accept HTTP PUT /POST /DELETE requests for VOD assets. Separate security tokens can be specified for PUT/POST and DELETE requests. It is possible to enable security token for PUT/POST and DELETE requests independent of each other i.e. say security token can be enabled only for DELETE and disabled for PUT/POST or vice versa or enabled for both or disabled for both.

Set Level Manifest publishing to static

Origin 1.2 has a default http endpoint at /static. This can be used for hosting set level manifests. The URI for static location is : http://host:port/static

The static end point only supports HTTP GET/PUT/POST/DELETE for set level manifests. Requests for any other media type results in error 415(unsupported media type).

Config for /static endpoint:

Security token can be specified in origin.xml which will be applicable for HTTP PUT/POST/DELETE requests for set level manifests at "/static" endpoint. This security token setting in origin.xml is NOT inherited by modules. To apply security or TTL settings on modules, corresponding config must be applied in stream.xml/vod.xml

Security Token

Now Security Token will not be enabled by default for modules. Token will be expected only if the Auth/SecurityToken tag is present in the applicable module/origin config file. If Auth/SecurityToken is present then a valid AES 128 Key must be provided in the Key tag. No default key will be used for token generation. If Auth/SecurityToken tag is present and Key is not provided then ModuleConfigException will be thrown.

Delete Security Token can also be specified for Stream Modules in stream.xml for http delete requests.

DRM/FAXS Impact

PT Origin 1.2 will no longer create an expired leaf license for chained license policy.

All the leaf license properties like validity, start date and end date, will be determined from the policy. To create an expired leaf license to enforce the license validity from root license, the end data should be specified as such in the policy.

HSM Module Access in Origin

Primetime Origin 1.2 has support to access Packager Credential certificate (required for JIT encryption) and common key from an HSM (Hardware security module).

Configuration Changes:

Following configs have been added/modified in vod.xml

Name	Description
Config/MP4/TargetDuration	*HLS Target duration in milliseconds to be used for MP4 JIT HLS Packaging. It's an optional parameter. In case it's not specified, target duration is set as the duration of the longest fragment. For details about target duration, refer HLS Spec at http://tools.ietf.org/html/draft-pantos-http-live-streaming-10. If the longest fragment created is longer than the target duration specified by user, then the longest fragment duration is used as the target duration.*
Config/MP4/HTTP/HttpAccessPlugin/Init	*XML Config for the Http Access Plugin must be specified under this element.*
Config/MP4/HTTP/HttpAccessPlugin/Class	*Fully qualified class name of the Http Access Plugin to access remote mp4 files*
Config/MP4/FragmentDuration	*Fragment duration in milliseconds to be used for MP4 JIT Packaging(both HDS and HLS). If not specified, default value of 4000 ms is used.*
Config/CueMode	*The format of cue info to be added in the JIT generated f4m/m3u8. The allowed values for this config option are:PT_1_0, DPISimple, DPIScte35. The default value is DPIScte35*
Config/Auth/SecurityToken/Timeout	*Optional 'Timeout' for Token values in seconds. Default value is 600 seconds (10 minutes).*
Config/Auth/SecurityToken/Key	*The AES-128 bit key to be used for decrypting the PUT/POST token. This value shall be a string of 32 hexadecimal digits.*
Config/Auth/DeleteSecurityToken/Timeout	*Optional 'Timeout' for DELETE Token values in seconds. Default value is 600 seconds (10 minutes).*
Config/Auth/DeleteSecurityToken/Key	*The AES-128 bit key to be used for decrypting the DELETE token. This value shall be a string of 32 hexadecimal digits.*

Following configs have been added/modified in stream.xml:

Name	Description
Config/Auth/DeleteSecurityToken/Key	*The AES-128 bit key to be used for decrypting the DELETE token. This value shall be a string of 32 hexadecimal digits.*
Config/Auth/DeleteSecurityToken/Timeout	*Optional 'Timeout' for DELETE Token values in seconds. Default value is 600 seconds (10 minutes).*
Config/CueMode	*The format of cue info to be added in the JIT generated m3u8. The allowed values for this config option are:PT_1_0, DPISimple, DPIScte35. The default value is DPIScte35*
Config/FaultTolerance/ConstantLagFromLiveEdge	*Time (in seconds) by which JIT created M3U8 will always lag from the live edge of source HDS. If there is gap, then there can be additional lag at most the value of MaxLagFromLiveEdge.*

JMX Operations

Now JMX Stats MBean is also exposed for Stream Container Module, in addition to the Stream Module and VOD Module.

JMX Stats MBean expose following attributes:

Name	Type	Description	Example
get_requests	*Long*	*The number of GET requests to the module*	*756*
put_requests	*Long*	*The number of PUT/POST requests to the module*	*231*
delete_requests	*Long*	*The number of DELETE requests to the module*	*7*
get_errors	*String*	*It's a string representation of the number of error response(4xx/5xx) for HTTP GET requests on the given module. The format for the String is:* *errorCode1 : counter1 ; errorCode2 : counter 2 ;*	*500 : 10 ; 503 : 178 ;*
put_errors	*String*	*It's a string representation of the number of error response(4xx/5xx) for HTTP PUT/POST requests on the given module. The format for the String is:* *errorCode1 : counter1 ; errorCode2 : counter 2 ;*	*409 : 72 ; 500 : 283 ;*
delete_errors	*String*	*It's a string representation of the number of error response(4xx/5xx) for HTTP DELETE requests on the given module. The format for the String is:* *errorCode1 : counter1 ; errorCode2 : counter 2 ;*	*409 : 72 ; 500 : 283 ;*

Also, invoking the reset() API on Stream/VOD/Container module will now reset these JMX counters as well.

Lifecycle MBean for Stream Container

Following APIs/attributes have been added to Lifecycle MBean of the Stream Container on the Origin:

- Attribute: Config
  This attribute represents the current XML config for the stream container module.
- API: setConfig(String xmlConfig)
  Updates the xml config for the module with the given xml config. But this API does not reload the module. Config changes are only reflected upon module reset using ModuleLifecycleManagerMBean.reset() method.

Logging

A new formatter has been added to PT Origin 1.2 to format access logs in a manner which can be easily indexed by Splunk. The formatter class is com.adobe.fms.http.KeyValueAccessFromatter.

Exposed APIs

com.adobe.fms.util.httpfileaccess.HttpAccessPlugin interface is provided to allow users to create custom plugins for accessing their mp4 vod assets located on remote http server.

Two default plugins provided by PT Origin 1.2:

- **com.adobe.fms.util.httpfileaccess.SimpleHttpAccess**
  It provides simple access to an http server without any authentication.
  Configuration Specification for the plugin

Config	Description
//Init/BaseURL	*Base URL of the Http Server*

- **com.adobe.fms.util.httpfileaccess.S3HttpAccess**
  It provides access to Amazon's S3 storage. Note that to use S3 as storage for mp4 files, the URI for JIT-HDS or JIT-HLS stream for such mp4s will be determined as

http://origin-host:port/container/vod_module/key_of_mp4_object_ons3_bucket.f4m or http://origin-host:port/container/vod_module/key_of_mp4_object_ons3_bucket.f4m

  For example, if the configured bucket is named "BucketXYZ" and the mp4 is stored in this bucket under the key **mymp4file.mp4**, then JIT streams for this mp4 will be hosted at following URIs at the Origin Server:

HDS	*http://origin-host:port/container/vod_module/mymp4file.mp4.f4m*
HLS	*http://origin-host:port/container/vod_module/mymp4file.mp4.m3u8*

Adobe Primetime Offline Packager

Welcome to the Adobe Primetime - Offline Packager 1.2 release!

Primetime Offline Packager is a standalone tool that converts MP4 and TS content into protected formats (HLS, HDS) for Primetime media player along with the ad markups derived from in-stream SCTE35 (in case of TS) and/or an external text file. The Offline Packager helps customers prepare content after the encoding stage to be streamed over HTTP.

Detailed configuration and feature details of Primetime Offline Packager can be found at
http://help.adobe.com/en_US/primetime/#Platform-concept-Origin_Server_Getting_Started_Guide

Minimum requirements

Software requirements

- ✓ Oracle Java SE 1.6 or higher

Installing Primetime - Offline Packager

- ✓ Download the Java SE software from the Oracle site and follow the install instructions.
- ✓ Extract the Adobe Primetime – Offline Packager 1.2 archive file to your disk.

Packaging with Primetime - Offline Packager

The Offline Packager takes input through the command line or from a configuration file available on disk. Every file path that is specified as a configuration parameter must be a path on the disk. The Offline Packager does not support http pull. Similarly for output, the Offline Packager supports only disk write. Push to some http locations and CMSs are not supported.

All the options for the Offline Packager can be set in the command line itself or set in a configuration file. The three required options are:

- in_path
- out_type
- out_path

Example:
$java -jar OfflinePackager.jar -in_path "/source/filename.mp4" -out_type hds -out_path "/target//"

Optionally, a configuration in a file can be used with the conf_path parameter. The parameter names to be used in the config file are the same as the command line parameter names.

Example:
$java -jar OfflinePackager.jar -conf_path /data/config1.xml

The sample configuration file, config1.xml, will look like the one below:

```
<config>
        <in_path>/source/filename.mp4</in_path>
        <out_type>hds</out_type>
        <out_path>/target/</out_path>
</config>
```

New features:

SCTE 35 Cues

The Primetime Offline Packager has the ability to parse SCTE 35 cues from input ts file and then add them as cue info to f4m for HDS output and m3u8 for HLS output.

SCTE 35 Cue Parsing

Primetime OfflinePackager has the capability to parse in-stream SCTE 35 cues and then add them as cue info in the generated manifest. The Primetime OfflinePackager captures SCTE 35 cues from input ts/mp4 file and passes them along so that they may be used for ad decisioning / insertion / blackouts by downstream processors. These cues are added in f4m for HDS output and m3u8 for HLS output.

In Primetime Offline Packagers 1.2, parsing of SCTE 35 cues has been enhanced to support a wider part of SCTE 35 spec. Packagers 1.2 generate cues in the formats described in Digital program insertion (DPI) Specifications.

Support for creating WebVTT HLS streams

PT Offline packager can be used for creating WebVTT HLS streams. A WebVTT HLS stream comprises of a WebVTT m3u8 and the corresponding WebVTT segments created as per the HLS specification. This feature will allow a WebVTT stream to be created from a valid WebVTT file (extenstion .vtt) specified in the in_path parameter. This functionality shall be available only when the out_type is HLS. Ad-Cues are not supported by this feature. DRM/Encryption workflows are not supported for WebVTT streams.

The following is an example of how an WebVTT stream can be created:
java -jar OfflinePackager.jar -in_path ~/vtt/orig/english_en_full.vtt -out_path ~/webroot/vtt/orig -out_type hls

The following is an example of how a VTT stream can be included in the HLS master playlist.

Sample Master Playlist

```
#EXTM3U
#EXT-X-MEDIA:TYPE=SUBTITLES, GROUP-ID="subs",NAME="English",DEFAULT=YES,URI="/vtt/orig/english_en_full.m3u8",LANGUAGE="en"

#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=150000,SUBTITLES="subs"
/vtt/video/movie.m3u8
```

DRM/FAXS Impact

PT Offline Packager 1.2 will no longer create an expired leaf license for chained license policy. All the leaf license properties like validity, start date and end date, will be determined from the policy. To create an expired leaf license to enforce the license validity from root license, the end data should be specified as such in the policy.

Configuration

Following configuration parameters have been added / modified.

Parameter Name	Description	Data Type	Type

cue_mode	*Cue info signaling mode to be used for manifest. Allowed values are PT_1_0/DPISimple/DPIScte35*	*String*	*Optional. If not specified, by default DPIScte35 is assumed.*