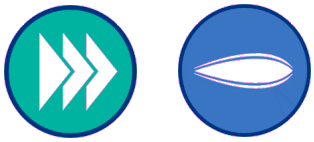

Packet Ship *Streamline* Media Server

Release Note

3.1 "Antigua" Update 4

Package	Version	Revision
ps-streamd	**3.1.4**	**1**
ps-index-mpeg2ts	**3.1.4**	**1**
ps-analyse-mpeg2ts	3.1.2	1

Release Note

The 3.1 “Antigua” Update 4 release of the Packet Ship Streamline media server contains a number of improvements to the HLS controller and a change to the treatment of near-live streams from the Timeline IPTV Recorder (ps-captured). There are also a number of further improvements to the indexer to cope with still more unusual H.264 encodings, and the ability to accept input from a pipe.

- Optimise HLS streaming where client opens a new connection for every chunk
- Tighten HLS playlist syntax to fix interoperation with Amino and Oregan clients
- Simplify HLS playlist URL format using cookies for session state
- Remove Content-Length header from HTTP chunked encoding
- Make HLS X-EXT-KEY tag generation dependent on licence parameter
- Improvement to live stream generation in HLS
- Change interface to ps-captured for live streams, and new configuration in ps-streamd
- New RTP extension carrying current stream NPT
- H.264 encoding fixes in ps-index-mpeg2ts
- Allow input from standard input (pipe) in ps-index-mpeg2ts

HLS optimisations

The HLS server has been further optimised to cope with the case where the client (inefficiently) opens a new connection for every chunk. Previously this would trigger a new stream start; the server is now able to switch the stream to the new connection.

HLS Playlist syntax tightening & simplification

Two minor issues were found with the way HLS playlists were generated:

1. Not including a comma and description in #EXTINF (broke Amino)
2. Including a space after colon in X-EXT-MEDIA-SEQUENCE (broke Oregan in live streams)

The server now also uses cookies to keep session state, making the playlist URLs shorter.

Removing Content-Length header in chunked encoding

The server no longer issues Content-Length headers when using chunked encoding (which is banned by the HTTP/1.1 RFC), and was initially done to work round a problem with encrypted HLS content in IOS (e.g. iPad). This fixes interoperation with Amino and makes the server more standard.

Encrypted content in IOS

However we have not been able to determine why IOS does not like encrypted files with chunked encoding (it works fine in MacOS and other devices), which means that the IOS client will only play the first chunk of encrypted streams.

We suspect a bug in the IOS HLS client which is only visible when reading from dynamically generated content. However since the IOS client does not provide a client certificate the encryption scheme is insecure in any case and we would not recommend using it.

HLS X-EXT-KEY tag dependent on 'licence' parameter

In the previous version each HLS controller either provided an X-EXT-KEY tag in the playlist or not; hence supporting both unencrypted and encrypted content required two controllers on different ports. Generation of the X-EXT-KEY tag is now triggered by the presence of a 'licence' parameter in the stream URL, which is required for secure use of the Packet Ship Keyline DRM server in any case. Hence both encrypted and unencrypted content can now be served from the same port (e.g. port 80).

If the old scenario is required without a licence parameter, a **`force`** attribute can be set on the **`<encryption>`** element in the **`<hls>`** controller. If this is set, all playlists are generated with an X-EXT-KEY tag, and if unencrypted content is required as well, two controllers will be required.

```
<encryption type="AES-128" force="yes">
   <url template="https://mykeyserver/get-key?asset=$ASSET"/>
</encryption>
```

Live stream generation in HLS

Generation of 'live' playlists in HLS – that is to say, streams from the **`<capture>`** Directory Provider without a defined end time – has been greatly improved. The resulting playlists are much shorter and now make use of session state to allow the URLs to remain consistent as the underlying recording files are rotated, as required by the HLS specification.

Capture provider interface

The interface to the Timeline ps-captured (defined by the **`<capture>`** Directory Provider) has changed to support the improvements in live streams, and `ps-captured` should be upgraded to version 1.2.2.

Another effect of the change is to move the configuration of the 'rewind' time for live streams from `ps-captured` to `ps-streamd`. The **`current_start_rewind`** and **`currrent_end_rewind`** attributes from `captured.cfg.xml` have been replaced by new parameters (with slightly different meanings) the **`<capture>`** element of `streamd.cfg.xml`:

```
<capture>
  <asset prefix = "tv/"/>
  <live delay="30" limit="15"/>
</capture>
```


The **delay** attribute of the **<live>** element gives the number of seconds to rewind from real time for the start of the playlist. The **limit** attribute gives the maximum length of the playlist. Hence the playlist ends at (**delay**-**limit**) seconds before real time, which is equivalent to the old **current_end_rewind** in `captured.cfg.xml`.

The new version of the server is also much better at reading captures right up to real time, although disk access can become very inefficient if it is less than a disk buffer size behind the write point. If very near real-time streaming is required it may be beneficial to disable **<direct read>** in the **<file>** input, so that the streaming server can make use of the disk cache which will have been recently written by the recorder.

RTP NPT extension

An optional RTP extension has been added which carries the current NPT of the stream, both in normal play and trickplay (in which case it changes at a multiple of real time). This can be used by some clients to better inform the current position for progress gauges.

If enabled the extension contains a 4 byte network byte order integer containing the NPT represented as a 90Khz clock – the same format as the RTP-AVP timestamp. The extension ID can be configured with the **id** attribute of an **<extension>** element inside the **<rtp>** filter definition of the relevant output:

```
<filters>
  ...
  <rtp>
    <payload type="33"/>
    <extension id="42"/>
  </rtp>
</filters>
```

If this is left out or set to zero (the default) then the extension is disabled).

H.264 fixes in ps-index-mpeg2ts

Yet more abstruse encoding cases in H.264 which confused the indexer have come to light and these have now been fixed in the new version.

Input from pipe for ps-index-mpeg2ts

To allow the indexer to be run as part of a pipeline (for example, 'tee'd off from the output of a transcoder), it will now allow '-' as an input filename, in which case it reads from its standard input.

```
$ ffmpeg -i anyoldfile.avi {codec params} - | tee newfile.ts
  | ps-index-mpeg2ts – newfile.ts.psi2
```

Because of the better support for near-to-live streaming in this version it is now safe to start playing the newly encoded file (and associated index) while the transcode is still happening.

