



Packet Ship *Timeline* IPTV Recorder

Release Note 1.2 “Hooke” Major Release

Package	Version	Revision
ps-captured	1.2.1	1



Release Note

The 1.2 “Hooke”¹ release of the Packet Ship Timeline IPTV recorder is a major release which adds support for HD, both in terms of H.264 capture and decoding of Huffman encoded EPG data as used in Freeview HD and Freesat HD in the UK. It also adds a new XML message interface for management of captures and a number of other bug fixes and new features.

In summary the changes are:

- Fix crash during EPG update
- Fix for indexing of BBC HD
- Support DVB text encoding
- Decode Huffman-encoded EPG data and DVB encoded text
- Create 'pscap' user and use it for capture files
- Breaking capture files when data stalls
- Optimised playlists
- Persistency of capture state
- Multiple captures on the same channel
- Add new capture-list, capture-add and capture-remove messages
- Provide selection of programmes by event ID
- Other changes to the messaging interface
- Disabling XMLMesh by default
- Clean shutdown

The main documentation for the 1.2 Hooke release is in the fully updated Packet Ship Timeline Installation & Configuration Guide (ICG-TL). This Release Note just highlights changes for users who were previously using the 1.1 Huygens release.

Fix crash during EPG update

Under certain conditions, the move of a programme within the EPG data would cause a crash and watchdog restart. This has been resolved.

Indexing in BBC HD

Because of additional H.264 syntax elements used in BBC HD channels the 1.1 release was able to capture the stream but was not able to generate an index from it. This has now been fixed and live indexing of BBC HD channels is now working.

¹Robert Hooke invented the balance spring at about the same time as Huygens, and also the anchor escapement



Support for DVB text encoding

The EPG capture process now supports non-ASCII characters in the default DVB character set (approximately ISO 6937)

Decoding Huffman-encoded EPG data

Freeview HD and Freesat HD in the UK use Huffman encoding of EPG data (EIT table). The encoding tables are only available from DTG or the BBC under NDA as a way to enforce implementation of HDCP in HD set-top boxes, since they act as a form of (insecure) encryption of the EPG data.

Despite requesting them for use in legitimate server-based applications Packet Ship have not been able to obtain the official tables, nor could we distribute them if we did have them. However, Open Source reverse-engineered versions of the tables exist in the VDR project, and we have implemented a generic Huffman decoder that can read the same table format (see the full Timeline Installation and Configuration Guide for details). We do not know what form the official tables take, but they should be trivially convertible into the format Timeline currently imports. If (as it is to be hoped) the official tables are made public at some point in the future, we will most likely implement an importer for that format as well.

Note: Packet Ship do not distribute the Freeview/Freesat HD Huffman tables in any form and the Timeline product provides Huffman decoding as a generic feature suitable for any Huffman encoded data. Users wishing to make use of Freeview/Freesat HD EPG data will need to source the tables themselves through official channels or otherwise, at their sole choice and liability.

Table location

The location of the Huffman table files is configured in a **<huffman tables>** path name within the **<epg>** element:

```
<epg>
...
  <huffman tables="/etc/packetship/freesat.t%i"/>
...
</epg>
```

The path name should contain a single “%i” string which represents the table number. Hence the default matches `/etc/packetship/freesat.t1` and `freesat.t2` for tables 1 and 2, which are the standard filenames for the Open Source tables.

Creation of 'pscap' user and group

The server installation process now creates a user and group called 'pscap', and the default configuration now uses this as the user/group to drop privileges to. This is to allow suitable write privileges to be granted to the capture directory (e.g. `/var/lib/packetship/capture`)



Breaking capture files on data stalls

If the multicast data for a captured channel stops for a configurable length of time, the capture daemon will close the current file and wait until the data returns for a further length of time before opening a new one. The effect of this is to properly record gaps in the recorded timeline, which allows the server both to accurately seek to parts of the channel it does have, and also to properly indicate whether a particular programme is available or not.

The time to wait before assuming the channel has stalled is configured in a **timeout** attribute of a **<receive>** element within the capture (default 1 second). The time to wait for data to become stable before starting again is configured by a **resume** attribute in the same element (default 30 seconds). These parameters are subject to the **<default>** capture settings like other parameters, and would usually be set there – e.g.

```
<captures>
  <default>
    ...
    <receive timeout="1 second" resume="30 seconds"/>
    ...
  </default>
  ...
</captures>
```

Optimised playlists

If a capture was running for a long time the number of files generated in a playlist could get out of hand, slowing down both the generation and parsing of the playlist files in the **capture-playlist** process used by the video server to find files to stream. This has now been optimised with a number of configurable parameters in a **<playlist>** element in the capture definition (or more usually, in **<default>**)

Near-live streaming

If the video server requests a channel with no start time (e.g. `rtsp://server/tv/BBC1`), the 1.1 capture daemon would return the entire recording available, with an estimate of the current time within it. The new version returns only back to a configurable 'lead time', allowing the client only to rewind that far. The lead time is set by the **current-lead** attribute of **<playlist>**, and defaults to 1 hour.

Specific programme lead and tail

If the video server specifies a particular programme event (e.g. `rtsp://server/tv/BBC1/event/57347`) the capture daemon returns a playlist covering the entire programme with some extra time at the start and end in case of early or late start. The 'lead' and 'tail' times are configured in **programme-tail** and **programme-lead** attributes of **<playlist>**, and both default to 5 minutes.



Minimum percentage available

The capture daemon can refuse to serve a programme if there is less than a certain percentage available (either because of signal loss or server restarts). The minimum percentage availability is set in a **min-percent** attribute of **<playlist>**, and defaults to 98%.

```
<default>
...
  <playlist programme-lead="5 minutes" programme-tail="5 minutes"
    current-lead="1 hour" min-percent="98" />
...
</default>
```

Multiple captures of the same channel

It is now possible to have multiple captures listening to the same multicast address. Note it only really ever makes sense to have two – one permanent circular buffer recording for catch-up TV, and one externally-controlled recording at specific for long-term DVR storage. If multiple users request the same programme it really does not make sense to record (and index) the same thing more than once; the middleware should manage sharing recordings between users.

Persistent capture state

The ps-captured daemon now stores a list of active captures to an external file so that they can be recreated if the daemon is restarted. The file to store them in is configured in **<disk file>** within the **<captures>** element:

```
<captures>
  <disk file="/var/lib/packetship/capture/captures.xml"/>
  ...
</captures>
```

New capture management messages

A new set of XML message handlers has been added which provide a way to dynamically manage the captures performed by the capture daemon. The new **capture-list**, **capture-add** and **capture-remove** messages are documented in a new version of Application Note *AN-TL-1101: Timeline XML Messaging*.

Selection of programmes by event ID

The EIT event ID for each programme is now recorded and made available in the EPG listing data returned by the **capture-epg-listing** XML request. This event ID can then be quoted in a **capture-playlist** request to select that particular programme.

The **capture-playlist** request with a stated start time also now accepts either an end time or a duration to select the playlist length see Application Note *AN-TL-1101: Timeline XML Messaging* for full details.



Other messaging changes

There were a number of other inconsistencies in the messaging interface which have now been fixed. These are backwards incompatible changes, although they reflect what the documentation stated before!

- The previous version wrongly returned XML attributes with underscores in (e.g. `service_id`, `history_length`) in the responses to `capture-epg-services` and `capture-epg-listing` requests. These are now changed to use hyphens.
- The previous version (contrary to the documentation) used 'channel' attributes in the `capture-start` and `capture-stop`. These are now 'id' attributes consistently with the rest of the messaging interface.
- `capture-start` and `capture-stop` requests now return a simple “ps:ok” or “x:ok” result rather than an empty `capture-start-result` or `capture-stop-result`.
- The format of a start time in a `capture-playlist` request is now more lenient will for example allow missing out the 'T' and seconds values. The main effect of this is to make the format of Streamline 'tv/' URLs less picky.

The messaging interface is now as documented in the new version of Application Note *AN-TL-1101: Timeline XML Messaging*.

XMLMesh disabled by default

To allow the Timeline server to install cleanly in a standalone recorder or demo system, the XMLMesh connection has been disabled by default. However this should be re-enabled for integration with the Streamline video server for playout – please see Application Note *AN-TL-1102: Integration of Streamline in Timeline*.

Clean shutdown

In some configurations using XMLMesh the capture daemon could fail to shut down properly and would need to be explicitly killed. This has been fixed.

Known bugs / limitations

This version has the following known issues:

- Audio-only streams (radio) can be captured but will not index properly.