PPSP Peer Protocol

draft-gu-ppsp-peer-protocol
PPSP WG
IETF 82 Taipei

Rui Cruz (presenter)
Yingjie Gu, Jinwei Xia, Mário Nunes,
David Bryan, João Taveira

Background

- The **PPSP Peer Protocol** (as well as the PPSP Tracker protocol) are **bound to VoD streaming or Live streaming of multimedia**, not “file sharing”.
- Each Peer, from the perspective of a Media Player can be seen as part of a distributed “**streaming server**”
- PPSP should be open to support:
  - Structured Media streaming (SVC/MDC/MVC/multi-bitrate)
  - Unstructured Media streaming (AVC or other formats)
  - but not being involved in the decoding/encoding processes.
- The Media Player application is the entity that should "know" (via a **requester/re-assembler** module) **how** and **what** to **request** (to a Supplier Peer) and **decode** the received Structured Media in order to “prepare” it to **present** to the User.

Terminology

- Scalable Streaming (Structured Media)
  - Multiple Description Coding (MDC): multiple additive descriptions can be independently played-out to refine the quality of the video when combined together.
  - Scalable Video Coding (SVC): nested dependent enhancement layers (hierarchical levels of quality), refine the quality of lower layers, from the lowest level (the playable Base Layer).
  - Multiple View Coding (MVC): multiple views allow the video to be played in stereoscopic 3D when the views are combined together.

Terminology

- SEGMENT (of partitioned media)
  - is a resource that can be identified by an ID, an HTTP-URL or a byte-range, and used by a Peer for the purpose of storage, advertisement and exchange among peers.
- SUBSEGMENT (of partitioned media)
  - the smallest unit within segments which may be indexed at the segment level.
- CHUNK
  - is a generic term used to refer to a **SEGMENT** or **SUBSEGMENT** of partitioned streaming media.

Terminology

- LEECH Peer
  - – A Peer that requests specific media content from other Peers.
- SEED Peer
  - A Peer that can supply all the media content chunks.
- Sender Peer
  - – A Peer that can supply the corresponding chunks requested by a Leech Peer.

Main functional entities related with PPSP

- **Client Media Player**
  - is the entity providing a direct interface to the end user at the client device, and includes the functions to select, request, decode and render contents.
  - interfaces with the Peer using request and response mechanisms.
- **Peer**
  - Is a logical entity at the client device embedding the P2P core engine, with a client serving side interface to respond to Client Media Player requests and a network side interface to exchange data and PPSP signaling with Trackers and with other Peers.
- **Tracker**
  - is a logical entity that maintains the lists, as well as the status, of PPSP active peers storing and exchanging chunks for a specific media content.

Design Philosophy I

- Support of Structured Media Streaming
  - Scheduling of structured media chunks can be optimized from monitoring of network and host conditions
  - Decoupling download/upload from presentation of media (Peer is not involved in media decoding)
  - Each Peer not only downloads the stream of interest (being presented) but also contributes on other streams (other swarms)
  - Peers can be assigned to “distribution groups”
- Bandwidth adaptation
  - In bandwidth-rich periods the media quality is maximized (average up bandwidth > full media rate)
  - In bandwidth-deficient periods the media quality is reduced (average up bandwidth < full media rate)

Design Philosophy II

- Chunk-based (mesh) pull-mode
  - Peers self-organize (by proximity, media chunk availability, etc.)
  - Chunks (segments/subsegments) are **specifically requested** to suppliers and scheduled:
    - From the Supplier side in a pair-wise bandwidth allocation, i.e., higher upload -> more bandwidth shared -> more segments -> higher media rate
    - From the Receiver side, by periodic evaluation of network conditions and suppliers (dropping the worst)

Protocol Overview

- The signaling steps for a Peer (LEECH) wishing to participate either in a Live streaming or a VoD or offline video is as follows:
  1. The leech peer using PPSP Peer Protocol messages, establishes a connection to at least one of the peers in the Peerlist, based on the known PeerID and Peer IP address.
  2. The peer sends request to selected candidate peers including one or more of the following information:
     a. Request for the content availability;
     b. Notify own content availability to the candidate peer;
     c. Request peer properties of the candidate peer;
     d. Notify own peer properties to the candidate peer;
     e. Request for additional peerlist;
     f. Negotiates Data Transport protocol.
  3. The peers exchange the actual chunks of data, using the mechanism/protocol negotiated.

Peer Signaling Primitives

- **GET_CHUNKMAP:**
  - sent from a Leech peer to one or more remote peers in order to receive the map of chunks (and/or buffer map) of a content (of a swarm identified by SwarmID) the other peer presently stores.

Peer Signaling Primitives

- **GET_CHUNK:**
  - – sent from a leech peer to sender peer in order to request the delivery of specific media content chunks.

Peer Signaling Primitives

- **GET_STATUS:**
  - sent from a leech peer to one or more remote peers in order to request the corresponding properties and status of the sender peers.

Peer Signaling Primitives

- **GET_PEERLIST:**
  - – sent from a leech peer to one or more remote peers in order to refresh/update the list of active peers in the swarm and corresponding properties.

Peer Signaling Primitives

- **TRANSPORT_NEGOTIATION:**
  - sent from a leech peer to a sender peer in order to negotiate the underlying data transport protocol. Leech peer may provide a set of transport protocols it supports to sender peer, and leave send peer to choose its preferences.

Comments are welcomed!

THANK YOU !

Media Presentation Description

- Provides formats to enable efficient and high-quality delivery of streaming services over the Internet
- Enables reuse of existing technologies (containers, codecs, DRM etc.)
- Enables very high user-experience (low start-up, no rebuffering, trick modes)
- Enables selection based on network capabilities, device capabilities, user preferences
- Moves intelligence from network to client,
- Enables distribution flexibility (e.g., live, on-demand, time-shift viewing)
- Provides Content Descriptors for Protection, Accessibility, Rating, etc.