PPSP
Internet Draft

Y. Zhang
China Mobile
N.Zong
Huawei Tech

Intended status: Informational
Expires: April 2013

October 20, 2012

Problem Statement and Requirements of Peer-to-Peer Streaming Protocol (PPSP)
draft-ietf-ppsp-problem-statement-11.txt

Abstract

Peer-to-Peer (P2P for short) streaming systems show more and more popularity in current Internet with proprietary protocols. This document identifies problems of the proprietary protocols, proposes the development of Peer to Peer Streaming Protocol (PPSP) including the tracker and peer protocol, and discusses the scope, requirements and use cases of PPSP.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

1.1. Background

Streaming traffic is among the largest and fastest growing traffic on the Internet [Cisco], where peer-to-peer (P2P) streaming contributes substantially. With the advantage of high scalability and fault tolerance against single point of failure, P2P streaming applications are able to distribute large-scale, live and video on demand (VoD) streaming programs to a large audience with only a handful of servers. What's more, along with the players like CDN providers joining in the effort of using P2P technologies in distributing their serving streaming content, there are more and more various players in P2P streaming ecosystem.

Given the increasing integration of P2P streaming into the global content delivery infrastructure, the lack of an open, standard P2P streaming signaling protocol suite becomes a major missing component. Almost all of existing systems use their proprietary protocols. Multiple, similar but proprietary protocols result in repetitious development efforts for new systems, and the lock-in effects lead to substantial difficulties in their integration with other players like CDN. For example, in the enhancement of existing caches and CDN systems to support P2P streaming, proprietary protocols may increase the complexity of the interaction with different P2P streaming applications.

In this document we propose the development of an open P2P Streaming Protocol, which is abbreviated as PPSP, to standardize signaling operations in P2P streaming systems to solve the above problems.

1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

2. Terminology and concepts

Chunk: A chunk is a basic unit of data block organized in P2P streaming for storage, scheduling, advertisement and exchange among peers [VoD]. A chunk size varies from several KBs to several MBs in different systems. In case of MBs size chunk scenario, a sub-chunk structure named piece is often defined to fit in a single transmitted packet. A streaming system may use different granularities for different usage, e.g., using chunks during data exchange, and using a larger unit such as a set of chunks during advertisement.

Chunk ID: The identifier of a chunk in a content stream.

Client: A client in general refers to the service requester in client/server computing paradigm. In this draft a client also refers to a participant in a P2P streaming system that only receives streaming content. In some cases, a node not having enough computing and storage capabilities will act as a client. Such node can be viewed as a specific type of peer.

Content Distribution Network (CDN): A CDN is a collection of nodes that are deployed, in general, at the network edge like Points of Presence (POP) or Data Centers (DC) and that store content provided by the original content servers. Typically, CDN nodes serve content to the clients located nearby topologically.

Live streaming: It refers to a scenario where all clients receive streaming content for the same ongoing event. It is desired that the lags between the play points of the clients and streaming source be small.

P2P cache: A P2P cache refers to a network entity that caches P2P traffic in the network and, either transparently or explicitly, streams content to other peers.

Peer: A peer refers to a participant in a P2P streaming system that not only receives streaming content, but also caches and streams streaming content to other participants.

Peer list: A list of peers which are in a same swarm maintained by the tracker. A peer can fetch the peer list of a swarm from the tracker or from other peers in order to know which peers have the required streaming content.

Peer ID: The identifier of a peer such that other peers, or the tracker, can refer to the peer by using its ID.

PPSP: The abbreviation of Peer-to-Peer Streaming Protocols. PPSP refer to the key signaling protocols among various P2P streaming system components, including the tracker and the peer.

Tracker: A tracker refers to a directory service that maintains a list of peers participating in a specific audio/video channel or in the distribution of a streaming file. Also, the tracker answers peer list queries received from peers. The tracker is a logical component which can be centralized or distributed.

Video-on-demand (VoD): It refers to a scenario where different clients may watch different parts of the same recorded streaming with downloaded content.

Swarm: A swarm refers to a group of peers who exchange data to distribute chunks of the same content (e.g. video/audio program, digital file, etc) at a given time.

Swarm ID: The identifier of a swarm containing a group of peers sharing a common streaming content.

Super-node: A super-node is a special kind of peer deployed by ISPs. This kind of peer is more stable with higher computing, storage and bandwidth capabilities than normal peers.

3. Problem statement

The problems caused by proprietary protocols for P2P streaming applications are listed as follows.

3.1. Heterogeneous P2P traffic and P2P cache deployment

ISPs are faced with different P2P streaming application introducing substantial traffic into their infrastructure, including their backbone and their exchange/interconnection points. P2P caches are used by ISPs in order to locally store content and hence reduce the P2P traffic. P2P caches usually operate at the chunk or file granularity.

However, unlike web traffic that is represented by HTTP requests and responses and therefore allows any caching device to be served (as long as it supports HTTP), P2P traffic is originated by multiple P2P applications which require the ISPs to deploy different type of caches for the different types of P2P streams.

This increases both engineering and operational costs dramatically.

3.2. QoS issue and CDN deployment

P2P streaming is often criticized due to its worse QoS performance compared to client/server streaming (e.g., longer startup delay, longer seek delay and channel switch delay). Hybrid CDN/P2P is a good approach in order to address this problem [Hybrid CDN P2P].

In order to form the hybrid P2P+CDN architecture, the CDN must be aware of the specific P2P streaming protocol in the collaboration. Similarly to what is described in section 3.1, proprietary P2P protocols introduce complexity and deployment cost of CDN.

3.3. Extended applicability in mobile and wireless environment

Mobility and wireless are becoming increasingly important in today's Internet, where streaming service is a major usage. It's reported that the average volume of video traffic on mobile networks has risen up to 50% in the early of 2012 [ByteMobile]. There are multiple prior studies exploring P2P streaming in mobile and wireless networks [Mobile Streaming1] [Mobile Streaming2].

However it's difficult to directly apply current P2P streaming protocols (even assuming we can re-use some of the proprietary ones) in mobile and wireless networks.

Following are some illustrative problems:
First, P2P streaming assumes a stable Internet connection in downlink and uplink direction, with decent capacity and peers that can run for hours. This isn't the typical setting for mobile terminals. Usually the connections are unstable and expensive in terms of energy consumption and transmission (especially in uplink direction). To enable mobile/wireless P2P streaming feasible, trackers may need more information on peers like packet loss rate, peer battery status and processing capability during peer selection compared to fixed peers. Unfortunately current protocols don't convey this kind of information.

Second, current practices often use a "bitmap" message in order to exchange chunk availability. The message is of kilobytes in size and exchanged frequently, e.g., an interval of several seconds or less. In a mobile environment with scarce bandwidth, the message size may need to be shortened or it may require more efficient methods for expressing and distributing chunk availability information, which is different from wire-line P2P streaming.

Third, for a resource constraint peer like mobile handsets or set-top boxes (STB), there are severe contentions on limited resource when using proprietary protocols. The terminal has to install different streaming client software for different usages, e.g., some for movies and others for sports. Each of these applications will compete for the same set of resources even when it is sometimes running in background mode. PPSP can alleviate this problem with the basic idea that the "one common client software with PPSP and different scheduling plug-ins" is better than "different client software running at the same time" in memory and disk consumption.

4. Tasks of PPSP: Standard peer to peer streaming protocols

PPSP is targeted to standardize signaling protocols to solve the above problems that support either live or VoD streaming. PPSP targets tracker-based architectures, as well as tracker-less architectures. In tracker-less architecture, the tracker functionality is distributed in decentralized peers. In the following part of this section, the tracker is a logic conception, which can be implemented in a dedicated tracker server or in peers.

The PPSP design includes a signaling protocol between trackers and peers (the PPSP "tracker protocol") and a signaling protocol among the peers (the PPSP "peer protocol") as shown in Figure 1. The two protocols enable peers to receive streaming content within the time constraints.

PPSP design in general needs to solve the following challenges, e.g.,:

1) When joining a swarm, how does a peer know which peers it should contact for content?

2) After knowing a set of peers, how does a peer contact with these peers? in which manner?

3) How to choose peers with better service capabilities, and how to collect such information from peers?

4) How to improve the efficiency of the communication, e.g. compact on-the-wire message format and suitable underlying transport mechanism (UDP or TCP)?

5) How to improve the robustness of the system using PPSP, e.g. when the tracker fails? How to make the tracker protocol and the peer protocol loose coupled?

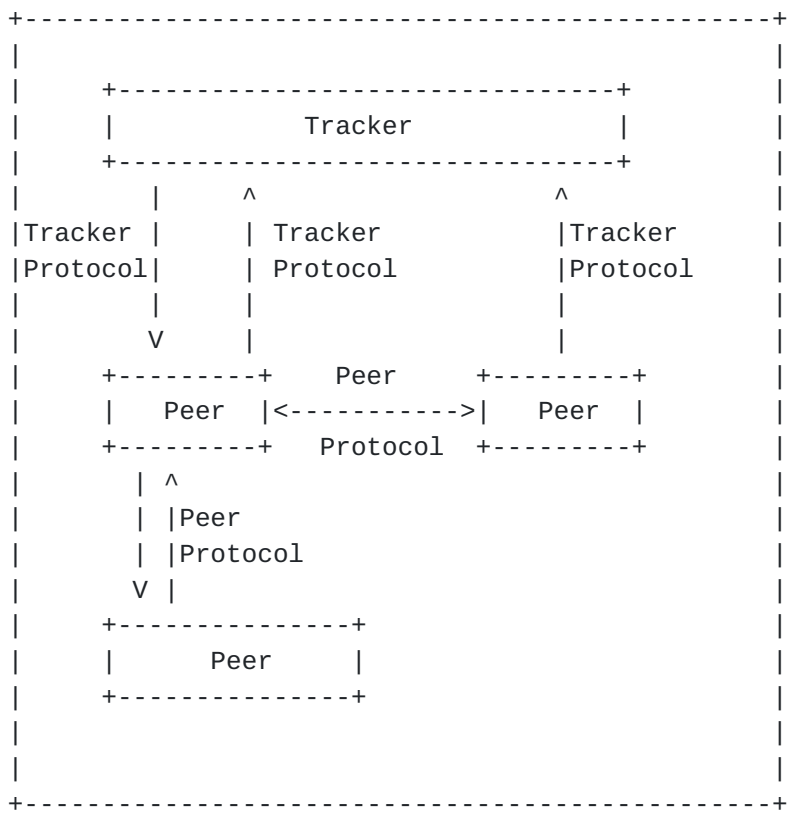

Figure 1 PPSP System Architecture

4.1. Tasks and design issues of Tracker protocol

The tracker protocol handles the initial and periodic exchange of meta-information between trackers and peers, such as peer list and content information.

Therefore tracker protocol is best modeled as a request/response protocol between peers and trackers, and will carry information needed for the selection of peers suitable for real-time/VoD streaming.

Special tasks for the design of the tracker protocol are listed as follows. This is a high-level task-list. The detailed requirements on the design of the tracker protocol are explicated in section 6.

1) How should a peer be globally identified? This is related to the peer ID definition, but irrelevant to how the peer ID is generated.

2) How to identify different peers, e.g. peers with public or private IP address, peers behind or not behind NAT, peers with IPV4 or IPV6 addresses, peers with different property?

3) How to enable the tracker protocol light-weight, since a tracker may need to server large amount of peers? This is related to the encoding issue (e.g., Binary based or Text based) and keep-alive mechanism.

4) How can the tracker be able to report optimized peer list to serve a particular content. This is related to status statistic, with which the tracker can be aware of peer status and content status.

PPSP tracker protocol will consider all these issues in the design according to the requirements from both peer and tracker perspective and also taking into consideration deployment and operation perspectives.

4.2. Tasks and design issues of Peer protocol

The peer protocol controls the advertising and exchange of content between the peers.

Therefore peer protocol is modeled as a gossip-like protocol with periodic exchanges of neighbor and chunk availability information.

Special tasks for the design of the peer protocol are listed as follows. This is a high-level task-list. The detailed requirements on the design of the peer protocol are explicated in section 6.

1) How does the certain content be globally identified and verified? Since the content can be retrieved from everywhere, how to ensure the exchanged content between the peers authentic?

2) How to identify the chunk availability in the certain content? This is related to the chunk addressing and chunk state maintenance. Considering the large amount of chunks in the certain content, light-weight expression is necessary.

3) How to ensure the peer protocol efficient? As we mentioned in section 3, the chunk availability information exchange is quite frequent. How to balance the information exchange size and amount is a big challenge. What kind of encoding and underlying transport mechanism (UDP or TCP) is used in the messages?

PPSP peer protocol will consider all the above issues in the design according to the requirements from the peer perspective.

5. Use cases of PPSP

This section is not the to-do list for the WG, but for the explanatory effect to show how PPSP could be used in practice.

5.1. Worldwide provision of live/VoD streaming

The content provider can increase live streaming coverage by introducing PPSP in between different providers.

We suppose a scenario that there is only provider A (e.g,, in China) providing the live streaming service in provider B(e.g., in USA) and C(e.g., in Europe)'s coverage. Without PPSP, when a user (e.g., a Chinese american) in USA requests the program to the tracker (which is located in A's coverage), the tracker may generally return to the user with a peer list including most of peers in China, because generally most users are in China and there are only few users in USA. This may affect the user experience.

But if we can use the PPSP tracker protocol to involve B and C in the cooperative provision, as shown in Figure 2, even when the streaming is not hot to attract many users in USA and Europe to view, the tracker in A can optimally return the user with a peer list including B's Super-nodes (SN for short) and C's SN to provide a better user performance.

Furthermore User@B and User@C can exchange data (availability) with these local SNs using the peer protocol.

5.2. Enabling CDN for P2P VoD streaming

Figure 3 shows the case of enabling CDN to support P2P VoD streaming from different content providers by introducing PPSP inside CDN overlays. It is similar to Figure 2 except that the intermediate SNs are replaced by 3rd party CDN surrogates. The CDN nodes talk with the different streaming systems (including trackers and peers) with the same PPSP protocols.

Furthermore the interaction between the CDN nodes can be executed by either existing (maybe proprietary) protocols or the PPSP peer protocol. The peer protocol is useful for building new CDN systems (e.g., operator CDN) supporting streaming in a low cost.

Note that for compatibility reason both HTTP streaming and P2P streaming can be supported by CDN from the users' perspective.

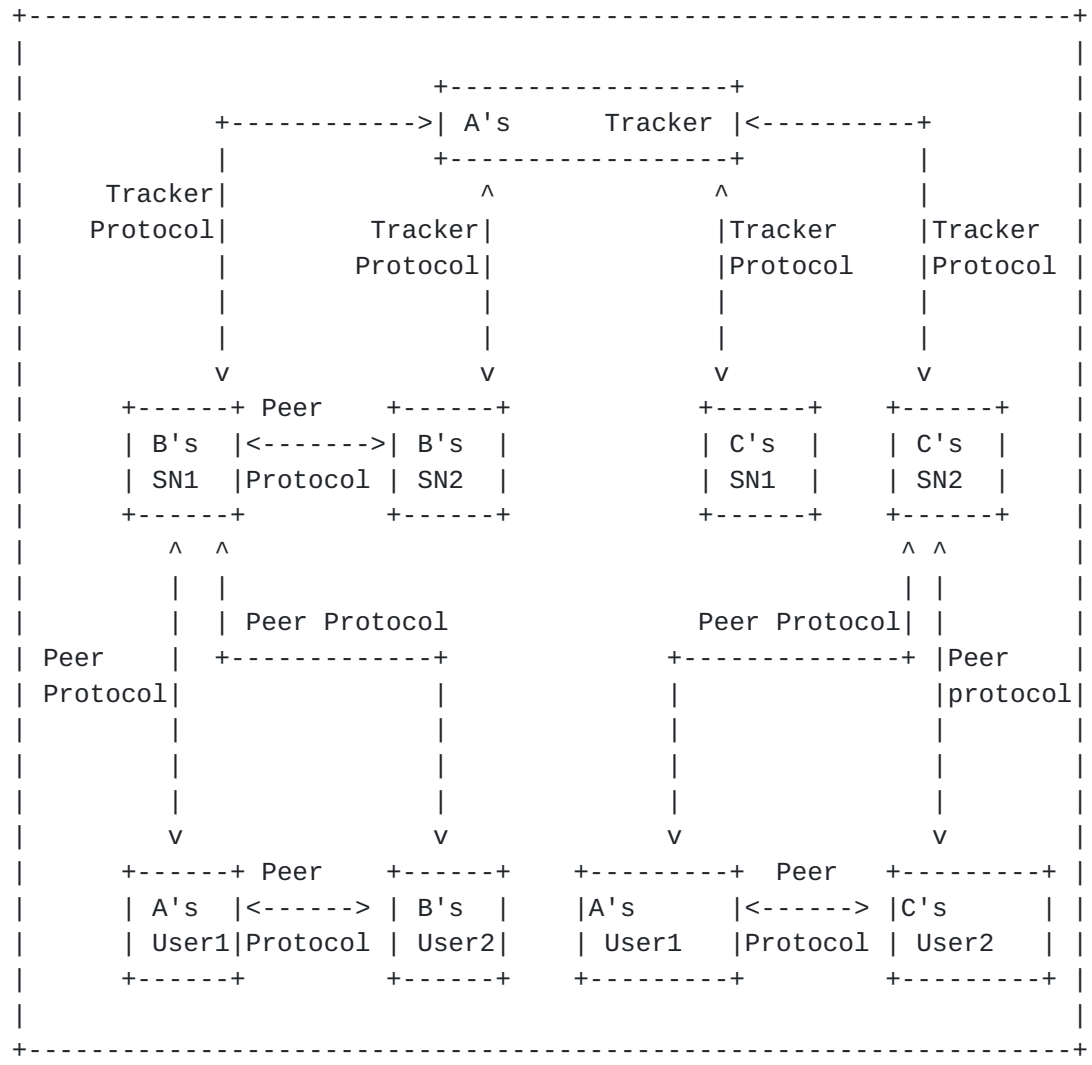

Figure 2 Cooperative Vendors Interaction

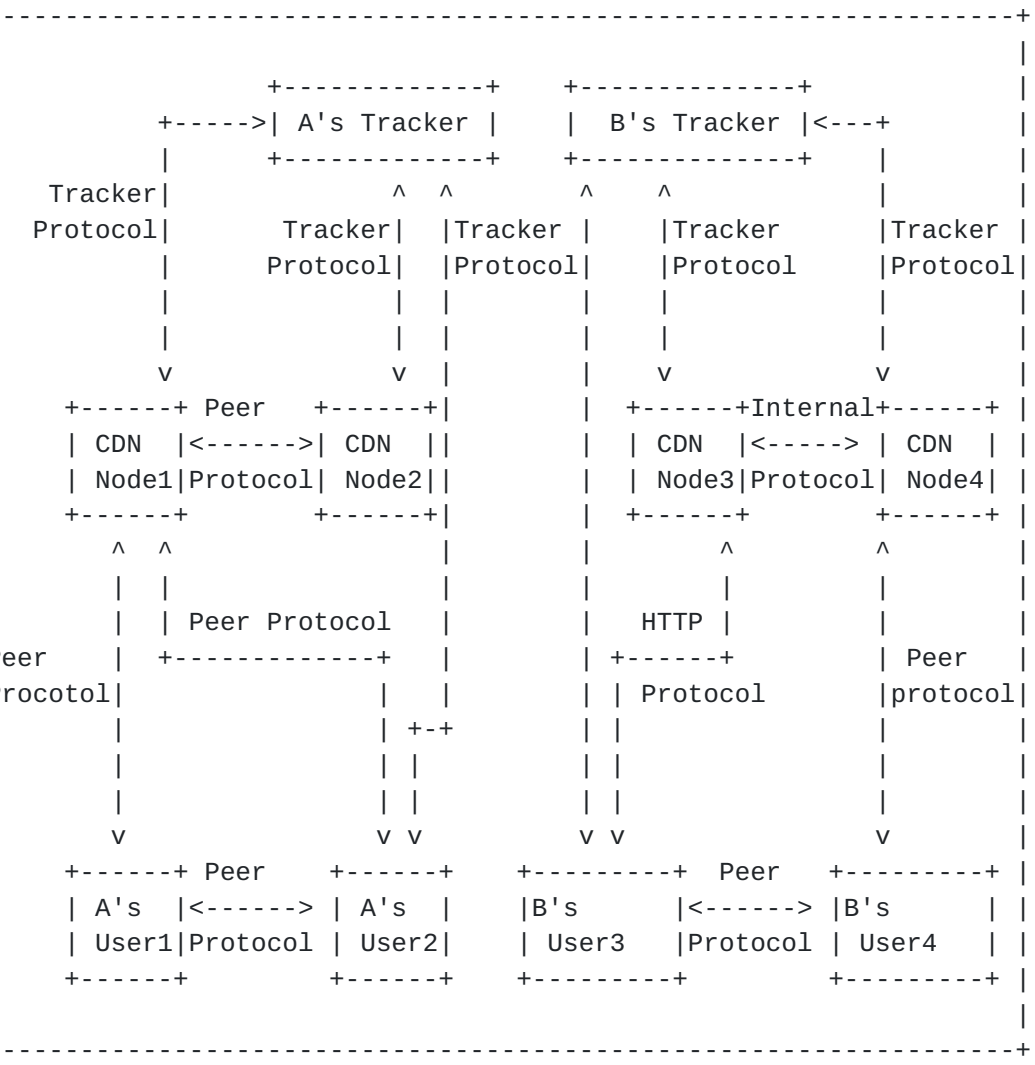

Figure 3 CDN Supporting P2P Streaming

5.3. Cross-screen streaming

In this scenario PC, STB/TV and mobile terminals from both fixed network and mobile/wireless network share the streaming content. With PPSP, peers can identify the types of access networks, average load, peer abilities and get to know what content other peers have even in different networks (potentially with the conversion of the content availability expression in different networks) as shown in Figure 4.

Such information will play an important role on selecting suitable peers, e.g., a PC or STB is more likely to provide stable content and a mobile peer within a high-load cell is unlikely to be selected, which may otherwise lead to higher load on the base station.

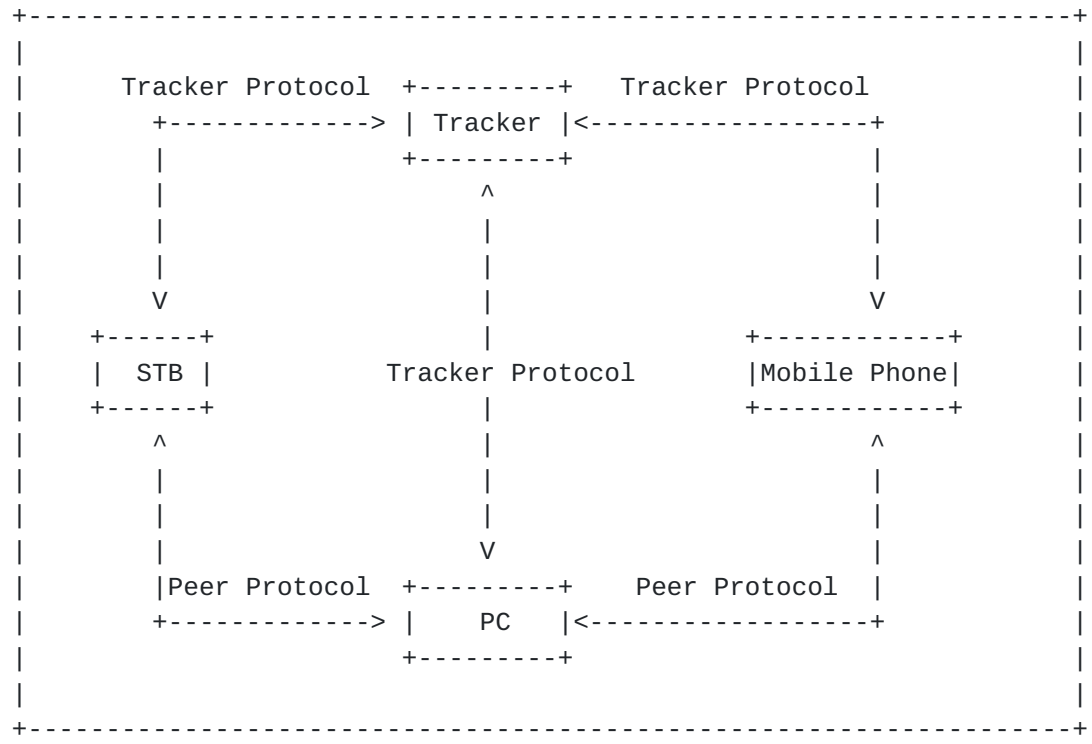

Figure 4 Heterogeneous P2P Streaming with PPSP

5.4. Cache service supporting P2P streaming

In Figure 5, when peers request the P2P streaming data, the cache nodes intercept the requests and ask for the frequently visited content (or part of) on behalf of the peers. To do this, it asks the tracker for the peer list and the tracker replies with external peers in the peer list. After the cache nodes exchange data with these peers, it can also act as a peer and report what it caches to the tracker and serve inside requesting peers afterward. This operation greatly decreases the inter-network traffic in many conditions and increases user experience.

The cache nodes do not need to update their library when new applications supporting PPSP are introduced, which reduces the cost.

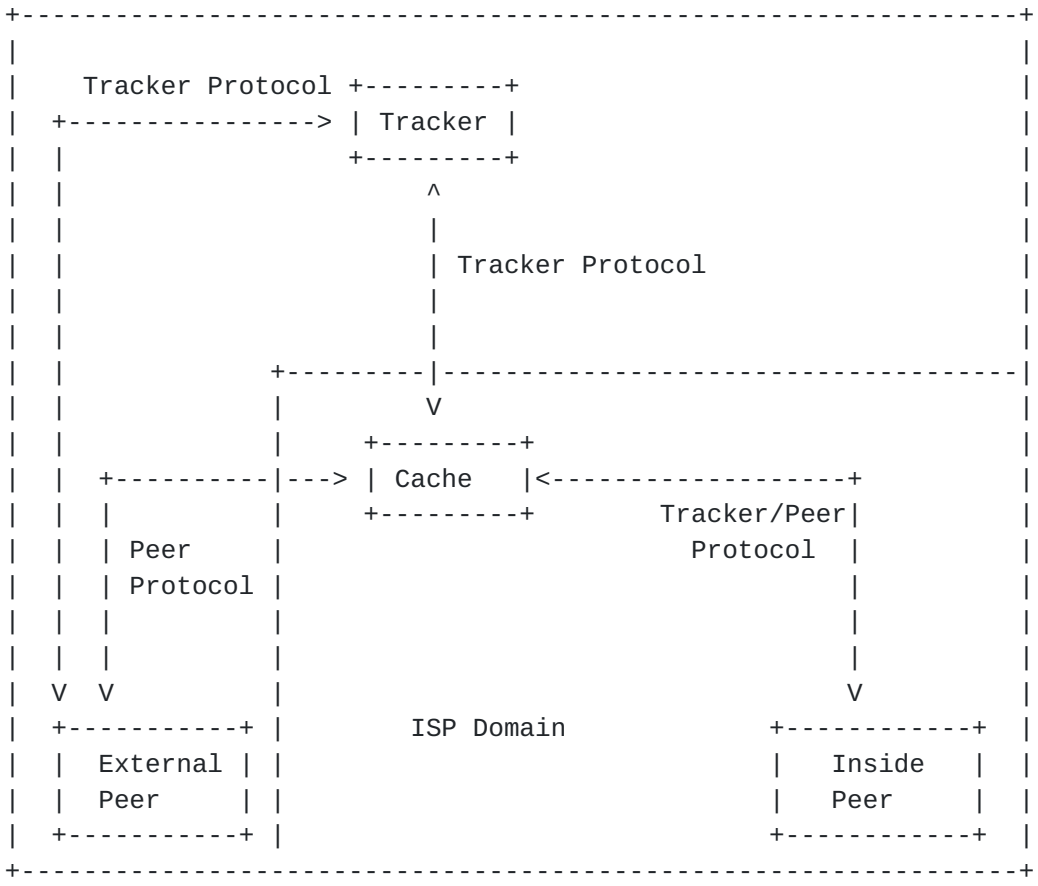

Figure 5 Cache Service Supporting Streaming with PPSP

5.5. Proxy service supporting P2P streaming

5.5.1. Home Networking Scenario

For applications where the peer is not co-located with the Media Player in the same device (e.g. the peer is located in a Home Media Gateway), we can use a PPSP Proxy, as shown in figure 6.

As shown in figure 6, the PPSP Proxy terminates both the tracker and peer protocol allowing the legacy presentation devices to access P2P streaming content. In figure 6 the DLNA protocol [DLNA] is used in order to communicate with the presentation devices thanks to its wide deployment. Obviously, other protocols can also be used.

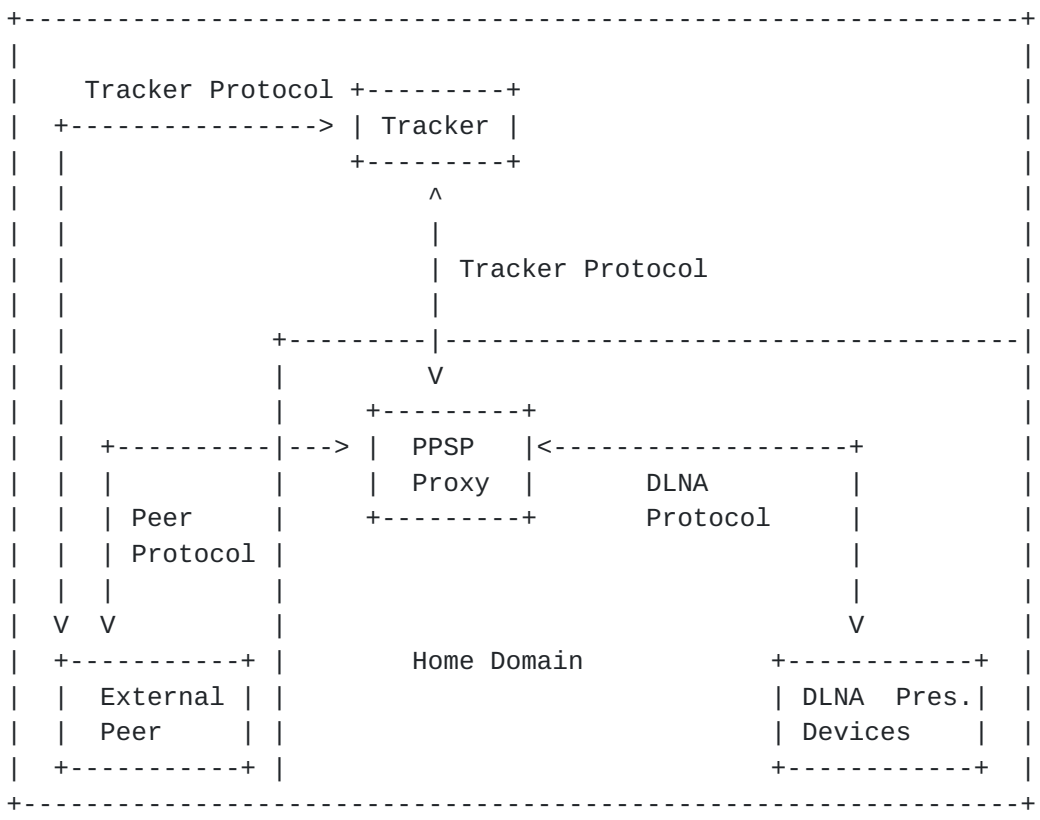

Figure 6 Proxy service Supporting P2P Streaming

5.5.2. Browser-based HTTP Streaming

P2P Plug-ins are often used in browser-based environment in order to stream content. With P2P plug-ins, HTTP streaming can be turned into a de facto P2P streaming. From the browser (and hence the user) perspective, it's just HTTP based streaming but the PPSP capable plug-in can actually accelerate the process by leveraging streams from multiple sources/peers [P2PYoutube]. In this case the plug-ins behave just like the proxy.

6. Requirements of PPSP

This section enumerates the requirements that should be considered when designing PPSP.

6.1. Basic Requirements

PPSP.REQ-1: Each peer MUST have a unique ID (i.e. peer ID) in a swarm.

It's a basic requirement for a peer to be uniquely identified in a swarm that other peers or tracker can refer to the peer by ID.

PPSP.REQ-2: The streaming content MUST be uniquely identified by a swarm ID.

A swarm refers to a group of peers sharing the same streaming content. A swarm ID uniquely identifies a swarm. The swarm ID can be used in two cases: 1) a peer requests the tracker for the peer list indexed by a swarm ID; 2) a peer tells the tracker about the swarms it belongs to.

PPSP.REQ-3: The streaming content MUST allow to be partitioned into chunks.

PPSP.REQ-4: Each chunk MUST have a unique ID (i.e. chunk ID) in the swarm.

Each chunk must have a unique ID in the swarm so that the peer can understand which chunks are stored in which peers and which chunks are requested by other peers.

PPSP.REQ-5: The tracker and peer protocol together MUST facilitate achieving QoS acceptable to both live and VoD streaming application.

There are basic QoS requirements for streaming systems. Setup time to receive a new streaming channel or to switch between channels should be reasonably small. End to end delay, which consists of the time between content generation (e.g., a camera) and content consumption (e.g., a monitor), will become critical in case of live streaming especially in provisioning of sport events where end to end delay of 1 minute and more are not acceptable.

For instance, the tracker and peer protocol can carry QoS related parameters (e.g. video quality and delay requirements) together with the priorities of these parameters in addition to the measured QoS situation (e.g., performance, available uplink bandwidth) of content providing peers.

There are also some other possible mechanisms like addition of super peers, in-network storage, request of alternative peer addresses, and the usage of QoS information for advanced peer selection mechanisms.

PPSP.REQ-6: The tracker and peer protocol do not include the algorithm required for scalable streaming. However, the tracker and peer protocol SHOULD NOT restrict or place limits on any such algorithm.

PPSP.REQ-7: The tracker SHOULD be robust, i.e., when centralized tracker fails, the P2P streaming system should still work by supporting distributed trackers.

6.2. PPSP Tracker Protocol Requirements

PPSP.TP.REQ-1: The tracker protocol MUST allow the peer to solicit the peer list from the tracker with respect to a specific swarm in a query and response manner.

The tracker request message may also include the requesting peer's preference parameter (e.g. preferred number of peers in the peer list) or preferred downloading bandwidth. The tracker will then be able to select an appropriate set of peers for the requesting peer according to the preference.

PPSP.TP.REQ-2: The tracker SHOULD support generating the peer list with the help of traffic optimization services, e.g. ALTO [I-D.ietf-alto-protocol].

PPSP.TP.REQ-3: The tracker protocol MUST support the report of the peer's activity in the swarm to the tracker.

PPSP.TP.REQ-4: The tracker protocol SHOULD support the report of the peer's chunk availability information to the tracker when tracker needs such information to steer peer selection.

PPSP.TP.REQ-5: The chunk availability information between peer and tracker MUST be expressed in a compactable method.

The peers may report chunk availability digest information (i.e., compact expression of chunk availability) to the tracker when possible in order to decrease the bandwidth consumption in mobile networks. For example, if a peer has a bitmap like 111111...1(one hundred continuous 1)xxx..., the one hundred continuous "1" can be expressed by one byte with seven bits representing the number of "1", i.e., "one hundred" and one bit representing the continuous sequence is "1" or "0". In this example, 100-8=92 bits are saved. Considering

the frequency of exchange of chunk availability and the fact that many bitmaps have a quite long length of continuous "1" or "0", such compression is quite useful.

PPSP.TP.REQ-6: The tracker protocol SHOULD support the report of the peer's status to the tracker when tracker needs such information to steer peer selection.

For example, peer status can be online time, physical link status including DSL/WiFi/etc., battery status, processing capability and other capabilities of the peer. Therefore, the tracker is able to select better candidate peers for streaming.

PPSP.TP.REQ-7: The tracker protocol MUST allow the tracker to authenticate the peer.

This ensures that only the authenticated users can access the original content in the P2P streaming system.

6.3. PPSP Peer Protocol Requirements

PPSP.PP.REQ-1: The peer protocol MUST allow the peer to solicit the chunk information from other peers in a query and response manner.

PPSP.PP.REQ-2: The chunk information exchanged between a pair of peers MUST NOT be passed to other peers, unless validated (e.g. prevent hearsay and DoS).

PPSP.PP.REQ-3: The peer protocol SHOULD allow the peer to solicit an additional list of peers to that received from the tracker.

It is possible that a peer may need additional peers for certain streaming content. Therefore, it is allowed that the peer communicates with other peers in the current peer list to obtain an additional list of peers in the same swarm.

PPSP.PP.REQ-4: When used for soliciting additional list of peers, the peer protocol MUST contain swarm-membership information of the peers that have explicitly indicated they are part of the swarm, verifiable by the receiver.

PPSP.PP.REQ-5: The additional list of peers MUST only contain peers which have been checked to be valid and online recently (e.g. prevent hearsay and DoS).

PPSP.PP.REQ-6: The peer protocol MUST support the report of the peer's chunk availability update.

Due to the dynamic change of the buffered streaming content in each peer and the frequent join/leave of peers in the swarm, the streaming content availability among a peer's neighbors (i.e. the peers known to a peer by getting the peer list from either tracker or peers) always changes and thus requires being updated on time. This update should be done at least on demand. For example, when a peer requires finding more peers with certain chunks, it sends a message to some other peers in the swarm for streaming content availability update. Alternatively, each peer in the swarm can advertise its streaming content availability to some other peers periodically. However, the detailed mechanisms for this update such as how far to spread the update message, how often to send this update message, etc. should leave to the algorithms, rather than protocol concerns.

PPSP.PP.REQ-7: The chunk availability information between peers MUST be expressed in a compactable method.

The peers may exchange chunk availability digest information with other peers, when possible, in order to decrease the messages bandwidth consumption.

PPSP.PP.REQ-8: The peer protocol MUST support the exchange of additional information, e.g. status about the peers.

This information can be, for instance, information about the access link or information about whether a peer is running on battery or is connected to a power supply. With such information, a peer can select more appropriate peers for streaming.

PPSP.PP.REQ-9: The peer protocol is RECOMMENDED to be carried over UDP.

This would reduce the message overhead and help the peers to retrieve streaming content in a timely manner.

7. Security Considerations

This document discusses the problem statement and requirements around P2P streaming protocols without specifying the protocols. However we believe it is important for the reader to understand areas of security introduced by the P2P nature of the proposed solution. The main issue is the usage of un-trusted entities (peers) for service provisioning. For example, malicious peers may:

- Originate denial of service (DOS) attacks to the trackers by sending large amount of requests with the tracker protocol;

- Originate fake information on behalf of other peers;
- Originate fake information about chunk availability;

For example, malicious peers/trackers may:
- Originate reply instead of the regular tracker (man in the middle attack).

We list some important security requirements for PPSP protocols as below:

PPSP.SEC.REQ-1: PPSP MUST support closed swarms, where the peers are authenticated or in a private network.

This ensures that only the trusted peers can access the original content in the P2P streaming system. This can be achieved by security mechanisms such as peer authentication and/or key management scheme.

PPSP.SEC.REQ-2: Confidentiality of the streaming content in PPSP SHOULD be supported.

In order to achieve this, PPSP should provide mechanisms to encrypt the data exchange among the peers.

PPSP.SEC.REQ-3: PPSP SHOULD support identifying badly behaving peers, and exclude or reject them from the P2P streaming system.

PPSP.SEC.REQ-4: Integrity of the streaming content in PPSP MUST be supported to provide a peer with the possibility to identify unauthentic content (undesirable modified by other entities rather than its genuine source).

In a P2P live streaming system a polluter can introduce corrupted chunks. Each receiver integrates into its playback stream the polluted chunks it receives from its neighbors. Since the peers forwards chunks to other peers, the polluted content can potentially spread through the P2P streaming network.

The PPSP protocol specifications will document the expected threats (and how they will be mitigated by each protocol) and also considerations on threats and mitigations when combining both protocols in an application. This will include privacy of the users and protection of the content distribution.

PPSP.SEC.REQ-5: The security mechanisms in PPSP, such as key management and checksum distribution SHOULD scale well in P2P streaming systems.

PPSP.SEC.REQ-6: Existing P2P security mechanisms SHOULD be re-used as much as possible in PPSP, to avoid developing new security mechanisms.

8. IANA Considerations

This document has no actions for IANA.

9. Acknowledgments

Thank you to J.Seng, G. Camarillo, R. Yang, C. Schmidt, R. Cruz, Y. Gu, A.Bakker and S. Previdi for contribution to many sections of this draft. Thank you to C. Williams, V. Pascual and L. Xiao for contributions to PPSP requirements section.
We would like to acknowledge the following people who provided review, feedback and suggestions to this document: M. Stiemerling, D. Bryan, E. Marocco, V. Gurbani, R. Even, H. Zhang, D. Zhang, J. Lei, H.Song, X.Jiang, J.Seedorf, D.Saumitra, A.Rahman, L.Deng, J.Pouwelse and W.Eddy.

This document was prepared using 2-Word-v2.0.template.dot.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[Cisco] Cisco Visual Networking Index: Forecast and Methodology, 2009-2014, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html

[VoD] Y. Huang et al, Challenges, "Design and Analysis of a Large-scale P2P-VoD System", Sigcomm08.

[ByteMobile] http://www.bytemobile.com/news-events/2012/archive_230212.html

[Mobile Streaming1] Streaming to Mobile Users in a Peer-to-Peer Network, J. Noh et al, MOBIMEDIA '09.

[Mobile Streaming2] J.Peltotaloet al.,"A real-time Peer-to-Peer streaming system for mobile networking environment", in Proceedings of the INFOCOM and Workshop on Mobile Video Delivery (MoVID '09).

[Hybrid CDN P2P]D. Xu et al, "Analysis of a CDN-P2P hybrid architecture for cost-effective streaming media distribution," Springer Multimedia Systems, vol.11, no.4, pp.383-399, 2006.

[PPTV] http://www.pptv.com

[PPStream] http://www.ppstream.com

[DLNA] http://www.dlna.org

[P2PYoutube] https://addons.opera.com/en/extensions/details/p2p-youtube/

Authors' Addresses

Yunfei Zhang
China Mobile Communication Corporation
zhangyunfei@chinamobile.com

Ning Zong
Huawei Technologies Co., Ltd.
zongning@huawei.com