Internet Engineering Task Force (IETF)
Request for Comments: 6972
Category: Informational
ISSN: 2070-1721

Y. Zhang
Coolpad
N. Zong
Huawei Technologies

July 2013

Problem Statement and Requirements of the Peer-to-Peer Streaming Protocol (PPSP)

Abstract

Peer-to-Peer (P2P) streaming systems becoming more and more popular on the Internet, and most of them are using proprietary protocols. This document identifies problems associated with proprietary protocols; proposes the development of the Peer-to-Peer Streaming Protocol (PPSP), which includes the tracker and peer protocols; and discusses the scope, requirements, and use cases of PPSP.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6972.

Table of Contents

1. Introduction

1.1. Backgrounds

Streaming traffic is among the largest and fastest growing traffic on the Internet [Cisco]. Peer-to-Peer (P2P) streaming contributes substantially to this growth. With the advantage of high scalability and fault tolerance against a 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. More and more providers are joining the P2P streaming ecosystem, e.g., Content Distribution Networks (CDN) providers started using P2P technologies to distribute their streaming content.

Given the increasing integration of P2P streaming in the global content delivery infrastructure, there is a need for an open and standard streaming signaling protocol suite. Almost all existing systems use proprietary protocols. Having multiple proprietary protocols that perform similar functions results in repetitious development efforts for new systems, and the lock-in effects lead to substantial integration difficulties with other players (e.g., CDN). For example, in the enhancement of existing caches and CDN systems to support P2P streaming, proprietary protocols may increase the complexity of interactions 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 the P2P streaming system to solve the above-mentioned problems.

1.2. Requirements Language

The key words "MUST" and "MUST NOT" 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 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 the case of the MB 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 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 a 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 store content provided by the original content servers. Typically, CDN nodes serve content to the users located nearby topologically.

LIVE STREAMING: LIVE STREAMING refers to a scenario where all the audiences receive streaming content for the same ongoing event. It is desired that the lags between the play points of the audiences 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 that are in the 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.

PEER-TO-PEER STREAMING PROTOCOL (PPSP): PPSPs refer to the primary 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 that can be centralized or distributed.

VIDEO ON DEMAND (VoD): VIDEO ON DEMAND refers to a scenario in which different audiences may watch different parts of the same recorded streaming with downloaded content.

SWARM: A SWARM refers to a group of PEERs that 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 described in this section.

3.1. Heterogeneous P2P Traffic and P2P Cache Deployment

ISPs are faced with different P2P streaming applications introducing substantial traffic into their infrastructure, including their backbone and their exchange/interconnection points. P2P caches are used by ISPs 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 that 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

When compared to client/server streaming, P2P streaming is often criticized due to its poorer QoS performance (e.g., longer startup delay, longer seek delay, and channel switch delay). Hybrid CDN/P2P is a good approach to address this problem [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. Similar to what is described in Section 3.1, proprietary P2P protocols introduce complexity and the deployment cost of CDN.

3.3. Extended Applicability in Mobile and Wireless Environments

Mobile and wireless networks, which make considerable use of streaming service, are becoming increasingly important in today's Internet. It's reported that the average volume of video traffic on mobile networks had risen up to 50% in the early part 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 reuse 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 directions, 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 make 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 as 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 size is in kilobytes and is 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 wireline P2P streaming.

Third, for resource-constrained peers, like mobile handsets or set-top boxes (STB), there are multiple systems competing for severely limited resources 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 one of the applications is 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 aims to solve the problems mentioned above by standardizing signaling protocols that support either live or VoD streaming. PPSP supports both centralized and distributed trackers. In distributed trackers, the tracker functionality is distributed in decentralized peers. In this section, the tracker is a logic conception that 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.

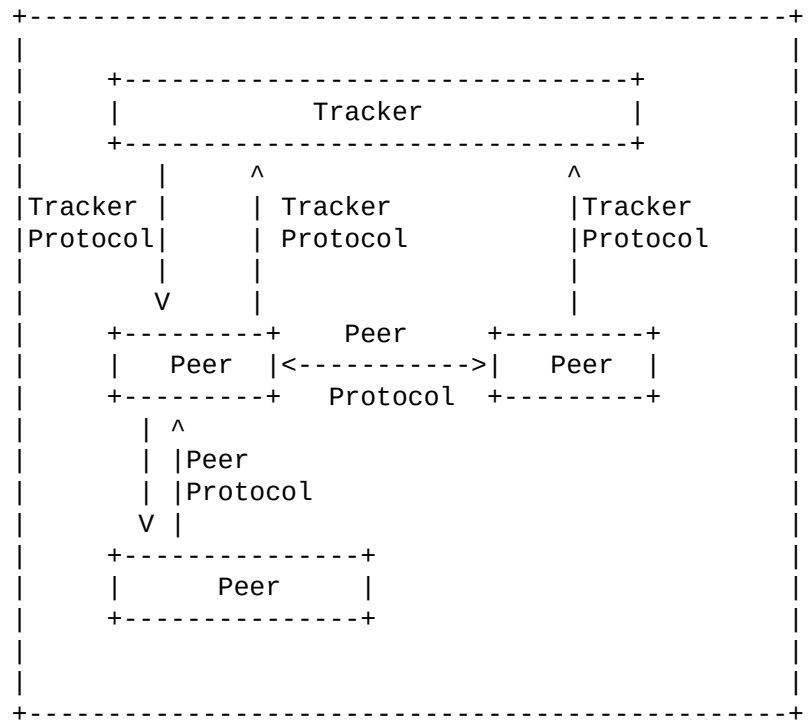

Figure 1: PPSP System Architecture

The PPSP design, in general, needs to solve the following challenges:

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

2) After determining a set of peers, how does a peer make 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., which 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 loosely coupled?

4.1. Tasks and Design Issues of the Tracker Protocol

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

Therefore, the 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 below. 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 addresses, peers behind or not behind NAT, peers with IPV4 or IPV6 addresses, peers with different properties?

3) The tracker protocol must be light weight, since a tracker may need to serve a large number 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 report an optimized peer list to serve particular content? This is related to the status statistic, with which the tracker can be aware of the peer status and content status.

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

4.2. Tasks and Design Issues of the Peer Protocol

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

Therefore, the 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 below. This is a high-level task-list. The detailed requirements on the design of the peer protocol are explicated in Section 6.

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

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

3) How to ensure the peer protocol efficiency? 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.

The 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 a to-do list for the WG; it provides details on 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 between different providers. This is quite similar to the case described in CDNI [RFC6707] [RFC6770].

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

users in the 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 does no attract many users in the USA and Europe, the tracker in A can optimally return a peer list to the user including B's and C's Super-Nodes (SN for short) to provide a better user performance. Furthermore, B's User2 and C's User3 can exchange data (availability) with these local SNs using the peer protocol.

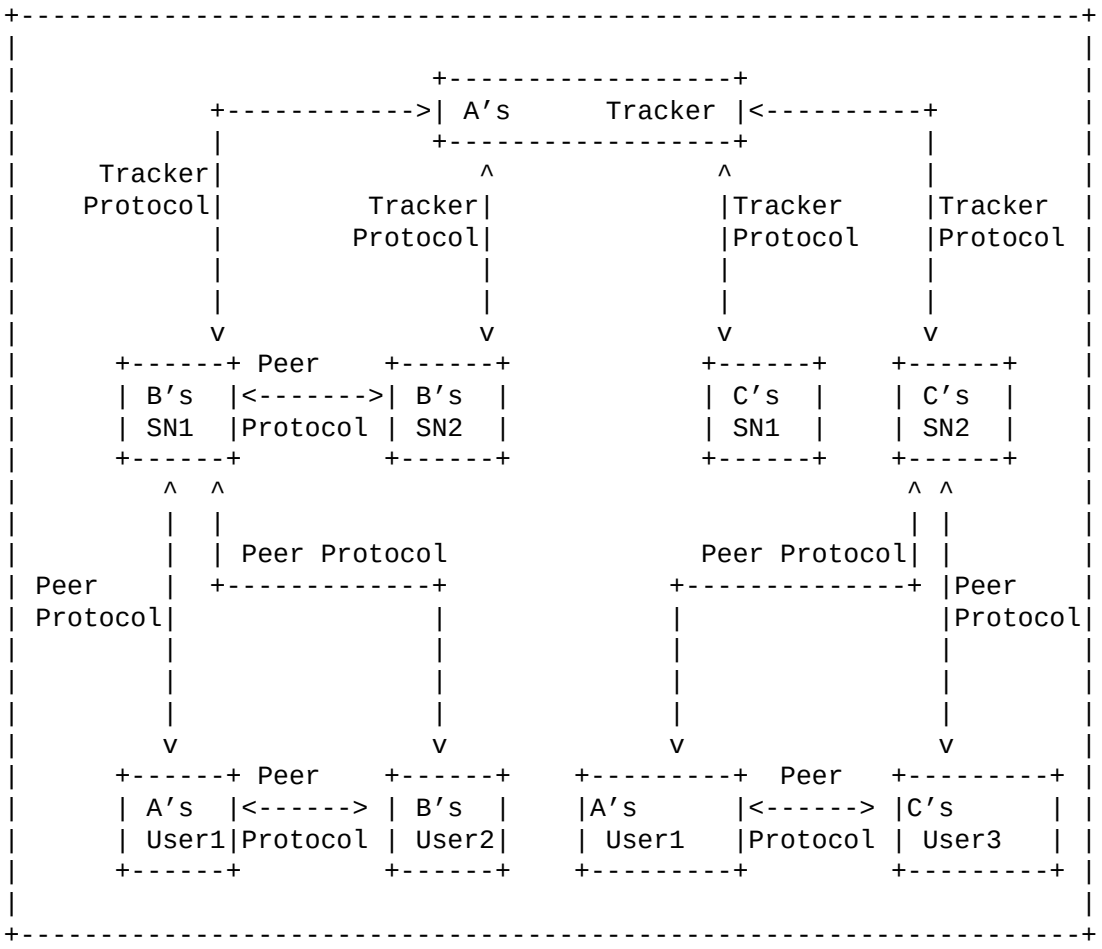

Figure 2: Cooperative Vendors Interaction

5.2. Enabling CDN for P2P VoD Streaming

Figure 3 shows an example 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) using the same PPSP protocols.

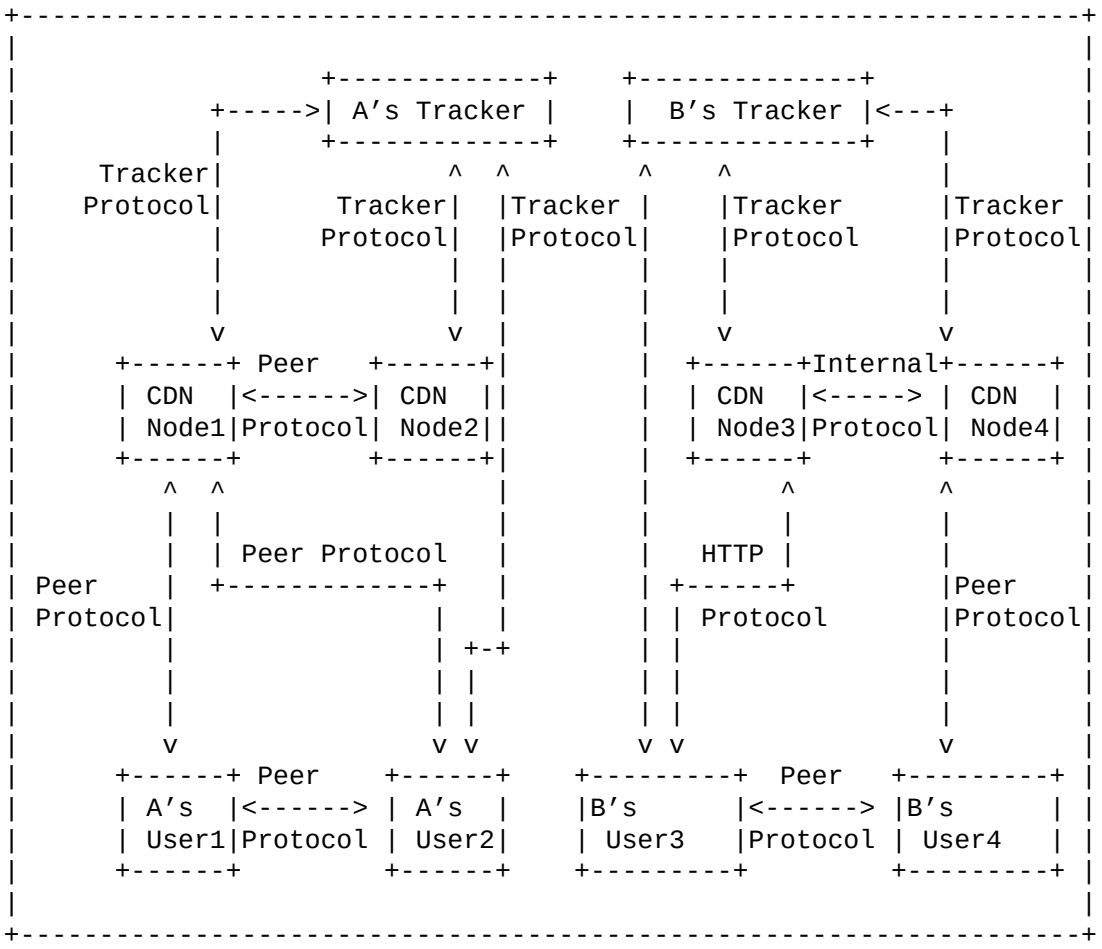

Figure 3: CDN Supporting P2P Streaming

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) that support streaming at a low cost.

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

5.3. Cross-Screen Streaming

In this scenario, PC, STB/TV, and mobile terminals from both fixed and mobile/wireless networks share the streaming content. With PPSP, peers can identify the types of access networks, average load, and 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.

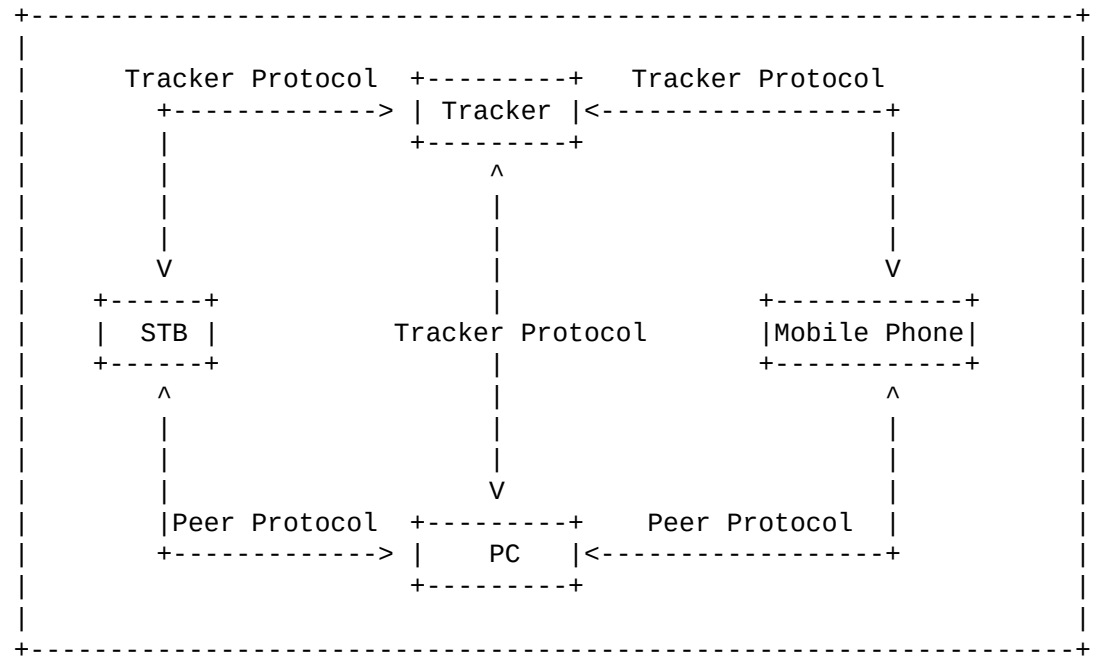

Figure 4: Heterogeneous P2P Streaming with PPSP

Such information will play an important role in 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 lead to a higher load on the base station.

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 enhances the user experience.

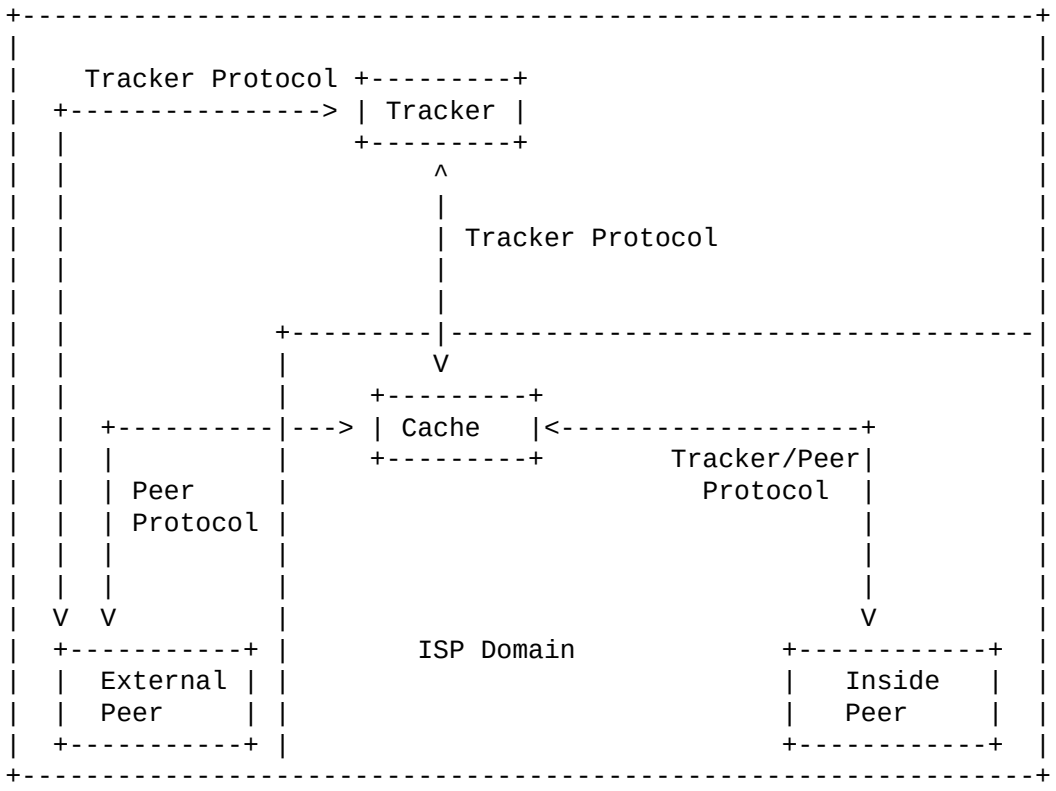

Figure 5: Cache Service Supporting Streaming with PPSP

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

5.5. Proxy Service Supporting P2P Streaming

5.5.1. Home Networking Scenario

For applications where the peer is not colocated 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.

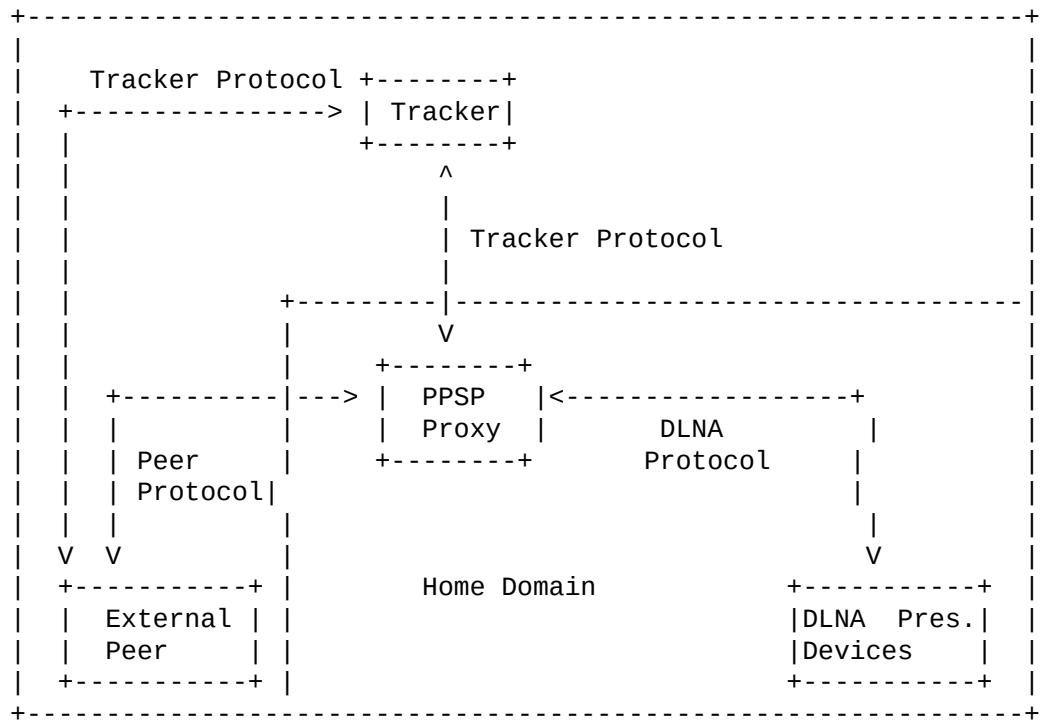

Figure 6: Proxy Service Supporting P2P Streaming

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 Digital Living Network Alliance (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.

5.5.2. Browser-Based HTTP Streaming

P2P Plug-ins are often used in browser-based environments to stream content. With P2P plug-ins, HTTP streaming can be turned into 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).

It's a basic requirement for a peer to be uniquely identified in a P2P streaming system so that other peers or trackers can refer to the peer by ID.

Note that a peer can join multiple swarms with a unique ID or change swarm without changing its 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 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.

6.2. Operational and Management Requirements

This section lists some operational and management requirements based on the checklist presented in Appendix A of [RFC5706].

6.2.1. Operational Considerations

PPSP.OAM.REQ-1: PPSP MUST be sufficiently configurable.

According to basic requirements, when setting up PPSP, a content provider should generate chunk IDs and a swarm ID for each stream of content. An original content server and tracker are configured and set up. The content provider should then publish this information, typically by creating web links.

The configuration should allow the proxy-based and end-client scenarios.

PPSP.OAM.REQ-2: PPSP MUST implement a set of configuration parameters with default values.

PPSP.OAM.REQ-3: PPSP MUST support diagnostic operations.

Mechanisms must be supported by PPSP to verify correct operation. The PPSP tracker should collect the status of the peers including the peer's activity, whether it obtained chunks in time, etc. Such information can be used to monitor the streaming behavior of PPSP.

PPSP.OAM.REQ-4: PPSP MUST facilitate achieving quality acceptable to the streaming application.

There are basic quality requirements for streaming systems. The 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 sporting events where an end-to-end delay of 1 minute or more are not acceptable.

For instance, the tracker and peer protocol can carry quality 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.

PPSP implementations may use techniques such as scalable streaming to handle bandwidth shortages without disrupting playback.

6.2.2. Management Considerations

PPSP.OAM.REQ-5: When management objectives need to be supported in implementations, PPSP MUST support remote management using a standard interface, as well as a basic set of management information.

Due to large-scale peer networks, PPSP tracker service or seeders should remotely collect information from peers and expose the information via a standard interface for management purposes. Peer information can be collected via a PPSP tracker protocol or peer protocol.

The minimum set of management objects should include swarm information such as content characteristics and rate limits; tracking information such as swarm list and log events; and peer information such as peer activity, chunk statistics, and log event.

PPSP.OAM.REQ-6: PPSP MUST support fault monitoring including peer and server health, as well as the streaming behavior of peers.

Peer and server health will at least include node activity and connectivity, especially for peers behind NAT. As mentioned in PPSP.OAM.REQ-4, streaming behavior of the peer can be learned from chunk distribution information.

PPSP.OAM.REQ-7: PPSP MUST support configuration management to define the configuration parameters.

A set of configurable parameters related to chunk generation in the PPSP setup stage can be defined by content providers via a management interface to content servers.

PPSP.OAM.REQ-8: PPSP MUST support performance management with respect to streaming performance based on chunk distribution statistics, network load, and tracker and peer monitoring.

PPSP.OAM.REQ-9: PPSP MUST support security management. See Section 7 of this document.

6.3. PPSP Tracker Protocol Requirements

PPSP.TP.REQ-1: The tracker protocol MUST allow the peer to solicit a peer list in a swarm generated and possibly tailored by the tracker in a query and response manner.

The tracker request message may 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.

The tracker may also generate the peer list with the help of traffic optimization services, e.g., Application-Layer Traffic Optimization [ALTO].

PPSP.TP.REQ-2: The tracker protocol MUST report the peer's activity in the swarm to the tracker.

PPSP.TP.REQ-3: The tracker protocol MUST take the frequency of message exchange and efficient bandwidth use into consideration when communicating chunk availability information.

For example, the chunk availability information between peer and tracker can be presented in a compact method, e.g., to express a sequence of continuous "1" more efficiently.

PPSP.TP.REQ-4: The tracker protocol MUST have a provision for the tracker to authenticate the peer.

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

6.4. 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 the chunk information is validated (e.g., preventing hearsay and DoS attacks).

PPSP.PP.REQ-3: The peer protocol MUST 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, the peer is allowed to communicate 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 an 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, which is verifiable by the receiver.

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

PPSP.PP.REQ-6: The peer protocol MUST report 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 the 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 a 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 be left to the algorithms, rather than protocol concerns.

PPSP.PP.REQ-7: The peer protocol MUST take the frequency of message exchange and efficient bandwidth use into consideration when communicating chunk information.

For example, the chunk availability information between peers can be presented in a compact method.

PPSP.PP.REQ-8: The peer protocol MUST exchange 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.

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 untrusted entities (peers) for service provisioning. For example, malicious peers/trackers may:

- Originate DoS attacks to the trackers by sending a large number of requests with the tracker protocol;

- Originate fake information on behalf of other peers;
- Originate fake information about chunk availability;
- Originate fake reply messages on behalf of the tracker;
- Leak private information about other peers or trackers.

We list some important security requirements for PPSP protocols 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 schemes.

Another aspect is that confidentiality of the streaming content in PPSP needs to be supported. In order to achieve this, PPSP should provide mechanisms to encrypt the data exchange among the peers.

PPSP.SEC.REQ-2: Integrity of the streaming content in PPSP MUST be supported to provide a peer with the possibility of identifying unauthentic content (undesirable modifications 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 forward 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-3: The security mechanisms in PPSP, such as key management and checksum distribution, MUST scale well in P2P streaming systems.

8. Acknowledgements

Thanks to J. Seng, G. Camarillo, R. Yang, C. Schmidt, R. Cruz, Y. Gu, A. Bakker, and S. Previdi for contributing to many sections of this document. Thank you to C. Williams, V. Pascual, and L. Xiao for contributing to the 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, J. Pouwelse, W. Eddy, B. Claise, D. Harrington, J. Arkko, and all the AD reviewers.

9. References

9.1. Normative References

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

[RFC5706] Harrington, D., "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions", RFC 5706, November 2009.

[RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", RFC 6707, September 2012.

[RFC6770] Bertrand, G., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", RFC 6770, November 2012.

9.2. Informative References

[ALTO] Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", Work in Progress, December 2009.

[ByteMobile] Bytemobile, "Mobile Video Traffic Hits Nearly 70% on Certain Networks", February 2012, <http://www.bytemobile.com/news-events/2012/archive_230212.html>.

[CDN-P2P] Xu, D., Kulkarni, S., Rosenberg, C., and H-K. Chai, "Analysis of a CDN-P2P Hybrid Architecture for Cost-Effective Streaming Media Distribution", Multimedia Systems, vol. 11, no. 4, pp. 383-399, 2006.

[Cisco] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2012 - 2017", Visual Networking Index (VNI), <http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/ white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html>.

[DLNA] "DLNA", <http://www.dlna.org>.

[Mobile-Streaming1] Noh, J., Makar, M., and B. Girod, "Streaming To Mobile Users In A Peer-to-Peer Network", MOBIMEDIA , 2009.

[Mobile-Streaming2] Peltotalo, J., Harju, J., Saukkoh, M., Vaatamoinen, L., Bouazizi, I., Curcio, I., and J. van Gassel, "A Real-Time Peer-to-Peer Streaming System for Mobile Networking Environment", Proceedings of the INFOCOM and Workshop on Mobile Video Delivery (MoVID '09), 2009.

[P2PYoutube] "Youtube Extension-Opera Add-Ons", Opera Software, <https://addons.opera.com/en/extensions/details/p2p-youtube/>.

[VoD] Huang, Y., Fu, T., Chiu, D-M., Lui, J., and C. Huang, "Challenges, Design and Analysis of a Large-Scale P2P-VoD System", SIGCOMM , 2008.

Authors' Addresses

Yunfei Zhang
Coolpad

EMail: hishigh@gmail.com

Ning Zong
Huawei Technologies

EMail: zongning@huawei.com