PPSP
INTERNET-DRAFT
Intended Status: Standards Track
Expires: April 25, 2013

Rachel Huang
Huawei
Rui S. Cruz
Mario S. Nunes
IST/INESC-ID/INOV
Joao P. Taveira
IST/INOV

October 22, 2012

PPSP Tracker Protocol--Extended Protocol (PPSP-TP/1.1) draft-huang-ppsp-extended-tracker-protocol-01

Abstract

This document specifies the extended Peer-to-Peer Streaming Protocol - Tracker Protocol (PPSP-TP/1.1), a new extension protocol complementing the basic core messages and usages specified in PPSP=TP/1.0 for the exchange of meta information between trackers and peers, such as initial offer/request of participation in multimedia content streaming, content information, peer lists and reports of activity and status. It extends PPSP-TP/1.0 to include new optional messages providing granular controls and usages in the communications between peer and tracker. The extension protocol is retro-compatible with PPSP-TP/1.0.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Table of Contents

1. Introduction

The PPSP Tracker Protocol is one of the Peer-to-Peer Streaming Protocol which specifies standard format/encoding of information and messages between PPSP peers and PPSP trackers. Based on the requirements defined in [I-D.ietf-ppsp-problem-statement], PPSP-TP/1.0 specified in [I-D.cruz-ppsp-base-tracker-protocol] has provided the basic core messages to be exchanged between trackers and peers in order to carry out some fundamental operations. They are mandatory messages covering most basic use cases and MUST be implemented in all PPSP-based streaming systems.

This document specifies PPSP-TP/1.1 to complement the basic core messages and usages specified in [I-D.cruz-ppsp-base-tracker-protocol]. It extends PPSP-TP/1.0 to include some new optional messages for providing granular controls and usages in some dedicated scenarios. The extension protocol is retro-compatible with PPSP-TP/1.0. For a peer, it can implement either PPSP-TP/1.0 or PPSP-TP/1.1. For a tracker, it is recommended to implement PPSP-TP/1.1, which is also able to deal with the requests of PPSP-TP/1.0.

2. Terminology

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].

This draft uses terms defined in [I-D.ietf-ppsp-problem-statement] and [I-D.cruz-ppsp-base-tracker-protocol].

3. PPSP-TP/1.1 Overview

3.1. Request Messages

3.1.1. Enhanced Request Messages

In this section, the request messages specified in PPSP-TP/1.0 are extended to enable granular control and optional information delivery.

3.1.1.1 CONNECT

Besides the semantics defined in PPSP-TP/1.0 specification, the enhanced CONNECT Request message is also used when a peer registers in the tracker without simultaneously requesting additional actions to be carried upon successful registration. In such a case, the tracker records the Peer-ID, connect-time (referenced to the absolute time), optional peer IP addresses and link status (if present in the

request), and waits for further requests from the peer.

This message is also extended to allow a peer participating in some swarms as LEECH and simultaneously joining other swarms as SEED when it registers in the tracker. For example, a peer is a seeder of several certain contents that it generated, but it is also a consumer of a streaming program. For some reason the connectivity failed and it wants to reconnect from the point it was before. Then the enhanced CONNECT request message could satisfy this kind of usage.

3.1.1.2 STAT_REPORT

The STAT_REPORT Request message is extended to allow the exchange of content data information, like chunkmaps, between an active peer and a tracker. The information can be used by a tracker as a qualification to select appropriate peer lists when peers request to the tracker for the peer lists of some specific contents. An example of a STAT_REPORT for multiple properties is illustrated in subsection 4.5.

3.1.2. New Request Messages

Three new messages, listed in Table 1, are introduced in this section to extend those specified in PPSP-TP/1.0 [I-D.cruz-ppsp-base-tracker-protocol].

PPSP Tracker Extension Req. Messages
DISCONNECT
JOIN
FIND

Table 1: Extended Request Messages

3.1.2.1. JOIN

The JOIN Request message is used for a peer to notify the tracker that it wishes to participate in a swarm. The tracker records the content availability, i.e., adds the peer to the peers list for the swarm. On receiving a JOIN message, the tracker first checks the PeerMode type and then decides the next step (more details are referred in section 4.3).

A use case of this request message is showed in Figure 1. In this

case, the seeder of a certain provider has already registered in a tracker through the CONNECT message. When it receives some "Start" activation signal for a program x, it could send the JOIN message to the connected tracker to join the swarm. When it receives another signal for joining another program y, it can send the JOIN message to the tracker to request for joining the other swarm, but remaining sharing the content for the first one.

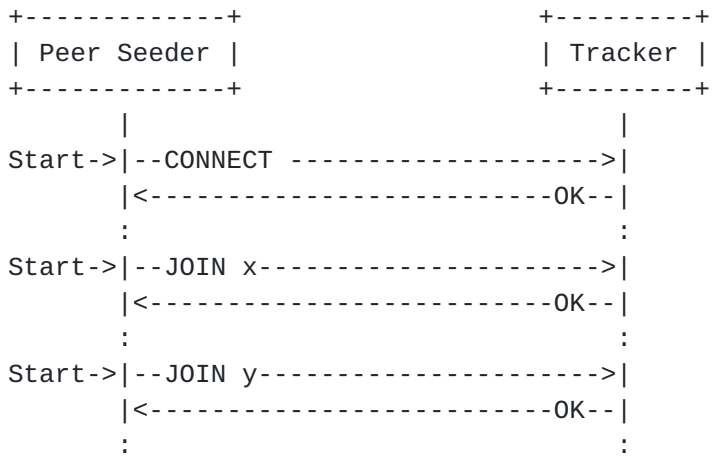

Figure 1: An Example of JOIN Request Message Usages

3.1.2.2. DISCONNECT

The DISCONNECT Request message is used when the peer intends to no longer participate in a specific swarm, or in all swarms. When receiving the message from a peer, the tracker deletes the corresponding activity records related to the peer (including its status and all content status for the corresponding swarms).

Continuing the example in 3.1.1, when the newly joined program is no longer available, a "DISCONNECT z" message is sent by the seeder to indicate that it leaves swarm z. Note that this activity does not affect other swarms (swarm x, y). The seeder still maintains active all the other swarms it participates in.

Another use case is that this request message can be used to stop peer participation in all swarms and de-register from the system. In such a case, DISCONNECT message will have the same effect of timer expiring (from the tracker perspective), but providing a graceful disconnect from the system.

3.1.2.3. FIND

The FIND Request message allows a peer to request the tracker for the peer list of a swarm. The request can include specific content

scopes, either media content representations or specific chunks of a media representation in a swarm. On receiving a FIND message, the tracker finds the peers, listed in content status of the specified swarm that can satisfy the requesting peer's requirements, returning the list to the requesting peer. To create the peer list, the tracker may take peer status, capabilities and peers priority into consideration. Peer priority may be determined by network topology preference, operator policy preference, etc.

This message can be used in scenarios when peers want to obtain the updated peer lists from the tracker. For example, when a peer acting as leech has joined a swarm, and after a while part of the content is not available in the previously known peers, it can send a FIND request message to the tracker to update the peer lists which could satisfy the peer's requirement.

Another scenario could be found when a peer has changed its primary network attachment. One example is that a peer with a LAN and a WiFi interface which are going through different routers. The peer is using some PPSPO-based P2P application which can keep working when the peer switches from the LAN to the WiFi (for example, unplugging the Ethernet cable, the P2P connection can recover automatically). In this case, the peer can send a FIND Request message to the tracker for the updated peer lists. And the response replied by the tracker to the FIND Request message should include the requesting peer transport address as well as the required peer list.

An example of usage of the enhanced request messages in PPSP-TP/1.1 is the illustrated in Figure 2.

In that figure a peers starts by connecting to the system (using the semantics of PPSP-TP/1.1 Registration CONNECT) after which JOINs a specific swarm (swarm_a) in SEED mode.

While active the peer periodically updates the tracker using STAT_REPORT messages. Later, the peer JOINs another swarm (swarm_b) but in LEECH mode, i.e., the end-user intended to watch that content while still sharing the first one. During the stream the peer requests an updated list of peers in that swarm to the tracker.

When finished streaming the second content the peer DISCONNECTs from the corresponding swarm (swarm_b) while still sharing the first content (swarm_a).

Later the peer DISCONNECTs from the system.

Figure 2: Example of a session for a PPSP-TP extended version.

3.2. Extended Tracker Transaction State Machine

The tracker state machine has been introduced in PPSP-TP/1.0 [I-D.cruz-ppsp-base-tracker-protocol]. Every tracker MUST keep a tracker state machine in which the state transitions are triggered by peer registrations. In addition to the tracker state machine, a transaction state machine for each registered Peer-ID is also specified. In this specification, as some additional granularity messages have been introduced, an extended "per-Peer-ID" transaction state machine (Figure 2) is specified to provide more functionality and detailed control to the tracker protocol. PPSP-TP/1.1 MUST include both "per-Peer-ID" transaction state machines to remain compatible with PPSP-TP/1.0. A Tracker implementing PPSP-TP/1.1 could instantiate corresponding "per-Peer-ID" transaction state machine based on the version of the CONNECT request message received from the peer.

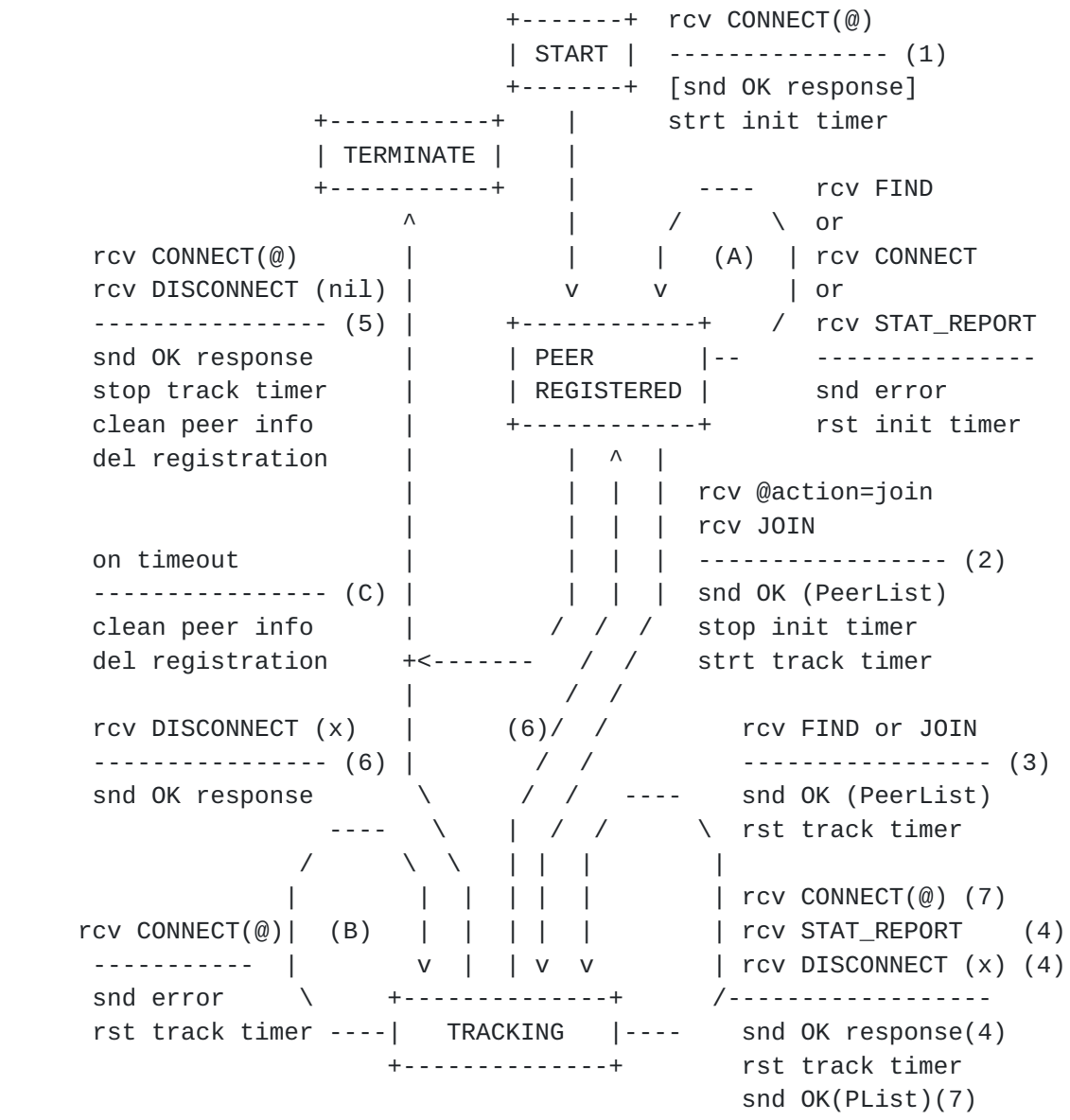

Figure 3: Extended Per-Peer-ID Transaction State Machine

The state diagram in Figure 3 illustrates the complete state changes together with the causing events and resulting actions when implementing basic tracker protocol with extended protocol. Note that specific error conditions are not shown in the state diagram.

3.2.1. Normal Operation

On normal operation the extended process consists of the following steps:

1) When an enhanced CONNECT message is received from a peer, if

successfully authenticated and validated, the tracker registers the Peer-ID and associated information (IP addresses). In case the CONNECT request also includes Swarm @action requests (using the same semantics of PPSP-TP/1.0), it moves to PEER REGISTERED state carrying the Swarm @action to be performed. In case the CONNECT request does not contain any Swarm @action requests, it sends the response of successful registration to peer, starts the "init timer" and moves to PEER REGISTERED state.

2) While PEER REGISTERED, when a JOIN message or swarm @action="JOIN" request is received with valid swarm information, the tracker stops the "init timer", starts the "track timer" and sends the response of successful join to the peer. The response MAY contain the appropriate list of peers in the swarm, depending on PeerMode. A successful request in this state starts the TRACKING state associated with the peer-ID for the requested swarm.

3) While TRACKING, a JOIN or FIND message received with valid swarm information from the peer resets the "track timer" and is responded with a successful condition, either for the JOIN to (an additional) swarm or for including the appropriate list of peers for the scope in the FIND request.

4) While TRACKING, a DISCONNECT(x) message received from the peer, containing a valid x=Swarm-ID resets the "track timer" and is responded with a successful condition. The tracker cleans the information associated with the participation of the Peer-ID in the specified swarm(s).

   In TRACKING state a STAT_REPORT message received from the peer resets the "track timer" and is responded with a successful condition. The STAT_REPORT message MAY contain information related with Swarm-IDs to which the peer is joined.

5) From either PEER REGISTERED or TRACKING states a DISCONNECT(x) message received from the peer, where x=nil, the tracker stops the "track timer", cleans the information associated with the participation of the Peer-ID in the the swarm(s) joined, responds with a successful condition, deletes the registration of the Peer-ID and transitions to TERMINATED state for that Peer-ID. The same operations are performed if a CONNECT message including valid Swarm @action="LEAVE" requests (using the same semantics of PPSP-TP/1.0) is received while in TRACKING state (valid actions in Table 9).

6) From TRACKING state a DISCONNECT(x) message received from the peer, where x=ALL or x=Swarm-ID is the last swarm, the tracker stops the "track timer", cleans the information associated with

the participation of the Peer-ID in the the swarm(s) joined, responds with a successful condition and transitions to PEER REGISTERED state.

7) While TRACKING, a CONNECT message received from the peer including a valid pair of Swarm @action requests (one being @action="LEAVE" and the other being @action="JOIN", using the same semantics of PPSP-TP/1.0), resets the "track timer" and is responded with a successful condition (valid actions in Table 9). For the @action="JOIN" the response to the request MAY include the appropriate list of peers.

3.2.2. Error Conditions

A) At the PEER REGISTERED state (while the "init timer" has not expired) receiving FIND, CONNECT or STAT_REPORT messages from the peer is considered as an error condition. The tracker responds with error code 403 Forbidden, and resets the "init timer" one last time.

B) At the TRACKING state (while the "track timer" has not expired) receiving an enhanced CONNECT message or swarm @action="JOIN" request from the peer is considered an error condition. The tracker responds with error code 403 Forbidden, and resets the "track timer".

NOTE: This situation may correspond to a malfunction at the peer or to malicious conditions. A preventive measure would be to reset the "track timer" one last time and if no valid message is received proceed to TERMINATE state for the Peer-ID by de-registering the peer and cleaning all peer information.

C) Without receiving messages from the peer, either from PEER REGISTERED sate (controlled by init timer) or TRACKING state (controlled by track timer), on timeout the tracker cleans all the information associated with the Peer-ID in all swarms it was joined, deletes the registration, and transitions to TERMINATE state for that Peer-ID. The same action is taken if no valid message is received at the PEER REGISTERED state after the last "init timer" expires.

3.3. Extended Request/Response Syntax and Format

The architecture of PPSP-TP/1.1 is consistent with the architecture of the the basic tracker protocol specified in PPSP- TP/1.0 [I-D.cruz-ppsp-base-tracker-protocol]. It still uses the layered architecture of PPSP-TP/1.0. Besides that, the message syntax is identical with that used by PPSP-TP/1.0. Note that PPSP-TP/1.1 is

compatible with all the syntax and formats of PPSP-TP/1.0. The difference is that some request/response syntax has been extended in PPSP-TP/1.1 to contain the new optional messages. This section extends the generic format of Request to satisfy the extended request messages. The extended generic format of a Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request></Request>
  <TransactionID></TransactionID>
  <PeerID></PeerID>
  <SwarmID></SwarmID>
  <PeerNum></PeerNum>
  <PeerGroup></PeerGroup>
  <ContentGroup></ContentGroup>
  <StatisticsGroup></StatisticsGroup>
</PPSPTrackerProtocol>
```

The generic format of a Response is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response></Response>
  <TransactionID></TransactionID>
  <PeerGroup></PeerGroup>
</PPSPTrackerProtocol>
```

The version of PPSPTrackerProtocol has been changed to 1.1 which indicates that the peer is using extended tracker protocol (PPSP-TP/1.1).

Besides that, syntax of some elements has been extended for three extended request messages.

The SwarmID element MUST be present in FIND, JOIN and DISCONNECT requests.

The PeerNum element MAY be present in JOIN and FIND requests and MAY contain the attribute @abilityNAT to inform the tracker on the preferred type of peers, in what concerns their NAT traversal situation, to be returned in a peer list.

The PeerGroup element MAY be present in CONNECT requests and

responses and MAY be present in responses to JOIN and FIND requests if the corresponding responses return information about peers.

One element "ContentGroup" is added to the format of Request. It MAY be present in requests referencing content, i.e., JOIN and FIND, if the request includes a content scope.

The extended semantics of the attributes and elements within a PPSPTrackerProtocol root element is described in subsection 3.3.1.

3.3.1. Extended Semantics of PPSPTrackerProtocol Elements

The extended semantics for PPSP-TP/1.1 are described bellow.

Element Name or Attribute Name	Use	Description
PPSPTrackerProtocol	1	The root element.
@version	M	Provides the version of PPSP-TP.
Request	0...1	Provides the request method and MUST be present in Request.
Response	0...1	Provides the response method and MUST be present in Response.
TransactionID	M	Root transaction Identification.
Result	0...N	Result of @action MUST be present in Responses.
@transactionID	CM	Identifier of the @action.
PeerID	0...1	Peer Identification. MUST be present in Request.
SwarmID	0...N	Swarm Identification. MUST be present in Requests.
@action	CM	Must be set to JOIN or LEAVE.
@peerMode	CM	Mode of Peer participation in the swarm, "LEECH" or "SEED".
@transactionID	CM	Identifier for the @action.
PeerNUM	0...1	Maximum peers to be received with capabilities indicated.
@abilityNAT	CM	Type of NAT traversal peers, as "No-NAT","STUN","TURN" or "PROXY"
@concurrentLinks	CM	Concurrent connectivity level of peers, "HIGH", "LOW" or "NORMAL"
@onlineTime	CM	Availability or online duration of peers, "HIGH" or "NORMAL"
@uploadBWlevel	CM	Upload bandwidth capability of peers, "HIGH" or "NORMAL"
PeerGroup	0...1	Information on peers (Table 3)
ContentGroup	0...1	Information on content (Table 4)
StatisticsGroup	0...1	Statistic data (Table 5)

Legend:
Use for attributes: M=Mandatory, OP=Optional, CM=Conditionally Mandatory
Use for elements: minOccurs...maxOccurs (N=unbounded)
Elements are represented by their name (case-sensitive)
Attribute names (case-sensitive) are preceded with an @

Table 2: Semantics of the Extended PPSPTrackerProtocol.

The semantics of PeerGroup element is almost identical with that in PPSP-TP/1.0. It is listed below for convenience of reading. The implementation of this element has been extended in this specification, which is described in 4.3.

Element Name or Attribute Name	Use	Description
PeerGroup	0...1	Contains description of peers.
PeerInfo	1...N	Provides information on a peer.
@swarmID	0...1	Swarm Identification.
PeerID	0...1	Peer Identification. MAY be present in responses.
PeerAddress	0...N	IP Address information.
@addrType	M	Type of IP address, which can be "ipv4" or "ipv6"
@priority	CM	The priority of this interface. Used for NAT traversal.
@type	CM	Describes the address for NAT traversal, which can be "HOST" "REFLEXIVE" or "PROXY".
@connection	OP	Access type ("3G", "ADSL", etc.)
@asn	OP	Autonomous System number.
@ip	M	IP address value.
@port	M	IP service port value.
@peerProtocol	OP	PPSP Peer Protocol supported.

Legend:
Use for attributes: M=Mandatory, OP=Optional, CM=Conditionally Mandatory
Use for elements: minOccurs...maxOccurs (N=unbounded)
Elements are represented by their name (case-sensitive)
Attribute names (case-sensitive) are preceded with an @

Table 3: Semantics of PeerGroup.

Table 4 describes the semantics of StatisticsGroup element.

Element Name or Attribute Name	Use	Description
StatisticsGroup	0...1	Provides statistic data on peer and content.
Stat	1...N	Groups statistics property data.
@property	M	The property to be reported. Property values in Table 5.
SwarmID	0...1	Swarm Identification.
UploadedBytes	0...1	Bytes sent to swarm.
DownloadedBytes	0...1	Bytes received from swarm.
AvailBandwidth	0...1	Upstream Bandwidth available.
Representation	0...N	Describes a component of content.
@id	CM	Unique identifier for this Representation.
SegmentInfo	1...N	Provides segment information by segment range. The chunkmap can be encoded in Base64 [RFC4648].
@startIndex	CM	The index of the first media segment in the chunkmap report for this Representation.
@endIndex	CM	The index of the last media segment in the chunkmap report for this Representation.
@chunkmapSize	CM	Size of chunkmap reported.

Legend:
Use for attributes: M=Mandatory, OP=Optional, CM=Conditionally Mandatory
Use for elements: minOccurs...maxOccurs (N=unbounded)
Elements are represented by their name (case-sensitive)
Attribute names (case-sensitive) are preceded with an @

Table 4: Semantics of StatisticsGroup.

@property	Description
StreamStatistics	Stream statistic values per SwarmID
ContentMap	Reports map of chunks the peer has per Representation of the content

Table 5: StatisticsGroup Default Stat @property Values.

ContentGroup is a new element extended in this specification. The semantics of this element is described in Table 6.

Element Name or Attribute Name	Use	Description
ContentGroup	0...1	Provides information on content.
Representation	1...N	Describes a component of content.
@id	M	Unique identifier for this Representation.
SegmentInfo	1	Provides segment information.
@startIndex	M	The index of the first media segment in the request scope for this Representation.
@endIndex	OP	The index of the last media segment in the request scope for this Representation.
Legend:		
Use for attributes: M=Mandatory, OP=Optional, CM=Conditionally Mandatory		
Use for elements: minOccurs...maxOccurs (N=unbounded)		
Elements are represented by their name (case-sensitive)		
Attribute names (case-sensitive) are preceded with an @		

Table 6: Semantics of ContentGroup

The Representation element describes a component of a content identified by its attribute @id in the MPD. This element MAY be present for each component desired in the scope of the JOIN or FIND request. The scope of each Representation is indicated in the SegmentInfo element by the attribute @startIndex and, optionally, @endIndex.

The peer may use this information in JOIN or FIND requests, for example, to join a swarm starting from a specific point (as is the case of a live program, by specifying the adequate @startIndex) and/or find adequate peers in the swarm for that content scope.

An example of on-demand usage is the case of an end-user that previously watched a content with a certain audio language, then interrupted for a while (having disconnected) and later continued by re-joining from that point onwards but selecting a different available audio language. In this case the JOIN request would specify the required Representations and the @startIndex for each, i.e., all the adequate video components and the selected audio

component. An example is illustrated in subsection 4.3.

3.3.2. Extended Request/Response Element in Request Messages

Table 7 specifies the valid string representations for the requests in PPSP-TP/1.1. These values MUST be treated as case-sensitive.

Extended XML Request Methods String Values
CONNECT
DISCONNECT
JOIN
FIND
STAT_REPORT

Table 7: Extended Valid Strings for Request Element of Requests.

The response elements in response messages of PPSP-TP/1.1 are identical with those specified in PPSP-TP/1.0, which can be found in subsection 7.2.3 of [I-D.cruz-ppsp-base-tracker-protocol].

3.4. Compatibility with PPSP-TP/1.0

Peers and trackers may implement different versions of PPSP tracker protocol. Table 8 illustrate the different conditions when peer and tracker have different PPSP-TP versions.

Peer Version	Tracker Version	Request Version	Response Version
1.0	1.0	1.0	1.0
1.0	1.1	1.0	1.0
1.1	1.0	1.1	invalid
		1.0	1.0
1.1	1.1	1.1	1.1

Table 8: The Implementations of Different PPSP-TP Versions

Just as Table 8 shows, when both peer and tracker implement PPSP-

TP/1.0, all the requests and responses are processed based on semantics and schema of PPSP-TP/1.0.

When peer implements PPSP-TP/1.0 and tracker implements PPSP-TP/1.1, the peer sends CONNECT 1.0 and tracker responds with the semantics and version of PPSP-TP/1.0.

When peer implements PPSP-TP/1.1 and tracker implements PPSP-TP/1.0, the peer sends CONNECT 1.1 and tracker will respond with 400 (Bad request, with reason-phrase "PPSP version 1.0") to indicate that the peer has sent the invalid version of the request. After the peer receives the response with 400, it MUST send CONNECT 1.0 to be retro-compatible with the tracker implementing PPSP-TP/1.0.

When both peer and tracker implement PPSP- TP/1.1, peer may use an advanced CONNECT 1.1 with the semantics, rules and restrictions similar to the PPSP-TP/1.0 to switch channels, re-connect after a failure or use an advanced CONNECT 1.1 to register first and then wait for all other messages to perform other steps, such as JOIN, FIND, etc.

4. Request/Response Processing

4.1. Enhanced CONNECT Request

This method is used when a peer registers to the system. The tracker records the Peer-ID, connect-time, IP addresses and link status.

The peer MUST properly form the XML message-body, set the Request method to CONNECT, generate and set the TransactionID, and set the PeerID with the identifier of the peer. The peer MAY optionally include the IP addresses of its network interfaces in the CONNECT message, and so, the element PeerInfo MAY contain multiple PeerAddress child elements with attributes @addrType, @ip, @port and @peerProtocol, and optionally @priority and @type (if PPSP-ICE NAT traversal techniques are used) corresponding to each of the network interfaces of the peer.

The enhanced CONNECT Request has two variants: a registration CONNECT Request and a fast CONNECT request (retro-compatible with PPSP-TP/1.0).

4.1.1 Registration CONNECT Request

An example of the message-body of a registration CONNECT Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>CONNECT</Request>
  <PeerID>656164657221</PeerID>
  <TransactionID>12345</TransactionID>
  <PeerGroup>
    <PeerInfo>
      <PeerAddress addrType="ipv4" ip="192.0.2.1" port="80"
                   priority="1"
                   type="HOST"
                   peerProtocol="PPSP-PP" />
      <PeerAddress addrType="ipv6" ip="2001:db8::1" port="80"
                   priority="2
                   type="HOST"
                   peerProtocol="PPSP-PP" />
    </PeerInfo>
  </PeerGroup>
</PPSPTrackerProtocol>
```

When receiving a well-formed CONNECT Request message, the tracker will first processes the peer authentication information (provided as Authorization scheme and token in the HTTP message) to check whether it is valid and that it can connect to the service, and then proceed to register the peer in the service. In case of success a Response message with a corresponding response value of SUCCESSFUL will be generated.

The response to the registration CONNECT request MUST have the same TransactionID value as the request. An example of a Response message for the CONNECT Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>12345</TransactionID>
  <PeerGroup>
    <PeerInfo>
      <PeerAddress addrType="ipv4" ip="198.51.100.1" port="80"
                   priority="1"
                   type="REFLEXIVE"
                   connection="ADSL"
                   asn="64496" />
    </PeerInfo>
  </PeerGroup>
</PPSPTrackerProtocol>
```

The Response MUST include a PeerGroup with PeerInfo data that includes the peer public IP address. If STUN-like function is enabled in the tracker, the PeerAddress includes the attribute @type with a value of REFLEXIVE, corresponding to the transport address "candidate" of the peer.

The tracker MAY also include the attribute @asn with network location information of the transport address, corresponding to the Autonomous System Number of the access network provider.

4.1.2 Fast CONNECT Request

Fast CONNECT request is retro-compatible with the CONNECT request message defined in PPSP-TP/1.0 specification. An example of the message-body of the fast CONNECT Request is the following. This example is identical with PPSP-TP/1.0.

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>CONNECT</Request>
  <PeerID>656164657220</PeerID>
  <SwarmID action="JOIN" peerMode="SEED"
           transactionID="12345.1">1111</SwarmID>
  <SwarmID action="JOIN" peerMode="SEED"
           transactionID="12345.2">2222</SwarmID>
  <TransactionID>12345.0</TransactionID>
</PPSPTrackerProtocol>
```

Another example of fast CONNECT request usage is showed in below:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>CONNECT</Request>
  <PeerID>656164657220</PeerID>
  <SwarmID action="JOIN" peerMode="SEED"
           transactionID="12345.1">1111</SwarmID>
  <SwarmID action="JOIN" peerMode="SEED"
           transactionID="12345.2">2222</SwarmID>
  <SwarmID action="JOIN" peerMode="LEECH"
           transactionID="12345.3">3333</SwarmID>
  <TransactionID>12345.0</TransactionID>
</PPSPTrackerProtocol>
```

In this last example, the peer wants to re-join and participate in swarm 3333 to watch the program as a leecher, while sharing as seeder swarm 1111 and swarm 2222 with other peers.

SwarmID Elements	@peerMode value	@action value	Initial State	Final State	Request validity
1	LEECH	JOIN	START	TRACKING	Valid
1	LEECH	LEAVE	START	TERMINATE	Invalid
1	LEECH	LEAVE	TRACKING	TERMINATE	Valid
1	LEECH	JOIN	START	TERMINATE	Invalid
1	LEECH	LEAVE			
1	LEECH	JOIN	TRACKING	TRACKING	Valid
1	LEECH	LEAVE			
x	LEECH	JOIN	START	TRACKING	Valid
y	SEED	JOIN			
N	SEED	JOIN	START	TRACKING	Valid
N	SEED	JOIN	TRACKING	TERMINATE	Invalid
N	SEED	LEAVE	TRACKING	TERMINATE	Valid

Table 9: Validity of SwarmID @action combinations in CONNECT Request

It is also possible to simultaneously participate in multiple swarms

as LEECH, for example in situations when the peer has stopped watching several programs at a certain moment of the timeline (it may not have all the chunks of the contents) but may wish to continue sharing these programs with other peers. As such, a fast CONNECT request message would include some @action="JOIN" in LEECH mode and some in SEED mode.

In Table 9, the valid sets of SwarmID @action combinations is extended for the fast CONNECT Request logic in PPSP-TP/1.1 (referring to the "per-peer-ID" transaction state machine). The allowed number of SwarmID elements in a request is indicated, where x, y and N take values greater or equal to 1. When N is indicated, it means that if N requests are @action="JOIN" for the named swarms, the same N requests MUST be used for the corresponding @action="LEAVE" of the same swarms when using the fast CONNECT Request logic.

The response MUST have the same TransactionID values as the corresponding request and actions.

The Response to the Fast CONNECT Request MUST include a PeerGroup with PeerInfo data of the requesting peer public IP address. If STUN-like function is enabled in the tracker, the PeerAddress includes the attribute @type with a value of REFLEXIVE, corresponding to the transport address "candidate" of the peer. The PeerGroup MAY also include PeerInfo data corresponding to the Peer-IDs and public IP addresses of the selected active peers in the requested swarm.

The tracker MAY also include the attribute @asn with network location information of the transport address, corresponding to the Autonomous System Number of the access network provider.

In case the @peerMode is SEED, the tracker responds with a SUCCESSFUL response and enters the peer information into the corresponding swarm activity.

In case the @peerMode is LEECH the tracker will search and select an appropriate list of peers satisfying the conditions set by the requesting peer. The peer list returned MUST contain the Peer-IDs and the corresponding IP Addresses. To create the peer list, the tracker may take peer status and network location information into consideration, to express network topology preferences or Operators' policy preferences, with regard to the possibility of connecting with other IETF efforts such as ALTO [I.D.ietf-alto- protocol].

An example of a Response message for the CONNECT Request from a peer leecher is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>
    <Result transactionID="12345.0">200 OK</Result>
    <Result transactionID="12345.1">200 OK</Result>
    <Result transactionID="12345.3">200 OK</Result>
  </TransactionID>
  <PeerGroup>
    <PeerInfo>
      <PeerID>656164657221</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.1" port="80"
                   priority="1"

                   type="REFLEXIVE"
                   connection="3G"
                   asn="64496" />
    </PeerInfo>
    <PeerInfo swarmID="2222">
      <PeerID>956264622298</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.22" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
    <PeerInfo swarmID="2222">
      <PeerID>3332001256741</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.201" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
  </PeerGroup>
</PPSPTrackerProtocol>
```

4.2. DISCONNECT Request

This method is used when the peer intends to leave one or multiple specific swarms, or the system, and no longer participate.

The tracker SHOULD delete the corresponding activity records related with the peer in the corresponding swarms (including its status and all content status).

The peer MUST properly form the XML message-body, set the Request method to DISCONNECT, set the PeerID with the identifier of the peer, randomly generate and set the TransactionID and include the SwarmID information.

The element SwarmID MUST be present with cardinality 1 to N, each

containing no child elements. The SwarmID value MUST be either specific Swarm-ID the peer had previously joined, the value "ALL" to designate all joined swarms, or the value "nil" to completely disconnect from the system.

An example of the message-body of a DISCONNECT Request for the peer leaving all joined swarms is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>DISCONNECT</Request>
  <PeerID>656164657221</PeerID>
  <SwarmID>ALL</SwarmID>

  <TransactionID>12345</TransactionID>
</PPSPTrackerProtocol>
```

An example of the message-body of a DISCONNECT Request for a peer seeder leaving several specific swarms is the following:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <PPSPTrackerProtocol xmlns="TBD"
                         schemaLocation="TBD"
                         version="1.1">
      <Request>DISCONNECT</Request>
      <PeerID>656164657220</PeerID>
      <SwarmID transactionID="12345.1">1111</SwarmID>
      <SwarmID transactionID="12345.2">2222</SwarmID>
      <TransactionID>12345.0</TransactionID>
    </PPSPTrackerProtocol>
```

In case of success a Response message with a corresponding response value of SUCCESSFUL will be generated. The response MUST have the same TransactionID value as the request.

Upon receiving a DISCONNECT message, the tracker cleans the information associated with the participation of the Peer-ID in the specified swarms (or in all swarms).

An example of a Response message for the first DISCONNECT Request is the following:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <PPSPTrackerProtocol xmlns="TBD"
                         schemaLocation="TBD"
                         version="1.1">
      <Response>SUCCESSFUL</Response>
      <TransactionID>12345</TransactionID>
    <PPSPTrackerProtocol>
```

An example of a Response message for the second DISCONNECT Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>
    <TransactionID>12345</TransactionID>
    <Result transactionID="12345.0">200 OK</Result>
    <Result transactionID="12345.1">200 OK</Result>
    <Result transactionID="12345.2">200 OK</Result>
  <TransactionID>
<PPSPTrackerProtocol>
```

If the scope of SwarmID in the DISCONNECT request is "nil", the tracker will also delete the registration of the Peer-ID.

4.3. JOIN Request

This method is used for peers to notify the tracker that they wish to participate in a particular swarm.

The JOIN message is used when the peer has none or just some chunks (LEECH), or has all the chunks (SEED) of a content. The JOIN is used for both on-demand or Live streaming modes.

The peer MUST properly form the XML message-body, set the Request method to JOIN, set the PeerID with the identifier of the peer, set the SwarmID with the identifier of the swarm it is interested in, and randomly generate and set the TransactionID.

A peer seeder may send the JOIN message to participate in several swarms. The PeerMode element SHOULD be set to SEED.

An example of the message-body of a JOIN Request for a peer seeder joining two swarms is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>JOIN</Request>
  <PeerID>656164657221</PeerID>
  <TransactionID>12345.0</TransactionID>
  <SwarmID peerMode="SEED" transactionID="12345.1">1111</SwarmID>
  <SwarmID peerMode="SEED" transactionID="12345.1">2222</SwarmID>
</PPSPTrackerProtocol>
```

When receiving a well-formed JOIN Request the tracker processes the information to check if it is valid and if the peer can join the swarm of interest. In case of success a response message with a Response value of SUCCESSFUL will be generated and the tracker enters the peer information into the corresponding swarm activity. In case the PeerMode is SEED, the tracker just responds with a SUCCESSFUL response and enters the peer information into the corresponding swarm activity.

The response MUST have the same TransactionID value as the request.

An example of a Response message for the above JOIN Request of a seeder is:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>
    <Result transactionID="12345.0">200 OK</Result>
    <Result transactionID="12345.1">200 OK</Result>
    <Result transactionID="12345.2">200 OK</Result>
  <TransactionID>
<PPSPTrackerProtocol>
```

As a leecher only, the peer can participate in one swarm at a time using JOIN requests. In this case, the JOIN request message may include the ContentGroup element to indicate a specific point in the stream. The PeerMode element SHOULD be set to LEECH.

An example of the message-body of a JOIN Request for a peer leecher is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>JOIN</Request>
  <PeerID>656164657220</PeerID>
  <TransactionID>12345</TransactionID>
  <SwarmID peerMode="LEECH">1111</SwarmID>
  <PeerNum abilityNAT="STUN"
           concurrentLinks="HIGH"
           onlineTime="NORMAL"
           uploadBWlevel="NORMAL">5</PeerNum>
  <ContentGroup>
    <Representation id="tag0">
      <SegmentInfo startIndex="20" />
    </Representation>
    <Representation id="tag6">
      <SegmentInfo startIndex="20" />
    </Representation>
  </ContentGroup>
</PPSPTrackerProtocol>
```

The JOIN request MAY include a PeerNum element to indicate to the tracker the number of peers to be returned in a list corresponding to the indicated properties, being @abilityNAT for NAT traversal (considering that PPSP-ICE NAT traversal techniques may be used), and optionally @concurrentLinks, @onlineTime and @uploadBWlevel for the preferred capabilities.

In the case of a JOIN to a specific point in a stream the request SHOULD include a ContentGroup to specify the joining point in terms of content Representations. The above example of a JOIN request would be for the case of an end-user that previously watched a content with a certain audio language, then interrupted for a while (having disconnected) and later continued by re-joining from that point onwards but selecting a different available audio language.

In case the PeerMode is LEECH the tracker will search and select an appropriate list of peers satisfying the conditions to include in the response. The peer list in the response MUST contain the Peer-IDs and the corresponding IP Addresses of the selected peers satisfying the conditions. To create the peer list, the tracker may also take peer status and network location information into consideration, to express network topology preferences or Operators' policy preferences, with regard to the possibility of connecting with other IETF efforts such as ALTO [I.D.ietf-alto-protocol].

The response MUST have the same TransactionID value as the request.

An example of a Response message for the JOIN Request from a leecher is:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>12345</TransactionID>
  <PeerGroup>
    <PeerInfo>
      <PeerID>956264622298</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.22" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
    <PeerInfo>
      <PeerID>3332001256741</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.201" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
  </PeerGroup>
</PPSPTrackerProtocol>
```

The Response MUST include a PeerGroup with PeerInfo data that includes the public IP address of the selected active peers in the swarm.

The tracker MAY also include the attribute @asn with network location information of the transport addresses of the peers, corresponding to the Autonomous System Numbers of the access network provider of each peer in the list.

4.4. FIND Request

This method allows peers to request to the tracker, whenever needed, a new peer list for the swarm or for specific scope of chunks of a media content representation of that swarm.

The peer MUST properly form the XML message-body, set the request method to FIND, set the PeerID with the identifier of the peer, set the SwarmID with the identifier of the swarm the peer is interested in. Optionally, in order to find peers having the specific chunks, the peer also may include information about the desired content in the JOIN request message.

This message is mainly used for leechers to update the peer list.

The peer MUST generate and set the TransactionID for the request.

An example of the message-body of a FIND Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>FIND</Request>
  <PeerID>656164657221</PeerID>
  <SwarmID>1111</SwarmID>
  <TransactionID>12345</TransactionID>
  <PeerNum abilityNAT="STUN"
           concurrentLinks="HIGH"
           onlineTime="NORMAL"
           uploadBWlevel="NORMAL">5</PeerNum>
  <ContentGroup>
    <Representation id="tag4">
      <SegmentInfo startIndex="110" endIndex="150" />
    </Representation>
  </ContentGroup>
</PPSPTrackerProtocol>
```

The FIND request MAY include a PeerNum element to indicate to the tracker the number of peers to be returned in a list corresponding to the indicated properties, being @abilityNAT for NAT traversal (considering that PPSP-ICE NAT traversal techniques may be used), and optionally @concurrentLinks, @onlineTime and @uploadBWlevel for the preferred capabilities.

In the case of a FIND with a specific scope of a stream content the request SHOULD include a ContentGroup to specify the content Representations segment range of interest.

When receiving a well-formed FIND Request the tracker processes the information to check if it is valid. In case of success a response message with a Response value of SUCCESSFUL will be generated and the tracker will include the appropriate list of peers satisfying the conditions requested. The peer list returned MUST contain the Peer-IDs and the corresponding IP Addresses.

The tracker may take peer status and network location information into consideration when selecting the peer list to return, to express network topology preferences or Operators' policy preferences, with regard to the possibility of connecting with other IETF efforts such as ALTO [I.D.ietf-alto-protocol].

An example of a Response message for the FIND Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>12345</TransactionID>
  <PeerGroup>
    <PeerInfo>
      <PeerID>956264622298</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.22" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
    <PeerInfo>
      <PeerID>3332001256741</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.201" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
  </PeerGroup>
</PPSPTrackerProtocol>
```

The Response MUST include a PeerGroup with PeerInfo data that includes the public IP address of the selected active peers in the swarm.

The tracker MAY also include the attribute @asn with network location information of the transport addresses of the peers, corresponding to the Autonomous System Numbers of the access network provider of each peer in the list.

The response MAY also include a PeerGroup with PeerInfo data that includes the requesting peer public transport IP address. If STUN-like function is enabled in the tracker, the PeerAddress includes the attribute @type with a value of REFLEXIVE, corresponding to the transport address "candidate" of the requesting peer.

An example of a Response message for the FIND Request including the requesting peer public IP address is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>12345</TransactionID>
  <PeerGroup>
    <PeerInfo>
      <PeerID>656164657221</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.1" port="80"
                   priority="1"
                   type="REFLEXIVE"
                   connection="3G"
                   asn="64496" />
    </PeerInfo>
    <PeerInfo>
      <PeerID>956264622298</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.22" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
    <PeerInfo>
      <PeerID>3332001256741</PeerID>
      <PeerAddress addrType="ipv4" ip="198.51.100.201" port="80"
                   asn="64496" peerProtocol="PPSP-PP" />
    </PeerInfo>
  </PeerGroup>
</PPSPTrackerProtocol>
```

4.5. Enhanced STAT_REPORT Request

This message still uses the specifications of PPSP-TP/1.0 defined in [I-D.cruz-ppsp-base-tracker-protocol]. The Stat element has been extended with one property, "ContentMap", to allow peers reporting map of chunks they have. The tracker would not have the ability to treat the FIND and JOIN requests for specific content chunks, unless peers report this kind of information. Examples are provided below.

An example of the message-body of an enhanced STAT_REPORT request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Request>STAT_REPORT</Request>
  <PeerID>656164657221</PeerID>
  <TransactionID>12345</TransactionID>
  <StatisticsGroup>
```

```
      <Stat property="StreamStatistics">
        <SwarmID>1111</SwarmID>
        <UploadedBytes>512</UploadedBytes>
        <DownloadedBytes>768</DownloadedBytes>
        <AvailBandwidth>1024000</AvailBandwidth>
      </Stat>
      <Stat property="StreamStatistics">
        <SwarmID>2222</SwarmID>
        <UploadedBytes>1024</UploadedBytes>
        <DownloadedBytes>2048</DownloadedBytes>
        <AvailBandwidth>512000</AvailBandwidth>
      </Stat>
      <Stat property="ContentMap">
        <SwarmID>1111</SwarmID>
        <Representation id="tag0">
          <SegmentInfo startIndex="0" endIndex="24"
                     chunkmapSize="25">
            A/8D/wP/A/8D/wP/A/8D/wP/A/8D/wP/....
          </SegmentInfo>
        </Representation>
        <Representation id="tag1">
          <SegmentInfo startIndex="0" endIndex="14"
                       chunkmapSize="15">
            A/8D/wP/A/8D/wP/A/8D/wP/A/8D/wP/....
          </SegmentInfo>
          <SegmentInfo startIndex="20" endIndex="24"
                           chunkmapSize="5">
            A/8D/wP/A/8D/wP/A/8D/wP/A/8D/wP/....
          </SegmentInfo>
        </Representation>
      </Stat>
        <Stat property="ContentMap">
          <SwarmID>2222</SwarmID>
          <Representation id="tag5">
          <SegmentInfo startIndex="0" endIndex="4"
                       chunkmapSize="5">
            A/8D/wP/A/8D/wP/A/8D/wP/A/8D/wP/....
          </SegmentInfo>
        </Representation>
        <Representation id="tag6">
          <SegmentInfo startIndex="0" endIndex="4"
                       chunkmapSize="5">
            A/8D/wP/A/8D/wP/A/8D/wP/A/8D/wP/....
          </SegmentInfo>
        </Representation>
      </Stat>
    </StatisticsGroup>
  </PPSPTrackerProtocol>
```

If the request is valid the tracker process the received information for future use, and generates a response message with a Response value of SUCCESSFUL.

The response MUST have the same TransactionID value as the request.

An example of a Response message for the START_REPORT Request is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSPTrackerProtocol xmlns="TBD"
                     schemaLocation="TBD"
                     version="1.1">
  <Response>SUCCESSFUL</Response>
  <TransactionID>12345</TransactionID>
</PPSPTrackerProtocol>
```

4.6. Error and Recovery Conditions

This document does not introduce any new error and recovery conditions. The implementation of error treatment MUST refer to PPSP-TP-1.0 specification [I-D.cruz-ppsp-base-tracker-protocol], subsection 8.6.

5. Security Considerations

The PPSP-TP/1.1 proposed in this document introduces no new security considerations beyond those described in PPSP-TP/1.0 specification [I-D.cruz-ppsp-base-tracker-protocol].

6. IANA Considerations

There are presently no IANA considerations with this document.

7. Acknowledgments

The authors would like to thank many people for their help and comments, particularly: Zhang Yunfei, Zong Ning, Martin Stiemerling, Johan Pouwelse and Arno Bakker. The authors would also like to thank the people participating in the EU FP7 project SARACEN (contract no. ICT-248474) [refs.saracenwebpage] for contributions and feedback to this document.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the SARACEN project or the European Commission.

8 References

8.1 Normative References

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

8.2 Informative References

[I-D.ietf-ppsp-problem-statement] Zhang, Y., Zong, N., Camarillo, G., Seng, J., and Y. Yang, "Problem Statement of P2P Streaming Protocol (PPSP)", draft-ietf-ppsp-problem-statement-11 (work in progress), October 2012.

[I-D.cruz-ppsp-base-tracker-protocol] Cruz, R., Nunes, M., Gu, Y., Xia, J., J. Taveira and D. Lingli, "PPSP Tracker Protocol-Base Protocol (PPSP-TP/1.0)", draft-cruz-ppsp-base-tracker-protocol-00 (work in progress), June 2012.

[I.D.ietf-alto-protocol] Alimi, R., Penno, R., Yang, Y., "ALTO Protocol", draft-ietf-alto-protocol-13, (work in progress), September 2012.

[refs.saracenwebpage] "SARACEN Project Website", http://www.saracen-p2p.eu/.

Authors' Addresses

Rachel Huang
Huawei
Phone: +86-25-56623633
EMail: rachel.huang@huawei.com

Rui Santos Cruz
IST/INESC-ID/INOV
Phone: +351.939060939
Email: rui.cruz@ieee.org

Mario Serafim Nunes
IST/INESC-ID/INOV
Rua Alves Redol, n.9
1000-029 LISBOA, Portugal
Phone: +351.213100256
Email: mario.nunes@inov.pt

Joao P. Taveira
IST/INOV
Email: joao.silva@inov.pt