International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.609.7

(12/2018)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

OSI networking and system aspects – Networking

Managed P2P communications: Content distribution peer protocol

Recommendation ITU-T X.609.7

International Telecommunication Union

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	
Services and facilities	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalling and switching	X.50–X.89
Network aspects	X.90–X.149
Maintenance	X.150–X.179
Administrative arrangements	X.180–X.199
OPEN SYSTEMS INTERCONNECTION	
Model and notation	X.200–X.209
Service definitions	X.210–X.219
Connection-mode protocol specifications	X.220–X.229
Connectionless-mode protocol specifications	X.230–X.239
PICS proformas	X.240–X.259
Protocol Identification	X.260–X.269
Security Protocols	X.270–X.279
Layer Managed Objects	X.280–X.289
Conformance testing	X.290–X.299
INTERWORKING BETWEEN NETWORKS	
General	X.300–X.349
Satellite data transmission systems	X.350–X.369
IP-based networks	X.370–X.379
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	
Networking	**X.600–X.629**
Efficiency	X.630–X.639
Quality of service	X.640–X.649
Naming, Addressing and Registration	X.650–X.679
Abstract Syntax Notation One (ASN.1)	X.680–X.699
OSI MANAGEMENT	
Systems management framework and architecture	X.700–X.709
Management communication service and protocol	X.710–X.719
Structure of management information	X.720–X.729
Management functions and ODMA functions	X.730–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	
Commitment, concurrency and recovery	X.850–X.859
Transaction processing	X.860–X.879
Remote operations	X.880–X.889
Generic applications of ASN.1	X.890–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
INFORMATION AND NETWORK SECURITY	X.1000–X.1099
SECURE APPLICATIONS AND SERVICES (1)	X.1100–X.1199
CYBERSPACE SECURITY	X.1200–X.1299
SECURE APPLICATIONS AND SERVICES (2)	X.1300–X.1499
CYBERSECURITY INFORMATION EXCHANGE	X.1500–X.1599
CLOUD COMPUTING SECURITY	X.1600–X.1699

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T X.609.7

Managed P2P communications: Content distribution peer protocol

Summary

Recommendation ITU-T X.609.7 specifies a content distribution peer protocol (CDPP) that runs on the interface among entities of managed peer-to-peer (P2P) communications. CDPP is used to distribute one or more contents to a number of peers. Content distribution over traditional P2P communications has incurred various issues such as distribution of illegal content, uncontrollable participation and synchronized distribution of the updated contents. Different from the content distribution over the traditional P2P communications which is not capable of providing manageability, the content distribution over managed P2P communications can be managed by a content provider or service provider. In the content distribution over managed P2P communications, as an example, participation in an overlay network can be controlled so that only predefined peers can join the overlay network and distribute contents to each other. In addition, content to be distributed over an overlay network can be updated anytime and every update will be applied to all peers in the overlay network. The protocol is capable of managing content distribution under the control of a content provider or service provider. This Recommendation provides protocol operations and message formats for content distribution over a managed P2P network.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.609.7	2018-12-14	11	11.1002/1000/13802

Keywords

Content distribution, managed P2P, peer protocol.

---

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

© ITU 2019

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

Table of Contents

Recommendation ITU-T X.609.7

Managed P2P communications: Content distribution peer protocol

1 Scope

This Recommendation specifies a content distribution protocol over managed P2P communications. It describes the following:

– overview of content delivery peer protocol;

– protocol messages and its parameters;

– protocol behaviours

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T X.609] Recommendation ITU-T X.609 (2015), *Managed peer-to-peer (P2P) communications: Functional architecture.*

[ITU-T X.609.6] Recommendation ITU-T X.609.6 (2018), *Managed P2P communications: Content distribution signalling requirements.*

[IETF RFC 7159] IETF RFC 7159 (2014), *The JavaScript Object Notation (JSON) Data Interchange Format.*

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 overlay network [b-ITU-T X.1162]: An overlay network is a virtual network that runs on top of another network. Like any other network, the overlay network comprises a set of nodes and links between them. Because the links are logical ones, they may correspond to many physical links of the underlying network.

3.1.2 peer [b-ITU-T X.1161]: Communication node on P2P network that functions simultaneously as both "client" and "server" to the other nodes on the network.

3.1.3 peer-to-peer (P2P) [b-ITU-T Y.2206]: A system is considered to be P2P if the nodes of the system share their resources in order to provide the service the system supports. The nodes in the system both provide services to other nodes and request services from other nodes.

NOTE – Peer is the node in a P2P system.

3.1.4 managed P2P [b-ISO/IEC TR 20002]: P2P with manageability features to manage the P2P-based service and P2P network by the P2P participants such as P2P service provider, ISP, and peer.

3.1.5 buffermap [ITU-T X.609]: A map showing downloading status of fragments comprising a shared content.

3.1.6 fragment [ITU-T X.609]: A piece of the shared content.

3.1.7 fragmentation [ITU-T X.609]: A process that divides the shared content into multiple fragments for sharing the content in a distributed manner.

3.1.8 source peer [b-ITU-T X.609.3]: A peer that streams the multimedia contents to the overlay network. The peer only provides content data to other peers and does not receive it. This peer generates fragments using the multimedia data received from the content source.

3.1.9 client peer [b-ITU-T X.609.3]: A peer that sends fragments received from other peers to other peers, and does not generate its own fragments.

3.2 Terms defined in this Recommendation

None.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

CDPP Content Distribution Peer Protocol

CP Completed Point

CS Completed Section

DP Downloading Point

DS Downloading Section

IXS Index Server

JSON JavaScript Object Notation

MP2P Managed Peer-to-Peer

NAT Network Address Translation

P2P Peer-to-Peer

SP Start Point

TCP Transmission Control Protocol

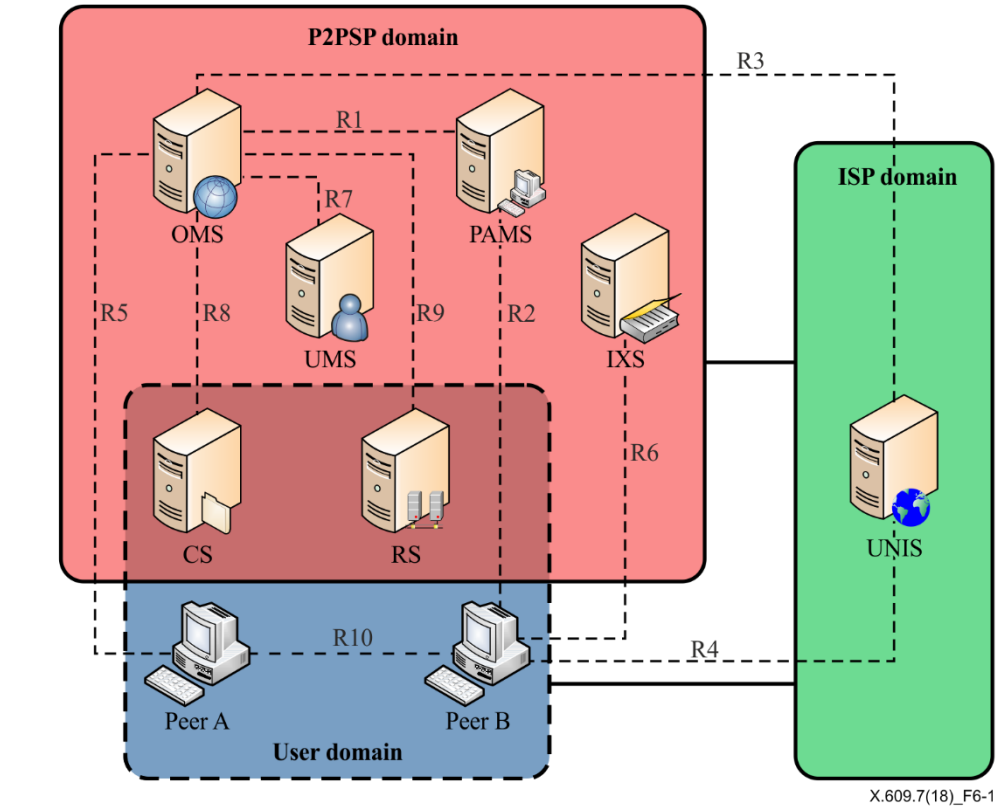
UDP User Datagram Protocol

5 Conventions

None.

6 Overview

Figure 6-1 shows the framework and reference points of managed peer-to-peer (MP2P) communications defined in [ITU-T X.609].



UMS: user profile management server
OMS: overlay management server
PAMS: peer activity management server
RS: relay server
CS: cache server
IXS: index server
UNIS: underlying network information server

Figure 6-1 – Framework and reference points of managed P2P [ITU-T X.609]

Based on the functional architecture defined in [ITU-T X.609], [ITU-T X.609.6] describes signalling requirements for content distribution services. Figure 6-2 shows an architectural overview of content distribution service over managed peer-to-peer (MP2P) communications defined in this Recommendation. This Recommendation defines a protocol for the reference point R10.

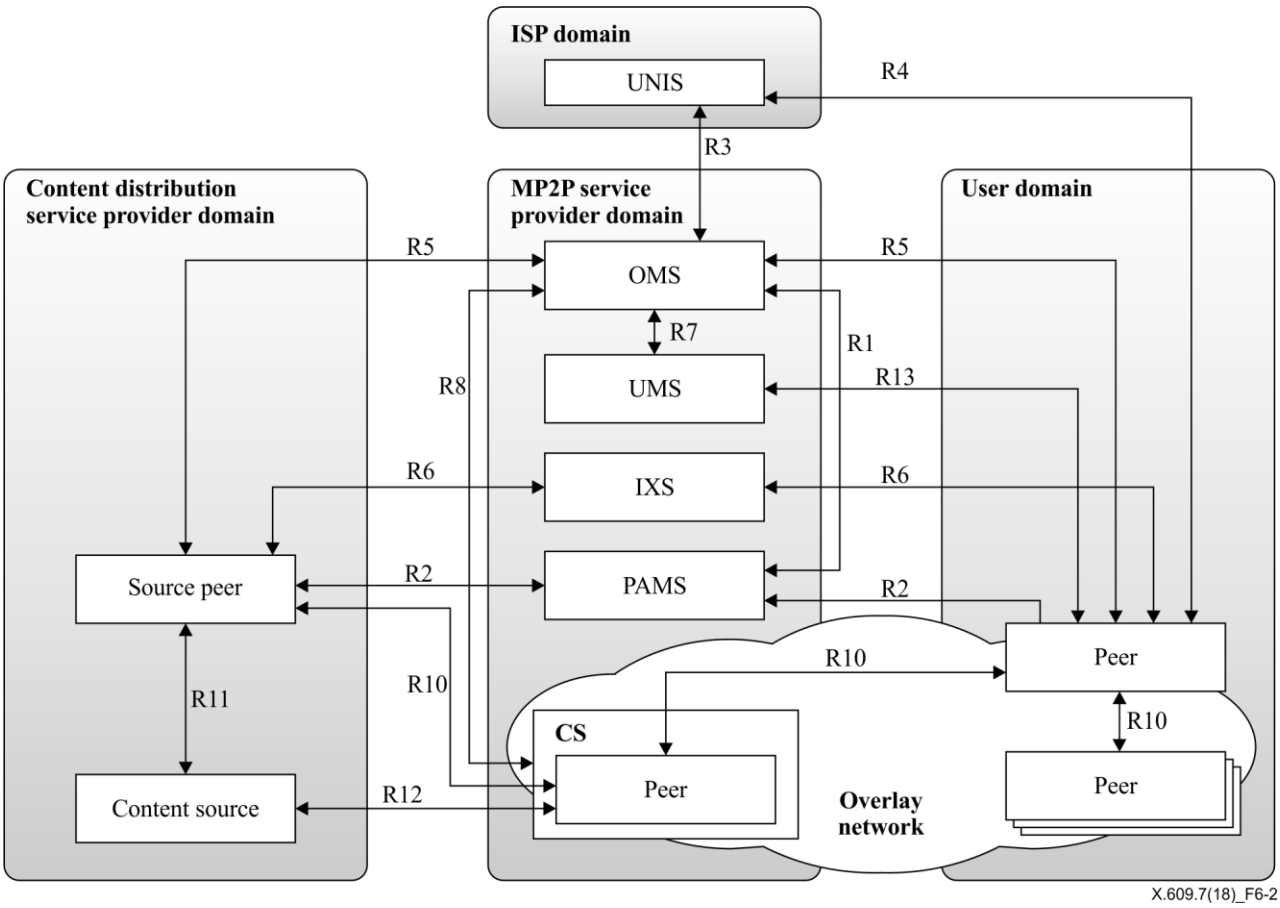


Figure 6-2 – Functional architecture of MP2P communications for content distribution

7 Messages and behaviours

7.1 Messages and behaviours

This clause describes messages and behaviours for content distribution service based on MP2P communications defined in this Recommendation. This Recommendation defines the peer protocol for reference point R10 of the functional architecture depicted in Figure 6-2.

7.2 Buffer and buffermap

A peer generates a buffer when it starts content distribution. A source peer willing to distribute a content has the buffer fully filled with entire fragments of the content. The client peer starts with an empty buffer and the buffer is gradually filled with the fragments by communication with the source peer and other client peers. For the communication, each peer maintains the information of its own buffer and shares the information with other peers in order to find which fragments it needs to get and which peer can send the fragments. The information is called as buffermap.

7.1.1 Structure of a buffer

Figure 7-1 shows the structure of a buffer.

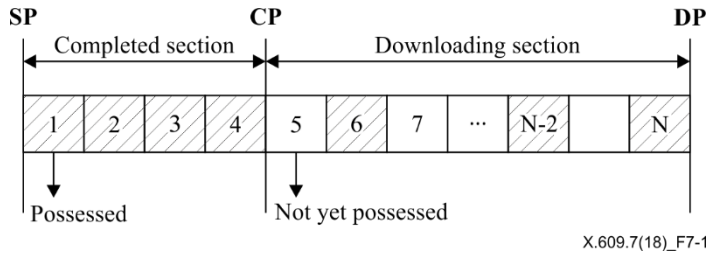


Figure 7-1 – Structure of buffer

A buffer is composed of the following sections;

– **Completed section (CS)**: CS contains the contiguous fragments received from other peer(s). Thus a peer can request any fragment in the CS from another peer;

– **Downloading section (DS)**: DS includes zero or more fragments received from other peer(s). Thus a peer needs a buffermap describing which fragments are in DS.

The following points are used to represent the start or end of each section in a buffer:

– **Completed point (CP)**: CP points to the end of the completed section in the buffermap;

– **Downloading point (DP)**: DP points to the end of the downloading section in the buffermap;

– **Start point (SP)**: SP points to the identifier at the beginning of the buffermap and is always set to zero.

7.1.2 Structure of a buffermap

A buffermap of a buffer represents the possession status of each fragment of a content within a downloading section. The possessed fragment is represented as one (1) and the fragment not yet possessed is represented as zero (0). The buffermap corresponding to the buffer shown in Figure 7-1 is depicted in Figure 7-2. A buffermap is used as a payload of HELLO messages, PELLO messages, and BUFFERMAP messages.

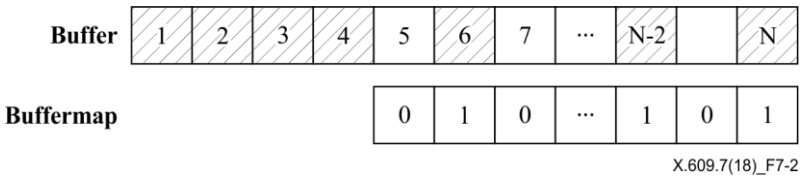


Figure 7-2 – A buffermap corresponding to an example buffer

7.2 Messages

This clause defines protocol messages for content distribution service and the messages are represented as JavaScript Object Notation (JSON) [IETF RFC 7159]. Since JSON itself cannot accommodate binary data, messages are converted to binary format such as BSON [b-BSON 1.1].

7.2.1 HELLO

HELLO message is used for establishing a connection between two peers. After client peer A establishes a transmission control protocol (TCP) connection with client peer B, it sends a HELLO message and then client peer B responds with a HELLO message. The syntax of the message is as follows;

```
{
   '"method":"HELLO",
   "proto-version":integer,
```

```
    "index-version":integer,
    "peer-id":string,
    "overlay-id":string,
    "sp-index":integer,
    "cp-length":integer,
    "dp-index":integer,
    "ds-length":integer,
    "buffermap":binary
}
```

The semantic of each field in the message is as follows:

– *proto-version* is the version of the protocol;
– *index-version* is the version of the index file;
– *peer-id* is the identifier of the peer;
– *overlay-id* is the identifier of overlay network;
– *sp-index* indicates the starting index number of fragment represented by the buffermap;
– *cp-length* indicates the length of completed section within buffermap beginning *sp-index*;
– *dp-index* indicates the starting index number of downloading section;
– *ds-length* indicates the length of downloading section;
– *buffermap* represents the possession status of fragments within downloading section. If the peer possesses particular fragment, it set to '1', and set to '0', if not.

7.2.2 PELLO

PELLO message is used to establish a connection with the peer behind a network address translation (NAT) or a firewall. If a source peer is located behind a NAT or a firewall, client peers cannot connect to the source peer. Client peer A receiving this message from client peer B acts as if it sent a HELLO message to client peer B and uses the same TCP connection in order to request and receive fragments. A source peer can actively establish TCP connections with client peers and sends this message to the client peers. The syntax of the message is as follows;

```
{
    "method":"PELLO",
    "proto-version":integer,
    "index-version":integer,
    "peer-id":string,
    "overlay-id":string,
    "sp-index":integer,
    "cp-length":integer,
    "dp-index":integer,
    "ds-length":integer,
    "buffermap":binary
}
```

The semantic of each field in the message is as follows:

– *proto-version* is the version of the protocol;
– *index-version* is the version of the index file;
– *peer-id* is the identifier of the peer;
– *overlay-id* is the identifier of overlay network;
– *sp-index* indicates the starting index number of fragment represented by the buffermap;
– *cp-length* indicates the length of completed section within buffermap beginning *sp-index*;
– *dp-index* indicates the starting index number of downloading section;
– *ds-length* indicates the length of downloading section;
– *buffermap* represents the possession status of fragments within downloading section. If the peer possesses particular fragment, it set to '1', and set to '0', if not.

7.2.3 REFRESH

This message is used to request a peer to send the latest buffermap. When a client peer receives all fragments of another peer, which may be a client peer or a source peer, it requests the peer to send the latest buffermap by sending this message. The syntax of the message is as follows;

```
{
  "method": "REFRESH",
  "piece-index"; integer,
  "piece-number": integer
}
```

The semantic of each field in the message is as follows:

– *piece-index* is an index number of fragment that requesting peer is interested. In this Recommendation, this value is always set to '1';
– *piece-number* indicates the number of fragments to be represented by buffermap. If this value is set to '0', the peer sends a whole buffermap after *piece-index*. In this Recommendation this value is set to '0' as default, but other values can be used.

7.2.4 BUFFERMAP

This message is used to deliver a buffermap of a peer. The syntax of the message is as follows;

```
{
  "method": "BUFFERMAP"
  "piece-index": integer,
  "cp-length": integer,
  "dp-index": integer,
  "ds-length": integer,
  "buffermap": binary
}
```

The semantic of each field in the message is as follows:

– *piece-index* is an index number requested by REFRESH message. this value is always set to '1';
– *cp-length* indicates the length of completed section within buffermap;
– *dp-index* indicates the starting index number of downloading section;

– *ds-length* indicates the length of downloading section;

– *buffermap* represents the possession status of fragments within downloading section. If the peer possesses particular fragment, it set to '1', and set to '0', if not.

7.2.5 GET

This message is used for requesting a particular fragment to a peer. The syntax of the message is as follows;

```
{
  "method":"GET"
  "piece-index":integer,
  "offset":integer
}
```

The semantic of each field in the message is as follows:

– *piece-index* is the index number of fragment being requested;

– *offset* is set to 0(zero) if it needs whole data of the fragment. If it has part of the fragment, a requesting peer specify the offset value to this field, and then requested peer sends the fragment from the offset.

When a client peer requests an index file of another peer, the client peer sends a GET message with the following format;

```
{
  "method":"GET"
  "piece-index":0
}
```

The semantic of each field in the message is as follows:

– *piece-index* is set to 0('zero') for *requesting* index file.

7.2.6 BUSY

This message is used for rejecting connection establishment with a requesting peer due to lack of resources of the requested peer. The syntax of the message is as follows;

```
{
  "method":"BUSY"
  "reason":string,
   "index-file":binary,
}
```

The semantic of each field in the message is as follows:

– *reason* is set to convey reasons of denial. This can be "shortage of bandwidth, the number of concurrent connection has been exceeding, internal resources such as storage and CPU are busy");

– *index-file* contains index file of the requested peer if the peer requesting a connection has the lower version of index file.

7.2.7 DATA

This message is used for delivering fragments. A peer can use the hash value to verify integrity of the fragment from another peer. This message may contain signature that is signed by a private key

of a source peer on the hash value to prevent from malicious manipulation. The syntax of the message is as follows;

```
{
  "method":"DATA"
  "piece-index":integer,
  "offset":integer,
  "data-size":integer,
  "timestamp":string,
  "hash":string,
  "signature":string,
  "encrypted-hash":string,
  "data":binary
}
```

The semantic of each field in the message is as follows:

– *piece-index* is an index number of fragment;
– *offset* indicates the offset for a particular fragment;
– *data-size* indicates the size of fragment;
– *timestamp* indicates the creation time of fragments with NTP timestamp format;
– *hash* contains SHA-1 hashing value for the fragment;
– *signature* contains the signed hash value by digital signature of source peer;
– *encrypted-hash* contains encrypted hash value by digital signature of source peer;
– *data* contains fragment.

When a peer needs to send its index file, it uses a DATA message with the following format;

```
{
  "method":"DATA"
  "piece-index":0,
 "data-size":integer,
 "data": binary
}
```

The semantic of each field in the message is as follows:

– *piece-index* is set to 0('zero') for sending index file;
– *data-size* indicates the size of index file;
– *data* contains index file.

7.2.8 CANCEL

This message is used to revoke a pending request on a particular fragment to a peer. The syntax of the message is as follows;

```
{
  "method": "CANCEL",
```

```
  "piece-index": integer,
  "offset": integer
}
```

The semantic of each field in the message is as follows:

– *piece-index* is an index number of fragment;

– *offset* indicates the offset for a particular fragment.

7.2.9 BYE

This message is used for the release of a connection between two peers. The syntax of the message is as follows;

```
{
  "method": "BYE"
}
```

7.3 Behaviour of peers

7.3.1 Overview

In this Recommendation, a peer is either a source peer or a client peer. A source peer is the origin of the content to be distributed and generates an overlay network for distributing the content. The source peer can also provide the latest index file if there is any change in the content. As a receiver of the content, a client peer participates in an overlay network corresponding to the content and receives the content. The client peer also shares the received fragments of the content with other client peers.

For distributing a content, peers need to conduct protocol behaviours and the behaviour of a peer is dependent on operational phases and the type of the peer. This Recommendation defines three operational phases; provisioning phase, peering phase and updating phase. This clause describes the peer behaviour in the peering phase and in the updating phase.

NOTE – A source peer establishes an overlay network for distributing a content in the provisioning phase. However, the provisioning phase is not under the scope of this Recommendation.

7.3.2 Peering phase

In the peering phase, peers exchange protocol messages with each other to establish a connection for sending and receiving fragments of a content. After establishing a connection between peers, they can exchange the fragments or an index file.

7.3.2.1 Source peer

A source peer in peering phase may exchange a HELLO message with client peers or send a PELLO message to client peers for establishing a connection between the client peers. After a connection is established, the source peer may receive a GET message requesting fragments of its content and then it can send a DATA message including the requested fragments.

7.3.2.1.1 Connection establishment

On receiving a HELLO message from a client peer, a source peer can send a HELLO message including its own buffermap composed of CS only. If there is no resource available, the source peer responds with a BUSY message and disconnects the TCP connection.

To establish a connection, a source peer can also send a PELLO message to a client peer instead of waiting for a HELLO message.

7.3.2.1.2 Fragment transmission

On receiving a GET message from a client peer, a source peer can send a DATA message including the fragment(s) as requested by the client peer. If the source peer receives a CANCEL message before sending the requested fragment(s), it cancels fragment transmission as requested.

7.3.2.1.3 Connection termination

A source peer can terminate a connection with a client peer by sending a BYE message. The source peer may receive a BYE message from a client peer. In both cases, the source peer disconnects the TCP connection with the client peer.

If the connection with a certain peer is lost without receiving a BYE message, the source peer assumes that a BYE message had been received.

7.3.2.2 Client peer

A client peer in peering phase may exchange a HELLO message with a source peer or other client peers. After a connection is established with another peer, the client peer can send a GET message in order to request fragments which it needs. The client peer may receive a GET message from other client peers and then it can send a DATA message including the requested fragments.

7.3.2.2.1 Connection establishment

After sending a HELLO message to a source peer or to another client peer, a client peer will receive a HELLO message as a response. After exchanging a HELLO message with either a source peer or a client peer, the client peer can request the fragments which the peer in the established connection has.

When a client peer receives a BUSY message, it will disconnect the TCP connection.

On receiving a PELLO message from a source peer, a client peer compares its own buffermap with the buffermap included in the PELLO message in order to find any fragment which the client peer needs. The client peer that received the PELLO message behaves as if it receives a HELLO message from another peer without sending a HELLO message.

7.3.2.2.2 Fragment reception

After establishing a connection, a client peer can request a fragment which the peer it is connected with has. The client peer sends a GET message indicating a specific fragment which it needs. As a response, the client peer may receive a DATA message including the requested fragments.

7.3.2.2.3 Fragment transmission

On receiving a GET message from another client peer, a client peer can send a DATA message including the fragment(s) as requested by the client peer. If the client peer receives a CANCEL message before sending the requested fragment(s), it cancels fragment transmission as requested.

7.3.2.2.4 Connection termination

A client peer can terminate a connection with another peer by sending a BYE message. The client peer may receive a BYE message from another peer. In both cases, the client peer disconnects the TCP connection with the peer.

If the connection with a certain peer is lost without receiving a BYE message, the client peer assumes that a BYE message had been received.

7.3.3 Updating phase

Updating phase happens when a content that had been distributed has changes. A source peer needs to distribute a new version of the index file and content. A client peer will try to get the new version of the content after having the new version of the index file.

7.3.3.1 Source peer

The change of a content includes addition or removal of files organizing the content. After the content is changed, a source peer generates a new version of the index file and distributes the index file and the new version of content.

7.3.3.1.1 Index file distribution

The source peer with a new version of the index file sends the index file to the index server (IXS), which is not under the scope of this Recommendation. By exchanging HELLO messages, the source peer can also notify the new version of the index file to client peers. Upon receiving a GET message requesting the new version of the index file, the source peer sends a DATA message including the index file.

7.3.3.1.2 Fragment transmission

Fragment transmission in the updating phase is the same as the fragment transmission in the peering phase.

7.3.3.2 Corresponding peer

By receiving a HELLO message or a PELLO message with a newer version, a client peer can be aware of changes in the content being distributed. Then the client peer tries to get the new version of the index file and to apply the change in the content. If a client peer participates in an overlay network after a source peer sends the latest index file to the IXS, the client peer can get the latest index file from the IXS.

7.3.3.2.1 New version awareness

A client peer may receive a HELLO message or a PELLO message from other client peers or a source peer, respectively. If the version information in the messages is higher than the version which the client peer knows, the client peer can be aware of changes in the content.

7.3.3.2.2 Index file reception

In order to get the latest version of the index file, a client peer sends a GET message to the peer that sent the HELLO message or the PELLO message notifying of change in the content. The requested index file will be delivered through a DATA message as a response of the GET message.

7.3.3.2.3 Index file transmission

A client peer with the latest version of the index file may receive a GET message requesting the index file. Upon receiving the GET message, the client peer can send a DATA message including the requested index file to the peer.

Annex A

Consideration on web-based content distribution

(This annex forms an integral part of this Recommendation.)

A.1 Overview

Nowadays, content is usually consumed in a web platform such as a web browser as well as in native applications. Hence, it is worth considering the support of web-based content distribution. For web-based content distribution, it is also considered that the web environment cannot use legacy transport protocol, such as TCP and UDP. Instead, the modern web application uses WebRTC and WebSocket for communication with other applications or a server. This annex describes how ITU-T X.609.7 can be used with WebRTC for web-based content distribution.

A.2 Recommendation ITU-T X.609.7 over WebRTC

Figure A.1 shows the general procedure for web-based content distribution.

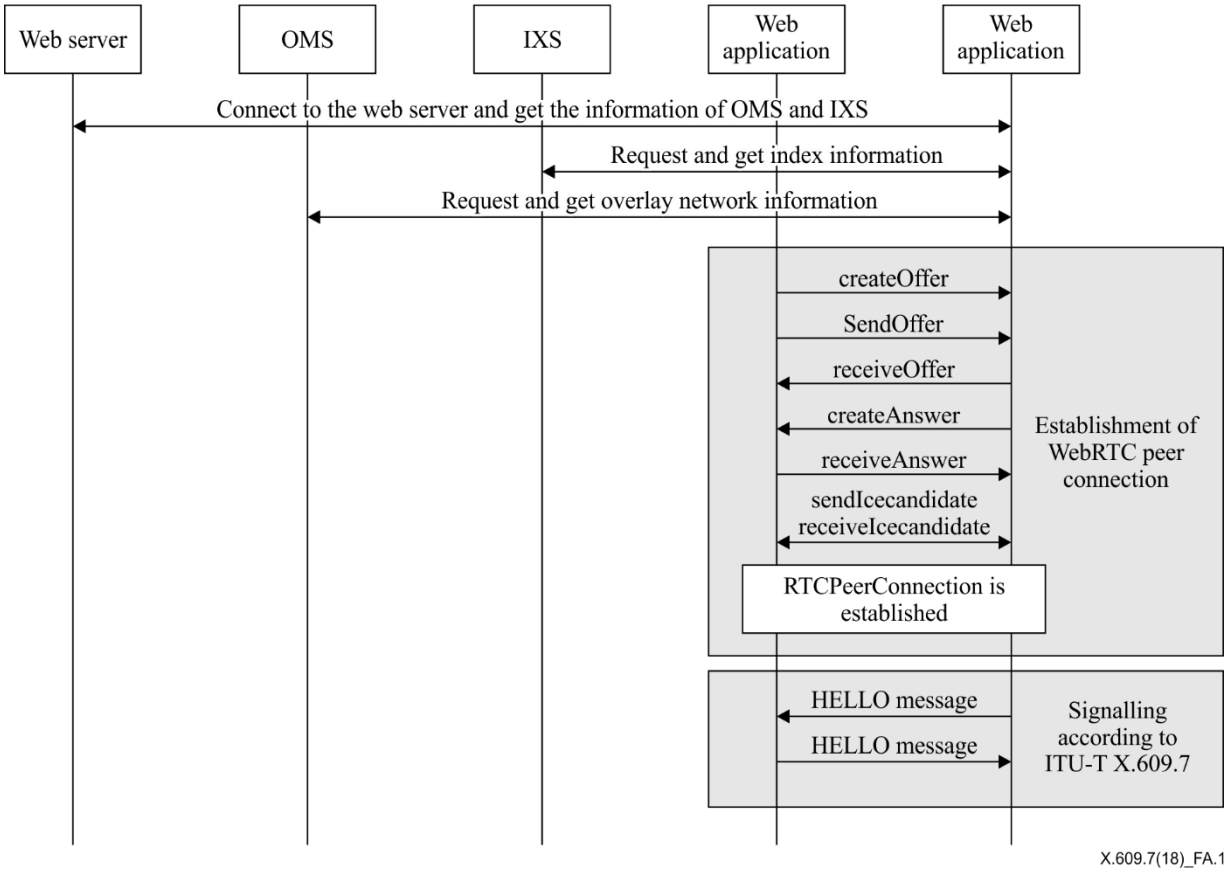


Figure A.1 – General procedure for web-based content distribution

Bibliography

[b-ITU-T X.609.3] Recommendation ITU-T X.609.3 (2017), *Managed P2P communications: Multimedia streaming signalling requirements*.

[b-ITU-T X.1161] Recommendation ITU-T X.1161 (2008), *Framework for secure peer-to-peer communications*.

[b-ITU-T X.1162] Recommendation ITU-T X.1162 (2008), *Security architecture and operations for peer-to-peer networks*.

[b-ITU-T Y.2206] Recommendation ITU-T Y.2206 (2010), *Requirements for distributed service networking capabilities*.

[b-ISO/IEC TR 20002] ISO/IEC TR 20002:2012 *Information technology - Telecommunications and information exchange between systems - Managed P2P: Framework*.

[b-BSON 1.1] BSON – *Binary JSON specification Version 1.1 (2013)*.
http://bsonspec.org

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	**Data networks, open system communications and security**
Series Y	Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2019