Audio/Video Transport Core Maintenance Working Group
Internet Draft
Intended status: Standards Track
Expires: September 2011

R. van Brandenburg
H.M. Stokking
M.O. van Deventer
O.A. Niamut
F.A. Walraven
TNO Netherlands
I. Vaishnavi
CWI Netherlands
F. Boronat
M. Montagud
Universidad Politecnica de Valencia

March 7, 2011

RTCP for inter-destination media synchronization
draft-brandenburg-avtcore-rtcp-for-idms-00.txt

Status of this Memo

This Internet-Draft is submitted 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/ietf/1id-abstracts.txt

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

This Internet-Draft will expire on September 7, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

This document gives information on an RTCP Packet Type and RTCP XR Block Type including associated SDP parameters for inter-destination media synchronization (IDMS). The RTCP XR Block Type, registered with IANA based on an ETSI specification, is used to collect media play-out information from participants in a group playing-out (watching, listening, etc.) a specific RTP media stream. The RTCP packet type specified by this document is used to distribute a summary of the collected information so that the participants can synchronize play-out.

Typical applications for IDMS are social TV, shared service control (i.e. applications where two or more geographically separated users are watching a media stream together), distance learning, network quiz shows, multi-playing online games, etc.

Table of Contents

1. Introduction

1.1. Inter-destination Media Synchronization

Inter-destination media synchronization (IDMS) refers to the play-out of media streams at two or more geographically distributed locations in a temporally synchronized manner. It can be applied to both unicast and multicast media streams and can be applied to any type and/or combination of streaming media, such as audio, video and text (subtitles). [Boronat2009] provides an overview of technologies and algorithms for IDMS.

IDMS requires the exchange of information on media receipt and playout times. It may also require signaling for the initiation and maintenance of IDMS sessions and groups.

The presented RTCP specification for IDMS is independent of the used synchronization algorithm, which is out-of-scope of this document.

1.2. Applicability of RTCP to IDMS

Currently, most multimedia applications make use of RTP and RTCP [RFC3550]. RTP (Real-time Transport Protocol) provides end-to-end network transport functions suitable for applications requiring real-time data transport, such as audio, video or data, over multicast or unicast network services. The timestamps and sequence number mechanisms provided by RTP are very useful to reconstruct the original media timing, reorder and detect some packet loss at the receiver side.

The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a manner that is scalable to large multicast networks, and to provide minimal control and identification functionality.

RTP receivers and senders provide reception quality feedback by sending out RTCP Receiver Report (RR) and Sender Report (SR) packets [RFC3550] respectively, which may be augmented by eXtended Reports (XR) [RFC3611]. Thus, the feedback reporting features provided by RTCP make QoS monitoring possible and can be used for troubleshooting and fault tolerance management in multicast distribution services such as IPTV.

These protocols are intended to be tailored through modification and/or additions in order to include profile-specific information required by particular applications, and the guidelines on doing so are specified in [RFC5968].

IDMS involves the collection, summarizing and distribution of RTP packet arrival and play-out times. As information on RTP packet arrival times and play-out times can be considered reception quality feedback information, RTCP becomes a promising candidate for carrying out IDMS, which may facilitate implementation in typical multimedia applications.

1.3. Applicability of SDP to IDMS

RTCP XR [RFC3611] defines the Extended Report (XR) packet type for the RTP Control Protocol (RTCP), and defines how the use of XR packets can be signaled by an application using the Session Description Protocol (SDP) [RFC4566].

SDP signaling is used to set up and maintain a synchronization group between Synchronization Clients (SCs). This document describes two SDP parameters for doing this, one for the RTCP XR block type and one for the new RTCP packet type.

This document also allows for a receiver to indicate a used clock source for synchronizing the receiver clock used in the IDMS session. This is also done using an SDP parameter, which is described in this document.

1.4. This document and ETSI TISPAN

ETSI TISPAN [TS 183 063] has specified architecture and protocol for IDMS using RTCP XR exchange and SDP signaling.

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] and indicate requirement levels for compliant implementations.

3. Overview of IDMS operation

This section provides a brief example of how the IDMS RTCP functionality is used. The section is tutorial in nature and does not contain any normative statements.

```
         Alice's  . . . . . . .tv:abc.com . . . . . . . . . . Bob's
         TV (SC)              (MSAS)                     Laptop (SC)
            |                       |                          |
            |      Media Session    |                          |
            |<=====================>|                          |
            |           Invite(URL,Sync-group ID)              |
            |------------------------------------------------->|
            |                       |  Media Session Set-up    |
            |                       |<========================>|
            |                       |                          |
            |                Call set-up                       |
            |<================================================>|
            |                       |                          |
            |       RTP Packet      |        RTP Packet        |
            |<----------------------|------------------------->|
            |      RR + IDMS XR     |                          |
            |---------------------->|       RR + IDMS XR       |
            |                       |<-------------------------|
            |    RTCP IDMS packet   |     RTCP IDMS packet     |
            |<----------------------|------------------------->|
            |                       |                          |
```

Alice is watching TV in her living room. At some point she sees that a football game of Bob's favorite team is on. She sends him an invite to watch the program together. Embedded in the invitation is the link to the media server and a unique sync-group identifier.

Bob, who is also at home, receives the invite on his laptop. He accepts Alice's invitation and the RTP client on his laptop sets up a session to the media server. A VoIP connection to Alice's TV is also set up, so that Alice and Bob can talk while watching the program.

As is common with RTP, both the RTP client in Alice's TV as well as the one in Bob's laptop send periodic RTCP Receiver Reports (RR) to the media server. However, in order to make sure Alice and Bob see the events in the football game at the same time, their clients also periodically send an IDMS XR block to the MSAS function of the media server. Included in the XR blocks are timestamps on when both Alice and Bob have received (or played out) a particular RTP packet.

The MSAS function in the media server calculates a reference client from the received IDMS XR blocks (e.g. by selecting whichever client received the packet the latest as the reference client). It then sends an RTCP IDMS packet containing the play-out information of this reference client to both Alice and Bob.

In this case Bob has the slowest connection and the reference client therefore includes a delay similar to the one experienced by Bob. Upon reception of this information, Alice’s RTP client can choose what to do with this information. In this case it decreases its play-out rate temporarily until it matches with the reference client play-out (and thus matches Bob’s play-out). Another option for Alice’s TV would be to simply pause playback until it catches up. The exact implementation of the synchronization algorithm is up to the client.

Upon reception of the reference client RTCP IDMS packet, Bob’s client does not have to do anything since it is already synchronized to the reference client (since it is based on Bob’s delay). Note that other synchronization algorithms may introduce even more delay than the one experienced by the most delayed client, e.g. to account for delay variations, for new clients joining an existing synchronization group, etc.

4. Inter-destination media synchronization use cases

Social TV is the combination of media content consumption by two or more users at different devices and locations and real-time communication between those users.

An example of Social TV, is when two or more users are watching the same television broadcast at different devices and locations, while communicating with each other using text, audio and/or video.

A skew in the media play-out of the two or more users can have adverse effects on their experience. A well-known use case here is one friend experiencing a goal in a football match well before or after other friend(s). Thus IDMS is required to provide play-out synchronization.

Another example of Social TV is Shared Service Control, where two or more users experience some content-on-demand together, while sharing the trick-play controls (play, pause, fast forward, rewind) of the content on demand.

Similar to the previous use case, without IDMS, differences in play-out speed and the effect of transit delay of trick-play control signals would desynchronize content play-out.

5. Architecture for inter-destination media synchronization

The architecture for IDMS, which is based on a sync-maestro architecture [Boronat2009], is sketched below. The Synchronization Client (SC) and Media Synchronization Application Server (MSAS) entities are shown as additional functionality for the RTP receiver and sender respectively.

It should be noted that a master/slave type of architecture is also supported by having one of the SC devices also act as an MSAS. In this case the MSAS functionality is thus embedded in an RTP receiver instead of an RTP sender.

```
+-----------------------+        +-----------------------+
|                       |  SR +  |                       |
|      RTP Receiver     |  RTCP  |      RTP Sender       |
|                       |  IDMS  |                       |
|  +-----------------+  | <----- |  +-----------------+  |
|  |                 |  |        |  |                 |  |
|  | Synchronization |  |        |  |      Media      |  |
|  |     Client      |  |        |  | Synchronization |  |
|  |      (SC)       |  |        |  |   Application   |  |
|  |                 |  |        |  |      Server     |  |
|  |                 |  | RR+XR  |  |      (MSAS)     |  |
|  |                 |  | -----> |  |                 |  |
|  +-----------------+  |        |  +-----------------+  |
|                       |        |                       |
+-----------------------+        +-----------------------+
```

5.1. Media Synchronization Application Server (MSAS)

An MSAS collects RTP packet arrival times and play-out times from one or more SC(s) in a synchronization group. The MSAS summarizes and distributes this information to the SCs in the synchronization group as synchronization settings, e.g. by determining the SC with the most lagged play-out and using its reported RTP packet arrival time and play-out time as a summary.

5.2. Synchronization Client (SC)

An SC reports RTP packet arrival times and play-out times of a media stream. It can receive summaries of such information, and use that to adjust its play-out buffer.

5.3. Communication between MSAS and SCs

Two different message types are used for the communication between MSAS and SCs. For the SC->MSAS message containing the play-out information of a particular client, an RTCP XR Block Type is used (see Section 6). For the MSAS->SC message containing the synchronization settings instructions, a new RTCP Packet Type is defined in Section 7.

6. RTCP XR Block Type for IDMS

This section describes the RTCP XR Block Type for reporting IDMS information on an RTP media stream. Its definition is based on [RFC3611]. The RTCP XR is used to provide feedback information on receipt times and presentation times of RTP packets to e.g. a Sender [RFC3611], a Feedback Target [RFC5760] or a Third Party Monitor [RFC3611].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P| Resrv   |   PT=XR=207   |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SSRC of packet sender                     |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|     BT=12     | SPST  |Resrv|P|         block length=7        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     PT      |                 Resrv                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Media Stream Correlation Identifier              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SSRC of media source                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Packet Received NTP timestamp, most significant word     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Packet Received NTP timestamp, least significant word    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Packet Received RTP timestamp                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Packet Presented NTP timestamp                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The first 64 bits form the header of the RTCP XR, as defined in [RFC3611]. The SSRC of packet sender identifies the sender of the specific RTCP packet.

The IDMS report block consists of 7 32-bit words, with the following fields:

Block Type (BT): 8 bits. It identifies the block format. Its value SHALL be set to 12.

Synchronization Packet Sender Type (SPST): 4 bits. This field identifies the role of the packet sender for this specific eXtended Report. It can have the following values:

SPST=0 Reserved For future use.

SPST=1 The packet sender is an SC. It uses this XR to report synchronization status information. Timestamps relate to the SC input.

SPST=2 This setting is reserved in order to preserve compatibility with ETSI TISPAN [TS 183 063]. See section 12. for more information.

SPST=3-15 Reserved For future use.

Reserved bits (Resrv): 3 bits. These bits are reserved for future definition. In the absence of such a definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

Packet Presented NTP timestamp flag (P): 1 bit. Bit set to 1 if the Packet Presented NTP timestamp field contains a value, 0 if it is empty. If this flag is set to zero, then the Packet Presented NTP timestamp shall not be inspected.

Block Length: 16 bits. This field indicates the length of the block in 32 bit words and shall be set to 7, as this RTCP Block Type has a fixed length.

Payload Type (PT): 7 bits. This field identifies the format of the media payload, according to [RFC3551]. The media payload is associated with an RTP timestamp clock rate. This clock rate provides the time base for the RTP timestamp counter.

Reserved bits (Resrv): 25 bits. These bits are reserved for future use and shall be set to 0.

Media Stream Correlation Identifier: 32 bits. This identifier is used to correlate synchronized media streams. The value 0 (all bits are set "0") indicates that this field is empty. The value 2^32-1 (all bits are set "1") is reserved for future use. If the RTCP Packet Sender is an SC (SPST=1), then the Media Stream Correlation Identifier maps on the SyncGroupId to which the SC belongs.

SSRC: 32 bits. The SSRC of the media source shall be set to the value of the SSRC identifier carried in the RTP header [RFC3550] of the RTP packet to which the XR relates.

Packet Received NTP timestamp: 64 bits. This timestamp reflects the wall clock time at the moment of arrival of the first octet of the RTP packet to which the XR relates. It is formatted based on the NTP timestamp format as specified in [RFC5905]. See section 8 for more information on how this field is set.

Packet Received RTP timestamp: 32 bits. This timestamp has the value of the RTP time stamp carried in the RTP header [RFC3550] of the RTP packet to which the XR relates.

Packet Presented NTP timestamp: 32 bits. This timestamp reflects the wall clock time at the moment the data contained in the first octet of the associated RTP packet is presented to the user. It is based on the time format used by NTP and consists of the least significant 16 bits of the NTP seconds part and the most significant 16 bits of the NTP fractional second part. If this field is empty, then it SHALL be set to 0 and the Packet Presented NTP timestamp flag (P) SHALL be set to 0.

7. RTCP Packet Type for IDMS (IDMS report)

This section specifies the RTCP Packet Type for indicating synchronization settings instructions to a receiver of the RTP media stream. Its definition is based on [RFC3550].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P| Resrv   |     PT=TBD    |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SSRC of packet sender                     |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                     SSRC of media source                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Media Stream Correlation Identifier              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Packet Received NTP timestamp, most significant word     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Packet Received NTP timestamp, least significant word    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Packet Received RTP timestamp                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Packet Presented NTP timestamp                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The first 64 bits form the header of the RTCP Packet Type, as defined in [RFC3550]. The SSRC of packet sender identifies the sender of the specific RTCP packet.

The RTCP IDMS packet consists of 6 32-bit words, with the following fields:

SSRC: 32 bits. The SSRC of the media source shall be set to the value of the SSRC identifier carried in the RTP header [RFC3550] of the RTP packet to which the RTCP IDMS packet relates.

Media Stream Correlation Identifier: 32 bits. This identifier is used to correlate synchronized media streams. The value 0 (all bits are set "0") indicates that this field is empty. The value 2^32-1 (all bits are set "1") is reserved for future use. The Media Stream Correlation Identifier maps on the SyncGroupId of the group to which this packet is sent.

Packet Received NTP timestamp: 64 bits. This timestamp reflects the wall clock time at the reference client at the moment it received the first octet of the RTP packet to which this packet relates. It can be used by the synchronization algorithm on the receiving SC to set the required playout delay. The timestamp is formatted based on the NTP timestamp format as specified in [RFC5905]. See section 8 for more information on how this field is set.

Packet Received RTP timestamp: 32 bits. This timestamp has the value of the RTP time stamp carried in the RTP header [RFC3550] of the RTP packet to which the XR relates.

Packet Presented NTP timestamp: 32 bits. This timestamp reflects the wall clock time at the reference client at the moment it presented the data contained in the first octet of the associated RTP packet to the user. It is based on the time format used by NTP and consists of the least significant 16 bits of the NTP seconds part and the most significant 16 bits of the NTP fractional second part. If this field is empty, then it SHALL be set to 0. This field MAY be left empty if none or only one of the receivers reported on presentation timestamps.

8. Timing and NTP Considerations

To achieve IDMS, the different receivers involved need synchronized clocks as a common timeline for synchronization. Depending on the synchronization accuracy required, different clock synchronization methods can be used. For social TV, synchronization accuracy should be achieved in order of hundreds of milliseconds. In that case, correct use of NTP on receivers will in most situations achieve the required accuracy. As a guideline, to deal with clock drift of receivers, receivers should synchronize their clocks at the beginning of a synchronized session.

IDMS may be used for other purposes, such as synchronization of multiple television outputs in a single physical location, or for the synchronization of different networked speakers throughout a house.

Because of the stringent synchronization requirements for achieving good audio, a high accuracy will be needed. In this case, NTP usage may not be sufficient. Either a local NTP server could be setup, or some other more accurate clock synchronization mechanism could be used, such as using GPS time or the Precision Time Protocol [IEEE-1588].

In this document, a new SDP parameter is introduced to signal the clock synchronization source or sources used or able to be used (see section 10). An SC can indicate which synchronization source is being used at the moment and the last time the SC synchronized with this source. An SC can also indicate any other synchronization sources available to it. This allows multiple SCs in an IDMS session to use the same or a similar clock synchronization source for their session.

Applications performing IDMS may or may not be able to choose a synchronization method for the system clock. How applications deal with this is up to the implementation. The application might control the system clock, or it might use a separate application clock or even a separate IDMS session clock. It might also report on the system clock and the synchronization method used, without being able to change it.

9. SDP Parameter for RTCP XR IDMS Block Type

The SDP parameter sync-group is used to signal the use of the RTCP XR block for inter-destination media synchronization. It is also used to carry an identifier for the synchronization group to which clients belong or will belong. This SDP parameter extends rtcp-xr-attrib as follows, using Augmented Backus-Naur Form [RFC5234].

```
rtcp-xr-attrib = "a=" "rtcp-xr" ":" [xr-format *(SP xr-format)] CRLF
; Original definition from [RFC3611], section 5.1

xr-format =/ grp-sync ; Extending xr-format for inter-destination
media synchronization

grp-sync = "grp-sync" [",sync-group=" SyncGroupId]

SyncGroupId = 1*DIGIT ; Numerical value from 0 till 4294967295

DIGIT = %x30-39
```

SyncGroupId is a 32-bit unsigned integer in network byte order and represented in decimal. SyncGroupId identifies a group of SCs for IDMS. It maps on the Media Stream Correlation Identifier as described in sections 6 and 7. The value SyncGroupId=0 represents an empty

SyncGroupId. The value 4294967295 (2^32-1) is reserved for future use.

The following is an example of the SDP attribute for IDMS

```
a=rtcp-xr:grp-sync,sync-group=42
```

10. SDP Parameter for RTCP IDMS Packet Type

The SDP parameter rtcp-idms is used to signal the use of the RTCP IDMS Packet Type for IDMS. It is also used to carry an identifier for the synchronization group to which clients belong or will belong. The SDP parameter is used as a media-level attribute during session setup. This SDP parameter is defined as follows, using Augmented Backus-Naur Form [RFC5234].

```
rtcp-idms  = "a=" "rtcp-idms" ":" [sync-grp] CRLF

sync-grp   = "sync-group=" SyncGroupId

SyncGroupId = 1*DIGIT ; Numerical value from 0 till 4294967295

DIGIT     = %x30-39
```

SyncGroupId is a 32-bit unsigned integer in network byte order and represented in decimal. SyncGroupId identifies a group of SCs for IDMS. The value SyncGroupId=0 represents an empty SyncGroupId. The value 4294967295 (2^32-1) is reserved for future use.

The following is an example of the SDP attribute for IDMS.

```
a=rtcp-idms:sync-group=42
```

11. SDP parameter for clock source

The SDP parameter clocksource is used to signal the source for clock synchronization. This SDP parameter is specified as follows, using Augmented Backus-Naur Form [RFC5234].

```
clocksource = "a=" "clocksource" ":" source SP [last-synced] CRLF

source      = local / ntp / gps / gal / ptp

local       = "local"

ntp        = "ntp" ["=" ntp-server]

ntp-server     =  host [ ":" port ]
```

```
host           =  hostname / IPv4address / IPv6reference

hostname       =  *( domainlabel "." ) toplabel [ "." ]

domainlabel    =  alphanum

                / alphanum *( alphanum / "-" ) alphanum

toplabel       =  ALPHA / ALPHA *( alphanum / "-" ) alphanum

IPv4address    =  1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT

IPv6reference  =  "[" IPv6address "]"

IPv6address    =  hexpart [ ":" IPv4address ]

hexpart        =  hexseq / hexseq "::" [ hexseq ] / "::" [ hexseq ]

hexseq         =  hex4 *( ":" hex4)

hex4           =  1*4HEXDIG

port           =  1*DIGIT

gps        = "gps"

gal        = "gal"

ptp        = "ptp" ["=" ptp-id]

ptp-id      = 1*alphanum

last-synced   = date SP time

date          =  2DIGIT "-" 2DIGIT "-" 4DIGIT

                 ; day month year (e.g., 02-06-1982)

time           =  2DIGIT ":" 2DIGIT ":" 2DIGIT

                 ; 00:00:00 - 23:59:59

alphanum     =  ALPHA / DIGIT

EXAMPLE

a=clocksource:ntp=139.63.192.5:123 19-02-2011 21:03:20
```

A client MAY include this attribute multiple times. If multiple time synchronization sources were used in the past, the client MUST only report the 'last synced' parameter on the latest synchronization performed. If a client supports a specific synchronization method, but does not know any sources to use for synchronization, it SHOULD indicate the method without specifying the source. A client MAY indicate itself as source if it is a clock synchronization source, but it SHOULD do so using a publicly reachable address.

The parameter can be used as both a session or media level attribute. It will normally be a session level parameter, since it is not directly media-related. In case of IDMS however, it can be used in conjunction with the rtcp-idms SDP parameter, and then it SHOULD be used as a media-level parameter as well.

The meaning of 'local' is that no clock synchronization is performed.

The 'last synced' parameter is used as an indication for the receiver of the parameter on the accuracy of the clock. If the indicated last synchronization time is very recent, this is an indication that the clock can be trusted to be accurate, given the method of clock synchronization used. If the indicated last synchronization time is longer ago or in the future, either the clock synchronization has been performed long ago, or the clock is synchronized to an incorrect synchronization source. Either way, this shows that the clock used can not be trusted to be accurate.

12. Compatibility with ETSI TISPAN

As described in section 1.4, ETSI TISPAN has also described a mechanism for IDMS in [TS 183 063]. One of the main differences between the TISPAN document and this document is the fact that the TISPAN solution uses an RTPC XR block for both the SC->MSAS message as well as for the MSAS->SC message (by selecting different SPST-types), while this document specifies a new RTCP Packet Type for the MSAS->SC message.

In order to maintain backward-compatibility, the RTCP XR block used for SC->MSAS signaling specified in this document is fully compatible with the TISPAN defined XR block.

For the MSAS->SC signaling, it is recommended to use the RTCP IDMS Packet Type defined in this document. The TISPAN XR block with SPST=2 MAY be used for purposes of compatibility with the TISPAN solution, but MUST NOT be used if all nodes involved support the new RTCP IDMS Packet Type.

The above means that the IANA registry contains two SDP parameters for the MSAS->SC signaling; one for the ETSI TISPAN solution and one for the IETF solution. This also means that if all elements in the SDP negotiation support the IETF solution they SHOULD use the new RTCP IDMS Packet Type.

13. Operational Considerations

On Echo Cancellation:

In the case of social TV: If the two locations have a "side channel" audio conference so the viewers can talk about what they are watching, this may cause an audio problem that will not be solved by just applying IDMS. The audio output of the television of one viewer will pass through the audio conference, and arrive at the second viewer out of sync with the television output of that second viewer. Different methods can be used to deal with this effect, e.g. using directional microphones to prevent this or applying echo cancellation to filter out the unwanted audio signals.

On Reception vs. Presentation Timing:

A receiver can report on different timing events, i.e. on packet arrival times and on playout times. A receiver SHALL report on arrival times and a receiver MAY report on playout times. RTP packet arrival times are relatively easy to report on. Normally, the processing and play-out of the same media stream by different receivers will take roughly the same amount of time. By synchronizing on packet arrival times, you may loose some accuracy, but it will be adequate for many applications, such as social TV. Also, if the receivers are in some way controlled, e.g. having the same buffer settings and decoding times, high accuracy can be achieved. However, if all receivers in a synchronization session have the ability to report on, and thus synchronize on, actual playout times, or packet presentation times, this may be more accurate. It is up to applications and implementations of this RTCP extension whether to implement and use this.

14. Security Considerations

The specified RTCP XR Block Type in this document is used to collect, summarize and distribute information on packet reception- and playout -times of streaming media. The information may be used to orchestrate the media play-out at multiple devices.

Errors in the information, either accidental or malicious, may lead to undesired behavior. For example, if one device erroneously reports a two-hour delayed play-out, then another device in the same synchronization group could decide to delay its play-out by two hours as well, in order to keep its play-out synchronized. A user would likely interpret this two hour delay as a malfunctioning service.

Therefore, the application logic of both Synchronization Clients and Media Synchronization Application Servers should check for inconsistent information. Differences in play-out time exceeding configured limits (e.g. more than ten seconds) could be an indication of such inconsistent information.

No new mechanisms are introduced in this document to ensure confidentiality. Encryption procedures, such as those being suggested for a Secure RTP (SRTP) at the time that this document was written, can be used when confidentiality is a concern to end hosts.

15. IANA Considerations

New RTCP Packet Types and RTCP XR Block Types are subject to IANA registration. For general guidelines on IANA considerations for RTCP XR, refer to [RFC3611].

[TS 183 063] assigns the block type value 12 in the RTCP XR Block Type Registry to "Inter-destination Media Synchronization Block". [TS 183 063] also registers the SDP [RFC4566] parameter "grp-sync" for the "rtcp-xr" attribute in the RTCP XR SDP Parameters Registry.

Further, this document defines a new RTCP packet type called IDMS report. This new packet type is registered with the IANA registry of RTP parameters, based on the specification in section 7.

Further, this document defines a new SDP parameter "rtcp-idms" within the existing IANA registry of SDP Parameters.

The SDP attribute "rtcp-idms" defined by this document is registered with the IANA registry of SDP Parameters as follows:

SDP Attribute ("att-field"):

Attribute name: rtcp-idms

Long form: RTCP report block for IDMS

Type of name: att-field

Type of attribute: media level

Subject to charset: no

Purpose: see sections 7 and 10 of this document

Reference: this document

Values: see this document

Further, this document defines a new SDP attribute, "clocksource", within the existing IANA registry of SDP Parameters.

The SDP attribute "clocksource" defined by this document is registered with the IANA registry of SDP Parameters as follows:

SDP Attribute ("att-field"):

Attribute name: clocksource

Long form: clock synchronization source

Type of name: att-field

Type of attribute: session level

Subject to charset: no

Purpose: see sections 8 and 11 of this document

Reference: this document

Values: see this document and registrations below

The attribute has an extensible parameter field and therefore a registry for these parameters is required. This document creates an IANA registry called the Clocksource Source Parameters Registry. It contains the five parameters defined in Section 11: "local", "ntp", "gps", "gal" and "ptp".

16. Conclusions

This document describes the RTCP XR block type for IDMS, the RTCP IDMS report and the associated SDP parameters for inter-destination media synchronization. It also describes an SDP parameter for indicating which source is used for synchronizing a (systems) (wall) clock.

17. References

17.1. Normative References

[RFC5234] Crocker, D. and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 5234, January 2008.

[RFC3550] Schulzrinne, H., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

[RFC3551] Schulzrinne, H. and Casner S., "RTP Profile for Audio and Video Conferences with Minimal Control", RFC 3551, July 2003

[RFC3611] Friedman, T. "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC4566] Handley, M., "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC5576] Lennox, J., "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, June 2009.

[RFC5905] Mills, D., "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC5968] Ott, J., Guidelines on Extending the RTP Control Protocol (RTCP), September 2010.

[TS 183 063] ETSI TISPAN, "IMS-based IPTV stage 3 specification", TS 183 063 v3.4.1, June 2010.

17.2. Informative References

[Boronat2009] Boronat, F., et al, "Multimedia group and inter-stream synchronization techniques: A comparative study", Elsevier Information Systems 34 (2009), pp. 108-131

[IEEE-1588] IEEE Standards Association, "1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", 2008

Authors' Addresses

Ray van Brandenburg
TNO
Brassersplein 2, Delft, the Netherlands

Phone: +31 88 86 63609
Email: ray.vanbrandenburg@tno.nl

Hans M. Stokking
TNO
Brassersplein 2, Delft, the Netherlands

Phone: +31 88 86 67278
Email: hans.stokking@tno.nl

M. Oskar van Deventer
TNO
Brassersplein 2, Delft, the Netherlands

Phone: +31 88 86 67078
Email: oskar.vandeventer@tno.nl

Omar A. Niamut
TNO
Brassersplein 2, Delft, the Netherlands

Phone: +31 88 86 67218
Email: omar.niamut@tno.nl

Fabian A. Walraven
TNO
Brassersplein 2, Delft, the Netherlands

Phone: +31 88 86 67722
Email: fabian.walraven@tno.nl

Ishan Vaishnavi
CWI
Science Park 123, Amsterdam, the Netherlands

Phone: +31 20 592 4323
Email: i.vaishnavi@cwi.nl

Fernando Boronat
IGIC Institute, Universidad Politecnica de Valencia-Campus de Gandia
C/ Paraninfo, 1, Grao de Gandia, 46730, Valencia, Spain

Phone: +34 962 849 341
Email: fboronat@dcom.upv.es

Mario Montagud
IGIC Institute, Universidad Politecnica de Valencia-Campus de Gandia
C/ Paraninfo, 1, Grao de Gandia, 46730, Valencia, Spain

Phone: +34 962 849 341
Email: mamontor@posgrado.upv.es

Audio/Video Transport Working Group   G. Hunt
Internet-Draft   P. Arden
Intended status: Informational   BT
Expires: September 15, 2011   Q. Wu, Ed.
                                                                Huawei
                                                        March 14, 2011

Monitoring Architectures for RTP
draft-hunt-avtcore-monarch-01.txt

Abstract

This memo proposes an architecture for extending RTCP with a new RTCP XR (RFC3611) block type to report new metrics regarding media transmission or reception quality, as proposed in RFC5968. This memo suggests that a new block should contain a single metric or a small number of metrics relevant to a single parameter of interest or concern, rather than containing a number of metrics which attempt to provide full coverage of all those parameters of concern to a specific application. Applications may then "mix and match" to create a set of blocks which covers their set of concerns. Where possible, a specific block should be designed to be re-usable across more than one application, for example, for all of voice, streaming audio and video.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

Service providers and network providers today suffer from lack of good service that can monitor the performance at the user’s home, handset or remote office. Without service performance metrics, it is difficult for network operators to properly locate the problem and solve service issues before problems impact subscriber/end user. The resolution generally involves deploying costly field network technician to conduct on-site troubleshooting and diagnostics. By reducing the expensive deployments with more automated remote monitoring capabilities, network operators can save significant costs, reduce mean time to repair and provider a better service offering.

As more users and subscribers rely on real time application services, uncertainties in the performance and availability of these services are driving the need to support new standard methods for gathering performance metrics from RTP applications. These rapidly emerging standards, such as RTCP XR [RFC3611]and other RTCP extension to Sender Reports(SR), Receiver Reports (RR) [RFC3550]are being developed for the purpose of collecting and reporting performance metrics from endpoint devices that can be used to correlate the metrics, provide end to end service visibility and measure and monitor QoE.

However the proliferation of RTP/RTCP specific metrics for transport and application quality monitoring has been identified as a potential problem for RTP/RTCP interoperability, which attempt to provide full coverage of all those parameters of concern to a specific application. Since different applications layered on RTP may have some monitoring requirements in common, therefore these metrics should be satisfied by a common design.

The objective of this document is to define an extensible RTP monitoring framework to provide a small number of re-usable QoS/QoE metrics which facilitate reduced implementation costs and help maximize inter-operability. [RFC5968] has stated that, where RTCP is to be extended with a new metric, the preferred mechanism is by the addition of a new RTCP XR [RFC3611] block. This memo assumes that any requirement for a new metric to be transported in RTCP will use a new RTCP XR block.

2. Requirements notation

This memo is informative and as such contains no normative requirements.

3. RTP monitoring architecture

The RTP monitoring architecture comprises the following three functional components shown below:

- o Real Time Application Quality Monitoring (RAQMON) Report Wrapper
- o Real Time Application Quality Monitoring (RAQMON) Report Collector
- o Real Time Application Quality Monitoring (RAQMON) Metric Block Structure

RAQMON Report Wrapper (RRW) is a functional component that acts as a source of information gathered for monitoring purposes. It also can be referred to as "Monitoring Client". The end system that source RTP streams, or an intermediate-system that forwards RTP packets to End-devices can be envisioned to act as RRWs within the RTP monitoring architecture.

A RAQMON Report Collector (RRC) is a functional component that act as monitoring server or monitoring center. It collects statistics from multiple RRWs, analyzes them, stores such information reported by RTCP XR or other RTCP extension appropriately as base metric or calculates composite metric. RRC is envisioned to be a middleware like RTP translator, Multipoint Conferencing Bridge, Distribution Source, serving an administrative domain defined by the network administrator.

The RAQMON Metric Block exposes real time Application Quality information in the report block format to RRC and Network Management Applications. The RTCP or RTCP XR can be extended to convey such information to accommodate the RTP monitoring architecture.

```
       +-------------------+
       | RTP Sender        |
       | +--------------+  |
       | |RAQMON Report |--- --------|
       | |Wrapper       |  |         |
       | +--------------+  |         |
       |+-----------------+|         |        +-----------+
       ||Application      ||         |        |Management |
       ||-streaming video ||         |        |Application|
    |---|-VOIP            ||         |        +------\----+
    |  ||-video conference||      5  |               |
    |  ||-telepresence    ||         |               |6
    |  ||-ad insertion    ||         |       +-------|-----+
    |  |+-----------------+|         ------->|RAQMON Report|
    |  +-------------------+         ------->|  Collector  |
    |               Report Block     |       +----------\--+
    |               transported over |  Report Block    |
    |               RTCP extension   |  transported over|5
    | 1                              |  RTCP XR         |
    | +------ -----------------+     |   +--------------|---- ----+
    | | RTP System             |     |   | RTP Receiver >--4-|--- |
    | | +--------------+       |  5  |   | +--------------+  |    |
    | | |RAQMON Report --------------|   | |RAQMON Report |<--    |
    | | |  Wrapper     |       |         | |  Wrapper     |<------|
    | | +--------------+       |         | +-------\------+      ||
    | |                        |         |         |             ||
    | |                        |         |         |2            ||
    | | +-----------------+    |         | +-------/---------+   ||
    | | |Application      |    |         | |Application      |   ||
    | | |-streaming video |    |         | |-streaming video |   ||
    | | |-VOIP            |    |    1    | |-VOIP            |   3|
    ---->-Video conference|---------------->|-Video conference    ||
      | |-Telepresence    |    |         | |-Telepresence    |   ||
      | |-Ad insertion    |    |         | |-Ad insertion    |   ||
      | +-----------------+    |         | +-----------------+   ||
      | +-----------------+    |         | +-----------------+   ||
      | |Transport        |    |         | |Transport        |   ||
      | |-IP/UDP/RTP      |    |         | |-IP/UDP/RTP      >---||
      | |-IP/TCP/RTP      |    |         | | -IP/TCP/RTP     |    |
      | |-IP/TCP/RTSP/RTP |    |         | |-IP/TCP/RTSP/RTP |    |
      | +-----------------+    |         | +-----------------+    |
      +------------------------+         +------------------------+
```

Figure 1: RTP Monitoring Architecture

1. RTP communication between real time applications

2. Application layer metrics

3. Transport layer metrics

4. End System metrics

5. Reporting Session- metrics transmitted over specified interfaces

6. Management application- RRC interaction using northbound interface. - RRC outputs reports to the management application. The management application collects raw data from RRC, organizes database, conducts data analysis and creates alerts to the users.

4. RTCP Metric Block Report and associated parameters

The basic RTCP Reception Report (RR) conveys reception statistics in metric block report format for multiple RTP media streams including

- o transport level statistics
- o the fraction of packet lost since the last report
- o the cumulative number of packets lost
- o the highest sequence number received
- o an estimate of the inter-arrival jitter
- o and information to allow senders to calculate the network round trip time.

The RTCP XRs [RFC3611] supplement the existing RTCP packets and provide more detailed feedback on reception quality in several categories:

- o Loss and duplicate RLE reports
- o Packet-receipt times reports
- o Round-trip time reports
- o Statistics Summary Reports

There are also various other scenarios in which it is desirable to send RTCP Metric reports more frequently. The Audio/Video Profile with Feedback [RFC4585]extends the standard A/V Profile[RFC3551] to allow RTCP reports to be sent early provided RTCP bandwidth allocation is respected. There are four use cases but are not limited to:

- o RTCP NACK is used to provide feedback on the RTP sequence number of the lost packets.
- o RTCP XR is extended to provide feedback on multicast acquisition statistics information and parameters.
- o RTCP is extended to convey requests for full intra-coded frames or select the reference picture, and signalchanges in the desired temporal/spatial trade-off and maximum media bit rate.

- o RTCP or RTCP XR is extended to provide feedback on ECN statistics information.

4.1. Classification of RTCP Metric Block parameters

4.1.1. Application level parameters

Measured data at the application level, i.e., QoE related parameters which focus on quality of content rather than network parameters. These include but are not limited to:

- o Sound/Noise Level
- o Echo return lost
- o Statistics Summary Info, e.g.,key frame lost key frame lost rate/ discard rate, key frame burst severity
- o Codec Control
- o Estimated Mean Opinion Score (MOS)

4.1.2. Transport level parameters

Measured data at the transport level. These include but are not limited to:

- o Lost packets
- o Round trip delay
- o Jitter
- o Congestion info
- o FEC
- o Codec Control
- o Media Synchronization info
- o Retransmission Info
- o RAMS info

4.1.3. End system parameters

Measured data from application residing in that device. These include but are not limited to:

- Error Concealment
- FEC
- Media Synchronization info
- Jitter Buffer Lost
- Jitter Buffer Delay

5. Monitoring Methodology

5.1. Option 1 - Monitoring every packet

The aim of "monitoring every packet" is to ensure that the information reported is not dependent on the application. In this scheme, RTP systems will report arrival data for each individual RTP packet. RTP (or other) systems receiving this "raw" data may use it to calculate any preferred heuristic metrics, but such calculations and the reporting of the results (e.g. to a session control layer or a management layer) are outside the scope of RTP and RTCP.

5.2. Option 2 - Real-time histogram methods

There are several potentially useful metrics which rely on the accumulation of a histogram in real time, so that a packet arrival results in a counter being incremented rather than in the creation of a new data item. These metrics may be gathered with a low and predictable storage requirement. Each counter corresponds to a single class interval or "bin" of the histogram. Examples of metrics which may be accumulated in this way include the observed distribution of packet delay variation, and the number of packets lost per unit time interval.

Different networks may have very different expected and achieved levels of performance, but it may be useful to fix the number of class intervals in the reported histogram to give a predictable volume of data. This can be achieved by starting with small class intervals ("bin widths") and automatically increasing the width (e.g. by factors of two) if outliers are seen beyond the current upper limit of the histogram. Data already accumulated may be assigned unambiguously to the new set of bins, given some simple conditions on the relationship between the old and new origins and bin widths.

A significant disadvantage of the histogram method is the loss of any information about time-domain correlations between the samples which build the histogram. For example, a histogram of packet delay variation provides no indication of whether successive samples of packet delay variation were uncorrelated, or alternatively that the packet delay variation showed a highly-correlated low-frequency wander.

5.3. Option 3 - Monitoring by exception

An entity which both monitors the packet stream, and has sufficient knowledge of the application to know when transport impairments may have degraded the application's performance, may choose to send exception reports containing details of the transport impairments to a receiving system. The crossing of a transport impairment threshold, or some application-layer event, would trigger such reports. RTP end systems and mixers are likely to contain application implementations which may, in principle, identify this type of exception.

It is likely that RTP translators will not contain suitable implementations which could identify such exceptions.

On-path devices such as routers and switches are not likely to be aware of RTP at all. Even if they are aware of RTP, they are unlikely to be aware of the RTP-level performance required by specific applications, and hence they are unlikely to be able to identify the level of impairment at which exceptional transport conditions may start to affect application performance.

This type of monitoring typically requires the storage of recent data in a FIFO (e.g. a circular buffer) so that data relevant to the period just before and just after the exception may be reported. It is not usually helpful to report transport data only from the period following an exception event detected by an application. This imposes some storage requirement (though less than needed for Option 1). It also implies the existence of additional cross-layer primitives or APIs to trigger the transport layer to generate and send its exception report. Such a capability might be considered architecturally undesirable, in that it complicates one or more interfaces above the RTP layer.

5.4. Option 4 - Application-specific monitoring

This is a business-as-usual option which suggests that the current approach should not be changed, based on the idea that previous application-specific approaches such as that of [RFC3611] were valid. If a large category of RTP applications (such as VoIP) has a requirement for a unique set of transport metrics, arising from its different requirements of the transport, then it seems reasonable for each application category to define its preferred set of metrics to describe transport impairments. We expect that there will be few such categories, probably less than 10.

It may be easier to achieve interworking for a well-defined set of application-specific metrics than it would be in the case that applications select a profile from a palette of many independent re-usable metrics.

6. Issues with RTCP XR extension

Issues that have come up in the past with extensions to RTCP or RTCP XR include (but are probably not limited to) the following:

- RFC 3611 [RFC3611] defines seven report block formats for network management and quality monitoring. However some of these block types defined in [RFC3611]are only specifically designed for conveying multicast inference of network characteristics(MINC) or voice over IP (VoIP) monitoring.

- Designing a single report block or metric containing a large number of parameters in different classes for a specific application may increase implementation cost and minimize interoperability.

- The RTCP XR block namespace is limited by the 8-bit block type field in the RTCP XR header Under current allocation pressure, we expect that the RTCP XR Block Type space will be exhausted soon. We therefore need a way to extend the block type space, so that new specifications may continue to be developed.

7. Guideline for reporting block format using RTCP XR

7.1. Using small blocks

Different applications using RTP for media transport certainly have differing requirements for metrics transported in RTCP to support their operation. For many applications, the basic metrics for transport impairments provided in RTCP SR and RR packets [RFC3550] (together with source identification provided in RTCP SDES packets) are sufficient. For other applications additional metrics may be required or at least sufficiently useful to justify the overheads, both of processing in endpoints and of increased session bandwidth. For example an IPTV application using Forward Error Correction (FEC) might use either a metric of post-repair loss or a metric giving detailed information about pre-repair loss bursts to optimise payload bandwidth and the strength of FEC required for changing network conditions. However there are many metrics available. It is likely that different applications or classes of applications will wish to use different metrics. Any one application is likely to require metrics for more than one parameter but if this is the case, different applications will almost certainly require different combinations of metrics. If larger blocks are defined containing multiple metrics to address the needs of each application, it becomes likely that many different such larger blocks are defined, which becomes a danger to interoperability.

To avoid this pitfall, this memo proposes the use of small RTCP XR metrics blocks each containing a very small number of individual metrics characterising only one parameter of interest to an application running over RTP. For example, at the RTP transport layer, the parameter of interest might be packet delay variation, and specifically the metric "IPDV" defined by [Y1540]. See Section 8 for architectural considerations for a metrics block, using as an example a metrics block to report packet delay variation.

7.2. Sharing the identity block

Any measurement must be identified. However if metrics are delivered in small blocks there is a danger of inefficiency arising from repeating this information in a number of metrics blocks within the same RTCP packet, in cases where the same identification information applies to multiple metrics blocks.

An instance of a metric must be identified using information which is likely to include most of the following:

- the node at which it was measured,
- the source of the measured stream (for example, its CNAME),
- the SSRC of the measured stream,
- the sequence number of the first packet of the RTP session,
- the extended sequence numbers of the first packet of the current measurement interval, and the last packet included in the measurement,
- the duration of the most recent measurement interval and
- the duration of the interval applicable to cumulative measurements (which may be the duration of the RTP session to date).

[Editor’s note: this set of information overlaps with, but is more extensive than, that in the union of similar information in RTCP RR packets. Should we assume that RR information is always present if XR is sent, and that measurement intervals are exactly coincident? If so, state assumption and remove overlaps. What were the design considerations which led to the additional information *not* being present in RRs? The reason for the additional information here is the perceived difficulty of "locating" the *start* of the RTP session (sequence number of 1st packet, duration of interval applicable to cumulative measurements) using only RR. Is this a misconception? It leads to redundant information in this design because equivalent information is provided multiple times, once in *every* identification packet. Though this ensures immunity to packet loss, the design is ugly and the overhead is not completely trivial.]

This section proposes an approach to minimise the inefficiency of providing this identification information, assuming that an architecture based on small blocks means that a typical RTCP packet will contain more than one metrics block needing the same identification. The choice of identification information to be provided is discussed in [IDENTITY] (work in progress).

The approach is to define a stand-alone block containing only identification information, and to tag this identification block with a number which is unique within the scope of the containing RTCP XR packet. The "containing RTCP XR packet" is defined here as the RTCP XR header with PT=XR=207 defined in Section 2 of [RFC3611] and the associated payload defined by the length field of this RTCP XR header. The RTCP XR header itself includes the SSRC of the node at which all of the contained metrics were measured, hence this SSRC need not be repeated in the stand-alone identification block. A single containing RTCP XR packet may contain multiple identification blocks limited by the range of the tag field. Typically there will be one identification block per monitored source SSRC, but the use of more than one identification block for a single monitored source SSRC within a single containing RTCP XR packet is not ruled out.

There will be zero or more metrics blocks dependent on each identification block. The dependence of an instance of a metrics block on an identification block is established by the metrics block’s having the same numeric value of the tag field as its identification block (in the same containing RTCP XR packet).

Figure 2 below illustrates this principle using as an example an RTCP XR packet containing four metrics blocks, reporting on streams from two sources. The measurement identity information is provided in two blocks with Block Type NMI, and tag values 0 and 1 respectively.

Note: in this example, RTCP XR block type values for four proposed new block types (work in progress) are given as NMI, NPDV, NBGL and NDEL. These represent numeric block type codepoints to be allocated by IANA at the conclusion of the work.

Each of these two identity blocks will specify the SSRC of one of the monitored streams, as well as information about the span of the measurement. There are two metrics blocks with tag=0 indicating their association with the measurement identity block which also has tag=0. These are the two blocks following the identity block with tag=0, though this positioning is not mandatory. There are also two metrics blocks with tag=1 indicating their association with the measurement identity block which also has tag=1, and these are the two blocks following the identity block with tag=1.

[Editor’s note: if we mandated that metrics blocks associated with an identity block must always follow the identity block we could save the tag field and possibly simplify processing. Is this preferable to cross-referencing with a numeric tag?]

In the example, the block types of the metrics blocks associated with tag=0 are BT=NPDV (a PDV metrics block) and BT=NBGL (a burst and gap loss metrics block). The block types of the metrics blocks associated with tag=1 are BT=NPDV (a second PDV metrics block) and BT=NDEL (a delay metrics block). This illustrates that:

o multiple instances of the same metrics block may occur within a containing RTCP XR packet, associated with different identification information, and

o differing measurements may be made, and reported, for the
   different streams arriving at an RTP system.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|reserved |   PT=XR=207   |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of RTCP XR packet sender                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     BT=NMI    |0|tag=0| resv  |         block length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of stream source 1                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.      ...measurement identity information, source 1...         .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     BT=NPDV   |I|tag=0|pdvtyp |         block length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.               ...PDV information for source 1...              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    BT=NBGL    |I|tag=0| resv  |         block length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.            ...burst-gap-loss information for source 1...      .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     BT=NMI    |0|tag=1| resv  |         block length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of stream source 2                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.      ...measurement identity information, source 2...         .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     BT=NPDV   |I|tag=1|pdvtyp |         block length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.              ...PDV information for source 2...               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    BT=NDEL    |I|tag=1| resv  |         block length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.             ...delay information for source 2...              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: RTCP XR block with identity blocks

This approach of separating the identification information is more costly than providing identification in each metrics block if only a single metrics block is sent in an RTCP packet, but becomes beneficial as soon as more than one metrics block shares common identification.

7.3. Expanding the RTCP XR block namespace

[Editor's note: the RTCP XR block namespace is limited by the 8-bit block type field in the RTCP XR header (Section 3 of [RFC3611]). IESG have noted that this is potentially restrictive. It would be possible to standardise an expansion mechanism, probably based on use of a new field near the start of the variable-length "type-specific block contents" field. Clearly this could apply only to new block types, so might be standardised to apply to some subrange of the current 8-bit range, for example the range 128 through 191 might be used. At time of writing, block types 12 to 254 are unassigned and 255 is reserved for future expansion. Is there a consensus for, or against, work to allow expansion? One potential use is through hierarchical control, where one or a few codepoints at the top level are given to other SDOs who may then define a number of metrics distinguished by values in the (so far hypothetical) new field.]

8. An example of a metric block

This section uses the example of an existing proposed metrics block to illustrate the application of the principles set out in Section 7.1.

The example [PDV] (work in progress) is a block to convey information about packet delay variation (PDV) only, consistent with the principle that a metrics block should address only one parameter of interest. One simple metric of PDV is available in the RTCP RR packet as the "jit" field. There are other PDV metrics which may be more useful to certain applications. Two such metrics are the IPDV metric ([Y1540], [RFC3393]) and the MAPDV2 metric [G1020]. Use of these metrics is consistent with the principle in Section 5 of [RFC5968] that metrics should usually be defined elsewhere, so that RTCP standards define only the transport of the metric rather than its nature. The purpose of this section is to illustrate the architecure using the example of [PDV] (work in progress) rather than to document the design of the PDV metrics block or to provide a tutorial on PDV in general.

Given the availability of at least three metrics for PDV, there are design options for the allocation of metrics to RTCP XR blocks:

- provide an RTCP XR block per metric
- provide a single RTCP XR block which contains all three metrics
- provide a single RTCP block to convey any one of the three metrics, together with a identifier to inform the receiving RTP system of the specific metric being conveyed

In choosing between these options, extensibility is important, because additional metrics of PDV may well be standardised and require inclusion in this framework. The first option is extensible but only by use of additional RTCP XR blocks, which may consume the limited namespace for RTCP XR blocks at an unacceptable rate. The second option is not extensible, so could be rejected on that basis, but in any case a single application is quite unlikely to require transport of more than one metric for PDV. Hence the third option was chosen. This implies the creation of a subsidiary namespace to enumerate the PDV metrics which may be transported by this block, as discussed further in [PDV] (work in progress).

9. Application to RFC 5117 topologies

An RTP system (end system, mixer or translator) which originates, terminates or forwards RTCP XR blocks is expected to handle RTCP, including RTCP XR, as specified in [RFC3550] for that class of RTP systems. Provided this expectation is met, an RTP system using RTCP XR is architecturally no different from an RTP system of the same class (end system, mixer, or translator) which does not use RTCP XR. This statement applies to the topologies investigated in [RFC5117], where they use RTP end systems, RTP mixers and RTP translators as these classes are defined in [RFC3550].

These topologies are specifically Topo-Point-to-Point, Topo-Multicast, Topo-Translator (both variants, Topo-Trn-Translator and Topo-Media-Translator, and combinations of the two), and Topo-Mixer.

9.1. Applicability to MCU

The topologies based on systems which do not behave according to [RFC3550], that is Topo-Video-Switch-MCU and Topo-RTCP-terminating-MCU, suffer from the difficulties described in [RFC5117]. These difficulties apply to systems sending, and expecting to receive, RTCP XR blocks as much as to systems using other RTCP packet types. For example, a participant RTP end system may send media to a video switch MCU. If the media stream is not selected for forwarding by the switch, neither RTCP RR packets nor RTCP XR blocks referring to the end system's generated stream will be received at the RTP end system. Strictly the RTP end system can only conclude that its RTP has been lost in the network, though an RTP end system complying with the robustness principle of [RFC1122] should survive with essential functions unimpaired.

9.2. Application to translators

Section 7.2 of [RFC3550] describes processing of RTCP by translators. RTCP XR is within the scope of the recommendations of [RFC3550]. Some RTCP XR metrics blocks may usefully be measured at, and reported by, translators. As described in [RFC3550] this creates a requirement for the translator to allocate an SSRC for itself so that it may populate the SSRC in the RTCP XR packet header (although the translator is not a Synchronisation Source in the sense of originating RTP media packets). It must also supply this SSRC and the corresponding CNAME in RTCP SDES packets.

In RTP sessions where one or more translators generate any RTCP traffic towards their next-neighbour RTP system, other translators in the session have a choice as to whether they forward a translator's RTCP packets. Forwarding may provide additional information to other

RTP systems in the connection but increases RTCP bandwidth and may in some cases present a security risk. RTP translators may have forwarding behaviour based on local policy, which might differ between different interfaces of the same translator.

[Editor’s note: for bidirectional unicast, an RTP system may usually detect RTCP from a translator by noting that the sending SSRC is not present in any RTP media packet. However even for bidirectional unicast there is a possibility of a source sending RTCP before it has sent any RTP media (leading to transient mis-categorisation of an RTP end system or RTP mixer as a translator), and for multicast sessions - or unidirectional/streaming unicast - there is a possibility of a receive-only end system being permanently mis-categorised as a translator. Is there a need for a translator to declare itself explicitly? Needs further thought.]

10. IANA Considerations

None.

11. Security Considerations

This document itself contains no normative text and hence should not give rise to any new security considerations, to be confirmed.

12. Acknowledgement

The authors would like to thank Colin Perkins, Graeme Gibbs, Debbie Greenstreet, Keith Drage,Dan Romascanu, Ali C. Begen, Roni Even for their valuable comments and suggestions on the early version of this document.

13. Change Log

13.1. draft-hunt-avtcore-monarch-00

The following are the major changes compared to previous version 00:

- o Provide some background texts and related work into Introduction section.
- o Add a new section 3 to describe RTP monitoring architecture.
- o Add a new section 4 to describe RTCP Metric Block Report and associated parameters.
- o Move section 3.1, 3.2,3.3 and 3.4 in draft-hunt-avt-monarch-00 to this version as section 5 to describe Monitoring Methodology.
- o Add a new section 6 to describe Issues with RTCP XR extension.
- o Merge section 3,4, 8 in previous version into one new section 9 to describe Guideline for reporting block format using RTCP XR.
- o Merge section 6,7 in previous version into one new section 9 to describe Application to RFC 5117 topologies.

13.2. draft-hunt-avtcore-monarch-01

The following are the major changes compared to previous version 00:

- o Update figure 1 to describe the interface between RTP Sender and Report Collector in precise granularity.
- o Add some texts to define the role of RRW and RRC.
- o Correct the order of the second figure in the document.
- o Other editorial changes.

14. Informative References

[G1020] ITU-T, "ITU-T Rec. G.1020, Performance parameter definitions for quality of speech and other voiceband applications utilizing IP networks", July 2006.

[IDENTITY] Hunt, G., "RTCP XR Report Block for Measurement Identity", ID draft-ietf-avt-rtcp-xr-meas-identity-02, May 2009.

[PDV] Hunt, G., "RTCP XR Report Block for Packet Delay Variation Metric Reporting", ID draft-ietf-avt-rtcp-xr-pdv-03, May 2009.

[RFC1122] Braden, R., "Requirements for Internet Hosts -- Communication Layers", RFC 1122, October 1989.

[RFC3393] Demichelis, C., "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, November 2002.

[RFC3550] Schulzrinne, H., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

[RFC3551] Schulzrinne , H. and S. Casner, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 3551, July 2003.

[RFC3611] Friedman, T., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC4585] Ott, J. and S. Wenger, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

[RFC5117] Westerlund, M., "RTP Topologies", RFC 5117, January 2008.

[RFC5968] Ott, J. and C. Perkins, "Guidelines for Extending the RTP Control Protocol (RTCP)", RFC 5968, September 2010.

[Y1540] ITU-T, "ITU-T Rec. Y.1540, IP packet transfer and availability performance parameters", November 2007.

Authors' Addresses

Geoff Hunt
BT
Orion 1 PP2
Adastral Park
Martlesham Heath
Ipswich, Suffolk IP5 3RE
United Kingdom

Phone: +44 1473 651704
Email: geoff.hunt@bt.com

Philip Arden
BT
Orion 3/7 PP4
Adastral Park
Martlesham Heath
Ipswich, Suffolk IP5 3RE
United Kingdom

Phone: +44 1473 644192
Email: philip.arden@bt.com

Qin Wu (editor)
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: sunseawq@huawei.com

Network Working Group M. Westerlund
Internet-Draft I. Johansson
Intended status: Standards Track Ericsson
Expires: September 15, 2011 C. Perkins
University of Glasgow
P. O'Hanlon
UCL
K. Carlberg
G11
March 14, 2011

Explicit Congestion Notification (ECN) for RTP over UDP
draft-ietf-avtcore-ecn-for-rtp-01

Abstract

This document specifies how explicit congestion notification (ECN) can be used with Real-time Transport Protocol (RTP) over UDP flows that use RTP Control Protocol (RTCP) as feedback mechanism. It defines one RTP Control Protocol Extended Reports (RTCP XR) extension for ECN summary, a RTCP transport feedback format for timely reporting of congestion events, and an Session Traversal Utilities for NAT (STUN) extension used in the optional initilization method using Interactive Connectivity Establishment (ICE). Signalling and procedures for negotiation of capabilities and initilization methods are also defined.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

This document outlines how Explicit Congestion Notification (ECN) [RFC3168] can be used for Real-time Transport Protocol (RTP) [RFC3550] flows running over UDP/IP which use RTP Control Protocol (RTCP) as a feedback mechanism. The solution consists of feedback of ECN congestion experienced markings to the sender using RTCP, verification of ECN functionality end-to-end, and how to initiate ECN usage. The initiation process will have some dependencies on the signalling mechanism used to establish the RTP session, a specification for signalling mechanisms using Session Description Protocol (SDP) [RFC4566] is included.

ECN is getting attention as a method to minimise the impact of congestion on real-time multimedia traffic. When ECN is used, the network can signal to applications that congestion is occurring, whether that congestion is due to queuing at a congested link, limited resources and coverage on a radio link, or other reasons.

ECN provides a way for networks to send congestion control signals to a media transport without having to impair the media. Unlike losses, the signals unambiguously indicate congestion to the transport as quickly as feedback delays allow, and without confusing congestion with losses that might have occurred for other reasons such as transmission errors, packet-size errors, routing errors, badly implemented middleboxes, policy violations and so forth.

The introduction of ECN into the Internet requires changes to both the network and transport layers. At the network layer, IP forwarding has to be updated to allow routers to mark packets, rather than discarding them in times of congestion [RFC3168]. In addition, transport protocols have to be modified to inform the sender that ECN marked packets are being received, so it can respond to the congestion. TCP [RFC3168], SCTP [RFC4960] and DCCP [RFC4340] have been updated to support ECN, but to date there is no specification how UDP-based transports, such as RTP [RFC3550], can use ECN. This is due to the lack of feedback mechanisms directly in UDP. Instead the signaling control protocol on top of UDP needs to provide that feedback, which for RTP is RTCP.

The remainder of this memo is structured as follows. We start by describing the conventions, definitions and acronyms used in this memo in Section 2, and the design rationale and applicability in Section 3. Section 4 provides an overview of how ECN is used with RTP over UDP. Then the definition of the RTCP extensions for ECN feedback in Section 5. Then the SDP signalling extensions required are specified Section 6.Then the full details of how ECN is used with RTP over UDP is defined in Section 7. In Section 8 we discuss how

RTCP ECN feedback is handled in RTP translators and mixers. Section 9 discusses some implementation considerations, Section 10 lists IANA considerations, and Section 11 discusses the security considerations.

2. Conventions, Definitions and Acronyms

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

Abbreviations

- o ECN: Explicit Congestion Notification
- o ECT: ECN Capable Transport
- o ECN-CE: ECN Congestion Experienced
- o not-ECT: Not ECN Capable Transport

This document uses the terms sender and receiver according to the following definition:

Sender: Sender of RTP packets carrying an encoded media stream. The sender has the possibility to effect how this transmission is performed. It is one end-point of the ECN control loop.

Receiver: A receiver of RTP packets with the intention to consume the media stream in some form. It sends RTCP feedback on the received stream. It is the other end-point of the ECN control loop.

Note: RTP mixers or translators that operate in such a manner that they terminate or split the ECN control loop will take on the role of receivers or senders. This is further discussed in Section 3.2.

The meaning of the term ECN support depends on which entity between the sender and receiver (inclusive) that is considered. We distinguish between:

- o ECN-Capable Host: Sender or receiver of media.
- o ECN-Capable Transport: ECT = all ends are ECN capable hosts.

- o ECN-Capable Packets: Packets are either ECT or CE.

- o ECN-Oblivious Relay: Router or middlebox that treats ECN-Capable Packets no differently from Not-ECT.

- o ECN-Capable Queue: Supports ECN marking of ECN-Capable Packets.

- o ECN-Blocking Middlebox: Discards ECN-Capable Packets.

- o ECN-Reverting Middlebox: Changes ECN-Capable Packets to Not-ECT.

3. Discussion, Requirements, and Design Rationale

ECN has been specified for use with TCP [RFC3168], SCTP [RFC4960], and DCCP [RFC4340] transports. These are all unicast protocols which negotiate the use of ECN during the initial connection establishment handshake (supporting incremental deployment, and checking if ECN marked packets pass all middleboxes on the path). ECN Congestion Experienced (ECN-CE) marks are immediately echoed back to the sender by the receiving end-point using an additional bit in feedback messages, and the sender then interprets the mark as equivalent to a packet loss for congestion control purposes.

If RTP is run over TCP, SCTP, or DCCP, it can use the native ECN support provided by those protocols. This memo does not concern itself further with these use cases. However, RTP is more commonly run over UDP. This combination does not currently support ECN, and we observe that it has significant differences from the other transport protocols for which ECN has been specified. These include:

Signalling: RTP relies on separate signalling protocols to negotiate parameters before a session can be created, and doesn’t include an in-band handshake or negotiation at session set-up time (i.e. there is no equivalent to the TCP three-way handshake in RTP).

Feedback: RTP does not explicitly acknowledge receipt of datagrams. Instead, the RTP Control Protocol (RTCP) provides reception quality feedback, and other back channel communication, for RTP sessions. The feedback interval is generally on the order of seconds, rather than once per network RTT (although the RTP/AVPF profile [RFC4585] allows more rapid feedback in most cases).

Congestion Response: While it is possible to adapt the transmission of many audio/visual streams in response to network congestion, and such adaptation is required by [RFC3550], the dynamics of the congestion response may be quite different to those of TCP or other transport protocols.

Middleboxes: The RTP framework explicitly supports the concept of mixers and translators, which are middleboxes that are involved in media transport functions.

Multicast: RTP is explicitly a group communication protocol, and was designed from the start to support IP multicast (primarily ASM, although a recent extension supports SSM with unicast feedback [RFC5760]).

Application Awareness: ECN support via TCP, DCCP, and SCTP constrain the awareness and reaction to packet loss within those protocols. By adding support of ECN through RTCP, the application is made aware of packet loss and may choose one or more approaches in response to that loss.

Counting vs Detecting Congestion: TCP and the protocols derived from it are mainly designed to respond the same whether they experience a burst of congestion indications within one RTT or just one. Whereas real-time applications may be concerned with the amount of congestion experienced, whether it is distributed smoothly or in bursts. When feedback of ECN was added to TCP [RFC3168], the receiver was designed to flip the echo congestion experienced (ECE) flag to 1 for a whole RTT then flop it back to zero. Whereas ECN feedback in RTCP will need to report a count of how much congestion has been experienced within an RTCP reporting period, irrespective of round trip times.

These differences will significantly alter the shape of ECN support in RTP-over-UDP compared to ECN support in TCP, SCTP, and DCCP, but do not invalidate the need for ECN support.

ECN support is more important for RTP sessions than, for instance, is the case for TCP. This is because the impact of packet loss in real-time audio-visual media flows is highly visible to users. Effective ECN support for RTP flows running over UDP will allow real-time audio-visual applications to respond to the onset of congestion before routers are forced to drop packets, allowing those applications to control how they reduce their transmission rate, and hence media quality, rather than responding to, and trying to conceal the effects of unpredictable packet loss. Furthermore, widespread deployment for ECN and active queue management in routers, should it occur, can potentially reduce unnecessary queueing delays in routers, lowering the round-trip time and benefiting interactive applications of RTP, such as voice telephony.

3.1. Requirements

Considering ECN, transport protocols supporting ECN, and RTP based applications one can create a set of requirements that must be satisfied to at least some degree if ECN is to used by RTP over UDP.

- o REQ 1: A mechanism MUST negotiate and initiate the usage of ECN for RTP/UDP/IP sessions so that an RTP sender will not send packets with ECT in the IP header unless it knows all potential receivers will understand any CE indications they might receive.

- o REQ 2: A mechanism MUST feedback the reception of any packets that are ECN-CE marked to the packet sender

- o REQ 3: Provided mechanism SHOULD minimise the possibility for cheating

- o REQ 4: Some detection and fallback mechanism SHOULD exist to avoid loss of communication due to the attempted usage of ECN in case an intermediate node clears ECT or drops packets that are ECT marked.

- o REQ 5: Negotiation of ECN SHOULD NOT significantly increase the time taken to negotiate and set-up the RTP session (an extra RTT before the media can flow is unlikely to be acceptable for some use cases).

- o REQ 6: Negotiation of ECN SHOULD NOT cause media clipping at the start of a session.

The following sections describes how these requirements can be meet for RTP over UDP.

3.2. Applicability

The use of ECN with RTP over UDP is dependent on negotiation of ECN capability between the sender and receiver(s), and validation of ECN support in all elements of the network path(s) traversed. RTP is used in a heterogeneous range of network environments and topologies, with various different signalling protocols, all of which need to be verified to support ECN before it can be used.

Due to the need for each RTP sender that intended to use ECN with RTP to track all participants in the RTP session the sub-sampling of the group membership as specified by "Sampling of the Group Membership in RTP" [RFC2762] MUST NOT be used.

The usage of ECN is further dependent on a capability of the RTP media flow to react to congestion signalled by ECN marked packets.

Depending on the application, media codec, and network topology, this adaptation can occur in various forms and at various nodes. As an example, the sender can change the media encoding, or the receiver can change the subscription to a layered encoding, or either reaction can be accomplished by a transcoding middlebox. RFC 5117 identifies seven topologies in which RTP sessions may be configured, and which may affect the ability to use ECN:

Topo-Point-to-Point: This is a standard unicast flow. ECN may be used with RTP in this topology in an analogous manner to its use with other unicast transport protocols, with RTCP conveying ECN feedback messages.

Topo-Multicast: This is either an any source multicast (ASM) group [RFC3569] with potentially several active senders and multicast RTCP feedback, or a source specific multicast (SSM) group [RFC4607] with a single sender and unicast RTCP feedback from receivers. RTCP is designed to scale to large group sizes while avoiding feedback implosion (see Section 6.2 of [RFC3550], [RFC4585], and [RFC5760]), and can be used by a sender to determine if all its receivers, and the network paths to those receivers, support ECN (see Section 7.2). It is somewhat more difficult to determine if all network paths from all senders to all receivers support ECN. Accordingly, we allow ECN to be used by an RTP sender using multicast UDP provided the sender has verified that the paths to all known receivers support ECN, and irrespective of whether the paths from other senders to their receivers support ECN. "all its known receivers" are all the SSRCs that the RTP sender has received RTP or RTCP from the last five reporting intervals, i.e. they are not timed out. Note that group membership may change during the lifetime of a multicast RTP session, potentially introducing new receivers that are not ECN capable or have a path that doesn’t support ECN. Senders must use the mechanisms described in Section 7.4 to monitor that all receivers continue to support ECN, and they need to fallback to non-ECN use if any senders do not.

Topo-Translator: An RTP translator is an RTP-level middlebox that is invisible to the other participants in the RTP session (although it is usually visible in the associated signalling session). There are two types of RTP translator: those do not modify the media stream, and are concerned with transport parameters, for example a multicast to unicast gateway; and those that do modify the media stream, for example transcoding between different media codecs. A single RTP session traverses the translator, and the translator must rewrite RTCP messages passing through it to match the changes it makes to the RTP data packets. A legacy, ECN-unaware, RTP translator is expected to ignore the ECN bits on received packets, and to set the ECN bits to not-ECT when sending packets, so causing ECN negotiation on the path containing the translator to fail (any new RTP translator that does not wish to support ECN may do so similarly). An ECN aware RTP translator may act in one of three ways:

* If the translator does not modify the media stream, it should copy the ECN bits unchanged from the incoming to the outgoing datagrams, unless it is overloaded and experiencing congestion, in which case it may mark the outgoing datagrams with an ECN-CE mark. Such a translator passes RTCP feedback unchanged.

* If the translator modifies the media stream to combine or split RTP packets, but does not otherwise transcode the media, it must manage the ECN bits in a way analogous to that described in Section 5.3 of [RFC3168]: if an ECN marked packet is split into two, then both the outgoing packets must be ECN marked identically to the original; if several ECN marked packets are combined into one, the outgoing packet must be either ECN-CE marked or dropped if any of the incoming packets are ECN-CE marked. If the outgoing combined packet is not ECN-CE marked, then it MUST be ECT marked if any of the incoming packets were ECT marked. When RTCP ECN feedback packets (Section 5) are received, they must be rewritten to match the modifications made to the media stream (see Section 8.1).

* If the translator is a media transcoder, the output RTP media stream may have radically different characteristics than the input RTP media stream. Each side of the translator must then be considered as a separate transport connection, with its own ECN processing. This requires the translator interpose itself into the ECN negotiation process, effectively splitting the connection into two parts with their own negotiation. Once negotiation has been completed, the translator must generate RTCP ECN feedback back to the source based on its own reception, and must respond to RTCP ECN feedback received from the receiver(s) (see Section 8.2).

It is recognised that ECN and RTCP processing in an RTP translator that modifies the media stream is non-trivial.

Topo-Mixer: A mixer is an RTP-level middlebox that aggregates multiple RTP streams, mixing them together to generate a new RTP stream. The mixer is visible to the other participants in the RTP session, and is also usually visible in the associated signalling session. The RTP flows on each side of the mixer are treated independently for ECN purposes, with the mixer generating its own RTCP ECN feedback, and responding to ECN feedback for data it sends. Since connections are treated independently, it would seem reasonable to allow the transport on one side of the mixer to use ECN, while the transport on the other side of the mixer is not ECN capable, if this is desired.

Topo-Video-switch-MCU: A video switching MCU receives several RTP flows, but forwards only one of those flows onwards to the other participants at a time. The flow that is forwarded changes during the session, often based on voice activity. Since only a subset of the RTP packets generated by a sender are forwarded to the receivers, a video switching MCU can break ECN negotiation (the success of the ECN negotiation may depend on the voice activity of the participant at the instant the negotiation takes place - shout if you want ECN). It also breaks congestion feedback and response, since RTP packets are dropped by the MCU depending on voice activity rather than network congestion. This topology is widely used in legacy products, but is NOT RECOMMENDED for new implementations and cannot be used with ECN.

Topo-RTCP-terminating-MCU: In this scenario, each participant runs an RTP point-to-point session between itself and the MCU. Each of these sessions is treated independently for the purposes of ECN and RTCP feedback, potentially with some using ECN and some not.

Topo-Asymmetric: It is theoretically possible to build a middlebox that is a combination of an RTP mixer in one direction and an RTP translator in the other. To quote RFC 5117 "This topology is so problematic and it is so easy to get the RTCP processing wrong, that it is NOT RECOMMENDED to implement this topology."

These topologies may be combined within a single RTP session.

The ECN mechanism defined in this memo is applicable to both sender and receiver controlled congestion algorithms. The mechanism ensures that both senders and receivers will know about ECN-CE markings and any packet losses. Thus the actual decision point for the congestion control is not relevant. This is a great benefit as the rate of an RTP session can be varied in a number of ways, for example a unicast media sender might use TFRC [RFC5348] or some other algorithm, while a multicast session could use a sender based scheme adapting to the lowest common supported rate, or a receiver driven mechanism using layered coding to support more heterogeneous paths.

To ensure timely feedback of CE marked packets when needed, this mechanism requires support for the RTP/AVPF profile [RFC4585] or any of its derivatives, such as RTP/SAVPF [RFC5124]. The standard RTP/AVP profile [RFC3551] does not allow any early or immediate transmission of RTCP feedback, and has a minimal RTCP interval whose default value (5 seconds) is many times the normal RTT between sender and receiver.

3.3. Interoperability

The interoperability requirements for this specification are that there is at least one common interoperability point for all implementations. Since initialization using RTP and RTCP is the one method that works in all cases, although is not optimal for all usages, it is selected as mandatory to implement this initialisation method. This method requires both the RTCP XR extension and the ECN feedback format, which requires the RTP AVPF profile to ensure timely feedback.

When one considers all the uses of ECN for RTP it is clear that congestion control mechanisms that are receiver driven only (Section 7.3.3) do not require timely feedback of congestion events. If such a congestion control mechanism is combined with an initialization method that also doesn’t require timely feedback using RTCP, like the leap of faith or the ICE based method then neither the ECN feedback format nor AVPF is strictly needed. However, we would like to point out that fault detection can be improved by using receiver side detection (Section 7.4.1) and early reporting of such cases using the ECN feedback mechanism.

For interoperability we do mandate the implementation of AVPF, with both RTCP extensions and the necessary signalling to support a common operations mode. This specification will still recommend the usage of AVPF in all cases as negotiation of the common interoperability point requires AVPF, and mixed negotiation of AVP and AVPF depending on other SDP attributes in the same media block are difficult and the fact that fault detection can be improved when using AVPF. The use of the ECN feedback format is also recommended but cases where there is no requirement for timely feedback will be noted. The term "no timely feedback required" will be used to indicate usage that employs this specification in combination with receiver driven congestion control, and initialization methods that do not require timely feedback, i.e. currently leap of faith and ICE based. We also note that any receiver driven congestion control solution that still requires RTCP for signalling of any adaptation information to the sender will still require AVPF.

4. Overview of Use of ECN with RTP/UDP/IP

The solution for using ECN with RTP over UDP/IP consists of four different pieces that together make the solution work:

1. Negotiation of the capability to use ECN with RTP/UDP/IP

2. Initiation and initial verification of ECN capable transport

3. Ongoing use of ECN within an RTP session

4. Handling of dynamic groups through failure detection, verification and fallback

The solution includes a new SDP attribute (Section 6.1), the definition of new extensions to RTCP (Section 5) and STUN (Section 7.2.2).

Before an RTP session can be created, a signalling protocol is often used to discover the other participants and negotiate session parameters (see Section 7.1). At the minimum a signalling protocol is used to configure RTP session participants through a declarative method. One of the parameters that can be negotiated is the capability of a participant to support ECN functionality, or otherwise. Note that all participants having the capability of supporting ECN does not necessarily imply that ECN is usable in an RTP session, since there may be middleboxes on the path between the participants which don't pass ECN-marked packets (for example, a firewall that blocks traffic with the ECN bits set). This document defines the information that needs to be negotiated, and provides a mapping to SDP for use in both declarative and offer/answer contexts.

When a sender joins a session for which all participants claim ECN capability, it must verify if that capability is usable. There are three ways in which this verification may be done (Section 7.2):

- The sender may generate a (small) subset of its RTP data packets with the ECN field set to ECT(0) or ECT(1). Each receiver will then send an RTCP feedback packet indicating the reception of the ECT marked RTP packets. Upon reception of this feedback from each receiver it knows of, the sender can consider ECN functional for its traffic. Each sender does this verification independently of each other. If a new receiver joins an existing session it will reveal whether or not it supports ECN when it sends its first RTCP report to each source. If the RTCP report includes ECN information, verification will have succeeded and sources can continue to send ECT packets. If not, verification fails and each sender MUST stop using ECN.

- Alternatively, ECN support can be verified during an initial end-to-end STUN exchange (for example, as part of ICE connection establishment). After having verified connectivity without ECN capability an extra STUN exchange, this time with the ECN field set to ECT(0) or ECT(1), is performed. If successful the path's capability to convey ECN marked packets is verified. A new STUN attribute is defined to convey feedback that the ECT marked STUN request was received (see Section 7.2.2), along with an ICE signalling option (Section 6.4).

- Thirdly, the sender may make a leap of faith that ECN will work. This is only recommended for applications that know they are running in controlled environments where ECN functionality has been verified through other means. In this mode it is assumed that ECN works, and the system reacts to failure indicators if the assumption proved wrong. The use of this method relies on a high confidence that ECN operation will be successful, or an application where failure is not serious. The impact on the network and other users must be considered when making a leap of faith, so there are limitations on when this method is allowed.

The first mechanism, using RTP with RTCP feedback, has the advantage of working for all RTP sessions, but the disadvantages of potential clipping if ECN marked RTP packets are discarded by middleboxes, and slow verification of ECN support. The STUN-based mechanism is faster to verify ECN support, but only works in those scenarios supported by end-to-end STUN, such as within an ICE exchange. The third one, leap-of-faith, has the advantage of avoiding additional tests or complexities and enabling ECN usage from the first media packet. The downside is that if the end-to-end path contains middleboxes that do not pass ECN, the impact on the application can be severe: in the worst case, all media could be lost if a middlebox that discards ECN marked packets is present. A less severe effect, but still requiring reaction, is the presence of a middlebox that re-marks ECT marked packets to non-ECT, possibly marking packets with a CE mark as non-ECT. This can force the network into heavy congestion due to non-responsiveness, and seriously impact media quality.

Once ECN support has been verified (or assumed) to work for all receivers, a sender marks all its RTP packets as ECT packets, while receivers rapidly feedback any CE marks to the sender using RTCP in RTP/AVPF immediate or early feedback mode, unless no timely feedback is required. An RTCP feedback report is sent as soon as possible according to the transmission rules for feedback that are in place. This feedback report indicates the receipt of new CE marks since the last ECN feedback packet, and also counts the total number of CE marked packets through a cumulative sum. This is the mechanism to provide the fastest possible feedback to senders about CE marks. On receipt of a CE marked packet, the system must react to congestion as-if packet loss has been reported. Section 7.3 describes the ongoing use of ECN within an RTP session.

This rapid feedback is not optimised for reliability, therefore an additional procedure, the RTCP ECN summary reports, is used to ensure more reliable, but less timely, reporting of the ECN information. The ECN summary report contains the same information as the ECN feedback format, only packed differently for better efficiency with reports for many sources. It is sent in a compound RTCP packet, along with regular RTCP reception reports. By using cumulative counters for seen CE, ECT, not-ECT, and packet loss the sender can determine what events have happened since the last report, independently of any RTCP packets having been lost.

RTCP traffic MUST NOT be ECT marked for the following reason. ECT marked traffic may be dropped if the path is not ECN compliant. As RTCP is used to provide feedback about what has been transmitted and what ECN markings that are received, it is important that these are received in cases when ECT marked traffic is not getting through.

There are numerous reasons why the path the RTP packets take from the sender to the receiver may change, e.g., mobility, link failure followed by re-routing around it. Such an event may result in the packet being sent through a node that is ECN non-compliant, thus re-marking or dropping packets with ECT set. To prevent this from impacting the application for longer than necessary, the operation of ECN is constantly monitored by all senders. Both the RTCP ECN summary reports and the ECN feedback packets allow the sender to compare the number of ECT(0), ECT(1), and non-ECT marked packets received with the number that were sent, while also reporting CE marked and lost packets. If these numbers do not agree, it can be inferred that the path does not reliably pass ECN-marked packets (Section 7.4.2 discusses how to interpret the different cases). A sender detecting a possible ECN non-compliance issue should then stop sending ECT marked packets to determine if that allows the packets to be correctly delivered. If the issues can be connected to ECN, then ECN usage is suspended and possibly also re-negotiated.

5. RTCP Extensions for ECN feedback

This documents defines two different RTCP extensions: one RTP/AVPF [RFC4585] transport layer feedback format for urgent ECN information, and one RTCP XR [RFC3611] ECN summary report block type for regular reporting of the ECN marking information. The full definition of these extensions usage as part of the complete solution is laid out in Section 7.

5.1. RTP/AVPF Transport Layer ECN Feedback packet

This RTP/AVPF transport layer feedback format is intended for usage in AVPF early or immediate feedback modes when information needs to urgently reach the sender. Thus its main use is to report on reception of an ECN-CE marked RTP packet so that the sender may perform congestion control, or to speed up the initiation procedures by rapidly reporting that the path can support ECN-marked traffic. The feedback format is also defined with reduced size RTCP [RFC5506] in mind, where RTCP feedback packets may be sent without accompanying Sender or Receiver Reports that would contain the Extended Highest Sequence number and the accumulated number of packet losses. Both are important for ECN to verify functionality and keep track of when CE marking does occur.

The RTP/AVPF transport layer feedback packet starts with the common header defined by the RTP/AVPF profile [RFC4585] which is reproduced here for the reader's information:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|   FMT   |  PT=RTPFB=205 |          length               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of packet sender                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of media source                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:            Feedback Control Information (FCI)                 :
:                                                               :
```

Figure 1: RTP/AVPF Common Packet Format for Feedback Messages

From Figure 1 it can be determined the identity of the feedback provider and for which RTP packet sender it applies. Below is the feedback information format defined that is inserted as FCI for this particular feedback messages that is identified with an FMT value = [TBA1].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Extended Highest Sequence Number      | Lost packets counter  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     CE Counter                | not-ECT Counter               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ECT (0) Counter               | ECT (1) Counter               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: ECN Feedback Format

The FCI information for the ECN Feedback format (Figure 2) are the following:

Extended Highest Sequence Number: The least significant 20-bits from an Extended highest sequence number received value as defined by [RFC3550]. Used to indicate for which packet this report is valid up to.

Lost Packets Counter: The cumulative number of RTP packets that the receiver expected to receive from this SSRC, minus the number of packets it actually received. This is the same as the cumulative number of packets lost defined in Section 6.4.1 of [RFC3550] except represented in 12-bit signed format, compared to 24-bit in RTCP SR or RR packets. As with the equivalent value in RTCP SR or RR packets, note that packets that arrive late are not counted as lost, and the loss may be negative if there are duplicates.

CE Counter: The cumulative number of RTP packets received from this SSRC since the receiver joined the RTP session that were ECN-CE marked. The receiver should keep track of this value using a local representation that is longer than 16-bits, and only include the 16-bits with least significance. In other words, the field will wrap if more than 65535 packets has been received.

ECT(0) Counter: The cumulative number of RTP packets received from this SSRC since the receiver joined the RTP session that had an ECN field value of ECT(0). The receiver should keep track of this value using a local representation that is longer than 16-bits, and only include the 16-bits with least significance. In other words, the field will wrap if more than 65535 packets have been received.

ECT(1) Counter: The cumulative number of RTP packets received from this SSRC since the receiver joined the RTP session that had an ECN field value of ECT(1). The receiver should keep track of this value using a local representation that is longer than 16-bits, and only include the 16-bits with least significance. In other words, the field will wrap if more than 65535 packets have been received.

not-ECT Counter: The cumulative number of RTP packets received from this SSRC since the receiver joined the RTP session that had an ECN field value of not-ECT. The receiver should keep track of this value using a local representation that is longer than 16-bits, and only include the 16-bits with least significance. In other words, the field will wrap if more than 65535 packets have been received.

Each FCI block reports on a single source (SSRC). Multiple sources can be reported by including multiple RTCP feedback messages in an compound RTCP packet. The AVPF common header indicates both the sender of the feedback message and on which stream it relates to.

The counters SHALL be initiated to 0 for a new receiver. This to enable detection of CE or Packet loss already on the initial report from a specific participant.

The Extended Highest sequence number and packet loss fields are both truncated in comparison to the RTCP SR or RR versions. This is to save bits as the representation is redundant unless reduced size RTCP is used in such a way that only feedback packets are transmitted, with no SR or RR in the compound RTCP packet. Due to that fact regular RTCP reporting will include the longer versions of the fields and there will be less of an issue with wrapping unless the packet rate of the application is so high that the fields will wrap within a regular RTCP reporting interval. In that case the feedback packet will need to be sent in a compound packet together with the SR or RR packet.

There is an issue with packet duplication in relation to the packet loss counter. If one avoids holding state for which sequence number has been received then the way one can count loss is to count the number of received packets and compare that to the number of packets expected. As a result a packet duplication can hide a packet loss. If a receiver is tracking the sequence numbers actually received and suppresses duplicates it provides for a more reliable packet loss indication. Reordering may also result in that packet loss is reported in one report and then removed in the next.

The CE counter is actually more robust for packet duplication. Adding each received CE marked packet to the counter is not an issue. If one of the clones was CE marked that is still a indication of congestion. Packet duplication has potential impact on the ECN verification. Thus the sum of packets reported may be higher than the number sent. However, most detections are still applicable.

5.2. RTCP XR Report block for ECN summary information

This unilateral XR report block combined with RTCP SR or RR report blocks carries the same information as the ECN Feedback Packet and shall be based on the same underlying information. However, there is a difference in semantics between the feedback format and this XR version. Where the feedback format is intended to report on a CE mark as soon as possible, this extended report is for the regular RTCP report and continuous verification of the ECN functionality end-to-end.

The ECN Summary report block consists of one report block header:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      BT       | Reserved      |         Block Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

and then followed of one or more of the following report data blocks:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SSRC of Media Sender                                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| CE Counter                    | not-ECT Counter               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ECT (0) Counter               | ECT (1) Counter               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

BT: Block Type identifying the ECN summary report block. Value is [TBA2].

Reserved: All bits SHALL be set to 0 on transmission and ignored on reception.

Block Length: The length of the report block. Used to indicate the number of report data blocks present in the ECN summary report. This length will be 3*n, where n is the number of ECN summary report blocks, since blocks are a fixed size.

SSRC of Media Sender: The SSRC identifying the media sender this report is for.

CE Counter: as in Section 5.1.

ECT(0) Counter: as in Section 5.1.

ECT(1) Counter: as in Section 5.1.

not-ECT Counter: as in Section 5.1.

The Extended Highest Sequence number and the packet loss counter for each SSRC is not present in RTCP XR report, in contrast to the feedback version. The reason is that this summary report will rely on the information sent in the Sender Report (SR) or Receiver Report (RR) blocks part of the same RTCP compound packet. The information available in SR or RR are the Extended Highest Sequence number and the accumulated number of packet losses.

All the SSRCs that are present in the SR or RR SHALL also be included in the RTCP XR ECN summary report. In cases where the number of senders are so large that the combination of SR/RR and the ECN summary for all the senders exceed the MTU, then only a subset of the senders SHOULD be included so that the reports for the subset fits within the MTU. The subsets SHOULD be selected round-robin across multiple intervals so that all sources are reported.

6. SDP Signalling Extensions for ECN

This section defines a number of SDP signalling extensions used in the negotiation of the ECN for RTP support when using SDP. This include one SDP attribute "ecn-capable-rtp" that negotiates the actual operation of ECN for RTP. Two SDP signalling parameters are defined to indicate the usage of the RTCP XR ECN summary block and the AVPF feedback format for ECN. One ICE option SDP reprensenation is also defined.

6.1. Signalling ECN Capability using SDP

One new SDP attribute, "a=ecn-capable-rtp", is defined. This is a media level attribute, thus it is normally included as part of the media description, but if present at session level the same configuration applies to all media descriptions. It is not subject to the character set chosen. The aim of this signalling is to indicate the capability of the sender and receivers to support ECN, and to negotiate the method of ECN initiation to be used in the session. The attribute takes a list of initiation methods, ordered in decreasing preference. The defined values for the initiation method are:

rtp: Using RTP and RTCP as defined in Section 7.2.1.

ice: Using STUN within ICE as defined in Section 7.2.2.

leap: Using the leap of faith method as defined in Section 7.2.3.

Further methods may be specified in the future, so unknown methods MUST be ignored upon reception.

In addition, a number of OPTIONAL parameters may be included in the "a=ecn-capable-rtp" attribute as follows:

mode: This parameter signals the endpoint’s capability to set and read ECN marks in UDP packets. An examination of various operating systems has shown that end-system support for ECN marking of UDP packets may be symmetric or asymmetric. By this we mean that some systems may allow end points to set the ECN bits in an outgoing UDP packet but not read them, while others may allow applications to read the ECN bits but not set them. This either/or case may produce an asymmetric support for ECN and thus should be conveyed in the SDP signalling. The "mode=setread" state is the ideal condition where an endpoint can both set and read ECN bits in UDP packets. The "mode=setonly" state indicates that an endpoint can set the ECT bit, but cannot read the ECN bits from received UDP packets to determine if upstream congestion occurred. The "mode=readonly" state indicates that the endpoint can read the ECN bits to determine if congestion has occurred for incomming packet, but it cannot set the ECT bits in outgoing UDP packets. When the "mode=" parameter is omitted it is assumed that the node has "setread" capabilities. This option can provide for an early indication that ECN cannot be used in a session. This would be case when both the offerer and answerer set the "mode=" parameter to "setonly" or "readonly", or when an RTP sender entity considers offering "readonly".

ect: This parameter makes it possible to express the preferred ECT marking. This is either "random", "0", or "1", with "0" being implied if not specified. The "ect" parameter describes a receiver preference, and is useful in the case where the receiver knows it is behind a link using IP header compression, the efficiency of which would be seriously disrupted if it were to receive packets with randomly chosen ECT marks. It is RECOMMENDED that ECT(0) marking be used.

The ABNF [RFC5234] grammar for the "a=ecn-capable-rtp" attribute is as follows:

```
ecn-attribute  = "a=ecn-capable-rtp:" SP init-list [SP parm-list]
init-list      = init-value *("," init-value)
init-value     = "rtp" / "ice" / "leap" / init-ext
init-ext       = token
parm-list      = parm-value *(";" SP parm-value)
parm-value     = mode / ect / parm-ext
mode           = "mode=" ("setonly" / "setread" / "readonly")
ect            = "ect=" ("0" / "1" / "random")
parm-ext       = parm-name "=" parm-value-ext
parm-name      = token
parm-value-ext = token / quoted-string
quoted-string  = DQUOTE *qdtext DQUOTE
qdtext         = %x20-21 / %x23-7E / %x80-FF
                 ; any 8-bit ascii except <">

; external references:
  ; token: from RFC 4566
  ; SP and DQUOTE from RFC 5234
```

When SDP is used with the offer/answer model [RFC3264], the party generating the SDP offer MUST insert an "a=ecn-capable-rtp" attribute into the media section of the SDP offer of each RTP flow for which it wishes to use ECN. The attribute includes one or more ECN initiation methods in a comma separated list in decreasing order of preference, with any number of optional parameters following. The answering party compares the list of initiation methods in the offer with those it supports in order of preference. If there is a match, and if the receiver wishes to attempt to use ECN in the session, it includes an "a=ecn-capable-rtp" attribute containing its single preferred choice of initiation method in the media sections of the answer. If there is no matching initiation method capability, or if the receiver does not wish to attempt to use ECN in the session, it does not include an "a=ecn-capable-rtp" attribute in its answer. If the attribute is removed in the answer then ECN MUST NOT be used in any direction for that media flow. If there are initilization methods that are unknown, they MUST be ignored on reception and MUST NOT be included in an answer. The answer may also include optional parameters, as discussed below.

If the "mode=setonly" parameter is present in the "a=ecn-capable-rtp" attribute of the offer and the answering party is also "mode=setonly", then there is no common ECN capability, and the answer MUST NOT include the "a=ecn-capable-rtp" attribute. Otherwise, if the offer is "mode=setonly" then ECN may only be initiated in the direction from the offering party to the answering party.

If the "mode=readonly" parameter is present in the "a=ecn-capable-rtp" attribute of the offer and the answering party is "mode=readonly", then there is no common ECN capability, and the answer MUST NOT include the "a=ecn-capable-rtp" attribute. Otherwise, if the offer is "mode=readonly" then ECN may only be initiated in the direction from the answering party to the offering party.

If the "mode=setread" parameter is present in the "a=ecn-capable-rtp" attribute of the offer and the answering party is "setonly", then ECN may only be initiated in the direction from the answering party to the offering party. If the offering party is "mode=setread" but the answering party is "mode=readonly", then ECN may only be initiated in the direction from the offering party to the answering party. If both offer and answer are "mode=setread", then ECN may be initiated in both directions. Note that "mode=setread" is implied by the absence of a "mode=" parameter in the offer or the answer.

In an RTP session using multicast all participants intending to send RTP packets needs support setting ECT in the RTP packets, and all participants receiving needs to have the capability to read ECN values on incoming packets. Especially the later is important, otherwise no sender in the multicast session will be able to enable ECN. If a session is negotiated using offer/answer it is preferable that intended session participant would be aware of the signalling attributes and if not capable but ECN for RTP aware SHOULD refuse to join the session. For intended session participants that are not aware of the ECN for RTP signalling and simple ignore the signalling attribute the other party in the offer/answer exchange SHOULD terminate the SIP dialog so that the participant leaves the session.

The "ect=" parameter in the "a=ecn-capable-rtp" attribute is set independently in the offer and the answer. Its value in the offer indicates a preference for the sending behaviour of the answering party, and its value in the answer indicates a sending preference for the behaviour of the offering party. It will be the senders choice to honour the receivers preference for what to receive or not. In multicast sessions, any sender SHOULD send using the value declared in the ect parameter.

Unknown optional parameters MUST be ignored on reception, and MUST NOT be included in the answer. That way new parameters may be introduced and verified to be supported by the other end-point by having them include it in any answer.

When SDP is used in a declarative manner, for example in a multicast session using the Session Announcement Protocol (SAP, [RFC2974]), negotiation of session description parameters is not possible. The "a=ecn-capable-rtp" attribute MAY be added to the session description to indicate that the sender will use ECN in the RTP session. The attribute MUST include a single method of initiation. Participants MUST NOT join such a session unless they have the capability to receive ECN-marked UDP packets, implement the method of initiation, and can generate RTCP ECN feedback (note that having the capability to use ECN doesn't necessarily imply that the underlying network path between sender and receiver supports ECN). The mode parameter MAY be included also in declarative usage, to indicate the minimal capability is required by the consumer of the SDP. So for example in a SSM session the participants configured with a particular SDP will all be in a media receive only mode, thus mode=readonly will work as the capability of reporting on the ECN markings in the received is what is required. However, using "mode=readonly" also in ASM sessions is reasonable, unless all senders are required to attempt to use ECN for their outgoing RTP data traffic, in which case the mode needs to be set to "setread".

The "a=ecn-capable-rtp" attribute MAY be used with RTP media sessions using UDP/IP transport. It MUST NOT be used for RTP sessions using TCP, SCTP, or DCCP transport, or for non-RTP sessions.

As described in Section 7.3.3, RTP sessions using ECN require rapid RTCP ECN feedback, unless timely feedback is not required due to a receiver driven congestion control. To ensure that the sender can react to ECN-CE marked packets timely feedback is usually required. Thus, the use of the Extended RTP Profile for RTCP-Based Feedback (RTP/AVPF) [RFC4585] or other profile that inherits AVPF's signalling rules, MUST be signalled unless timely feedback is not required. If timely feedback is not required it is still RECOMMENDED to used AVPF. The signalling of an AVPF based profile is likely to be required even if the preferred method of initialization and the congestion control does not require timely feedback, as the common interoperable method is likely to be signalled or the improved fault reaction is desired.

6.2. RTCP Feedback SDP Parameter

A new "nack" feedback parameter "ecn" is defined to indicate the usage of the RTCP ECN feedback packet format (Section 5.1). The ABNF [RFC5234] definition of the SDP parameter extension is:

```
rtcp-fb-nack-param  = <See section 4.2 of RFC 4585>
rtcp-fb-nack-param /= ecn-fb-par
ecn-fb-par          = SP "ecn"
```

The offer/answer rules for this SDP feedback parameters are specified in AVPF [RFC4585].

6.3. XR Block SDP Parameter

A new unilateral RTCP XR block for ECN summary information is specified, thus the XR block SDP signalling also needs to be extended with a parameter. This is done in the same way as for the other XR blocks. The XR block SDP attribute as defined in Section 5.1 of the RTCP XR specification [RFC3611] is defined to be extendible. As no parameter values are needed for this ECN summary block, this parameter extension consistis of a simple parameter name used to indicate support and intent to use the XR block.

```
xr-format       = <See Section 5.1 of [RFC3611]>
xr-format      /= ecn-summary-par
ecn-summary-par = "ecn-sum"
```

For SDP declarative and offer/answer usage, see the RTCP XR specification[RFC3611] and its specfciation of how to handle unilateral parameters.

6.4. ICE Parameter to Signal ECN Capability

One new ICE [RFC5245] option, "rtp+ecn", is defined. This is used with the SDP session level "a=ice-options" attribute in an SDP offer to indicate that the initiator of the ICE exchange has the capability to support ECN for RTP-over-UDP flows (via "a=ice-options: rtp+ecn"). The answering party includes this same attribute at the session level in the SDP answer if it also has the capability, and removes the attribute if it does not wish to use ECN, or doesn’t have the capability to use ECN. If the ICE initiation method (Section 7.2.2) actually is going to be used, it is also needs to be explicitly negotiated using the "a=ecn-capable-rtp" attribute. This ICE option SHALL be included when the ICE initiation method is offered or declared in the SDP.

> Note: This signalling mechanism is not strictly needed as long as the STUN ECN testing capability is used within the context of this document. It may however be useful if the ECN verification capability is used in additional contexts.

7. Use of ECN with RTP/UDP/IP

In the detailed specification of the behaviour below, the different functions in the general case will first be discussed. In case special considerations are needed for middleboxes, multicast usage etc, those will be specially discussed in related subsections.

7.1. Negotiation of ECN Capability

The first stage of ECN negotiation for RTP-over-UDP is to signal the capability to use ECN. This includes negotiating if ECN is to be used symmetrically and the method for initial ECT verification. This memo defines the mappings of this information onto SDP for both declarative and offer/answer usage. There is one SDP extension to indicate if ECN support should be used, and the method for initiation (Section 6.1). Further parameters to indicate support for the AVPF ECN feedback format (Section 6.2) and the ECN XR summary report (Section 6.3). In addition an ICE parameter is defined (Section 6.4) to indicate that ECN initiation using STUN is supported as part of an ICE exchange.

An RTP system that supports ECN and uses SDP in the signalling MUST implement the SDP extension to signal ECN capability as described in Section 6.1, the ECN feedback SDP parameter Section 6.2, and the ECN XR SDP parameter Section 6.3. It MAY also implement alternative ECN capability negotiation schemes, such as the ICE extension described in Section 6.4.

The "ecn-capable-rtp" SDP attribute MUST always be used when employing ECN for RTP according to this specification. As the XR ECN summary report is required independently of the initialization method, or congestion control scheme the "rtcp-xr" attribute with the "ecn-sum" parameter MUST also be used. The "rtcp-fb" attribute with the "nack" parameter "ecn" MUST be used whenever the initialization method or a congestion control algorithm requiring timely sender side knowledge of received CE markings. If the congestion control scheme uses additional signalling they should be indicated as appropriate for those signalling methods.

7.2. Initiation of ECN Use in an RTP Session

Once the sender and the receiver(s) have agreed that they have the capability to use ECN within a session, they may attempt to initiate ECN use.

At the start of the RTP session, when the first packets with ECT are sent, it is important to verify that IP packets with ECN field values of ECT or ECN-CE will reach their destination(s). There is some risk that the use of ECN will result in either reset of the ECN field, or loss of all packets with ECT or ECN-CE markings. If the path between the sender and the receivers exhibits either of these behaviours one needs to stop using ECN immediately to protect both the network and the application.

The RTP senders and receivers SHALL NOT ECT mark their RTCP traffic at any time. This is to ensure that packet loss due to ECN marking will not effect the RTCP traffic and the necessary feedback information it carries.

An RTP system that supports ECN MUST implement the initiation of ECN using in-band RTP and RTCP described in Section 7.2.1. It MAY also implement other mechanisms to initiate ECN support, for example the STUN-based mechanism described in Section 7.2.2 or use the leap of faith option if the session supports the limitations provided in Section 7.2.3. If support for both in-band and out-of-band mechanisms is signalled, the sender should try ECN negotiation using STUN with ICE first, and if it fails, fallback to negotiation using RTP and RTCP ECN feedback.

No matter how ECN usage is initiated, the sender MUST continually monitor the ability of the network, and all its receivers, to support ECN, following the mechanisms described in Section 7.4. This is necessary because path changes or changes in the receiver population may invalidate the ability of the system to use ECN.

7.2.1. Detection of ECT using RTP and RTCP

The ECN initiation phase using RTP and RTCP to detect if the network path supports ECN comprises three stages. Firstly, the RTP sender generates some small fraction of its traffic with ECT marks to act a probe for ECN support. Then, on receipt of these ECT-marked packets, the receivers send RTCP ECN feedback packets and RTCP ECN summary reports to inform the sender that their path supports ECN. Finally, the RTP sender makes the decision to use ECN or not, based on whether the paths to all RTP receivers have been verified to support ECN.

Generating ECN Probe Packets: During the ECN initiation phase, an RTP sender SHALL mark a small fraction of its RTP traffic as ECT, while leaving the reminder of the packets unmarked. The main reason for only marking some packets is to maintain usable media delivery during the ECN initiation phase in those cases where ECN is not supported by the network path. A secondary reason to send some not-ECT packets are to ensure that the receivers will send RTCP reports on this sender, even if all ECT marked packets are lost in transit. The not-ECT packets also provide a base-line to compare performance parameters against. A fourth reason for only probing with a small number of packets is to reduce the risk that significant numbers of congestion markings might be lost if ECT is cleared to Not-ECT by an ECN-Reverting Meddlebox. Then any resulting lack of congestion response is likely to have little damaging affect on others. An RTP sender is RECOMMENDED to send a minimum of two packets with ECT markings per RTCP reporting interval. In case an random ECT pattern is intended to be used, at least one with ECT(0) and one with ECT(1) per reporting interval, in case a single ECT marking is to be used, only that ECT value SHOULD be sent. The RTP sender will continue to send some ECT marked traffic as long as the ECN initiation phase continues. The sender SHOULD NOT mark all RTP packets as ECT during the ECN initiation phase.

This memo does not mandate which RTP packets are marked with ECT during the ECN initiation phase. An implementation should insert ECT marks in RTP packets in a way that minimises the impact on media quality if those packets are lost. The choice of packets to mark is clearly very media dependent, but the usage of RTP NO-OP payloads [I-D.ietf-avt-rtp-no-op], if supported, would be an appropriate choice. For audio formats, if would make sense for the sender to mark comfort noise packets or similar. For video formats, packets containing P- or B-frames, rather than I-frames, would be an appropriate choice. No matter which RTP packets are marked, those packets MUST NOT be sent in duplicate with and without ECT, since their RTP sequence number is used to identify packets that are received with ECN markings.

Generating RTCP ECN Feedback: If ECN capability has been negotiated in an RTP session, the receivers in the session MUST listen for ECT or ECN-CE marked RTP packets, and generate RTCP ECN feedback packets (Section 5.1) to mark their receipt. An immediate or early (depending on the RTP/AVPF mode) ECN feedback packet SHOULD be generated on receipt of the first ECT or ECN-CE marked packet from a sender that has not previously sent any ECT traffic. Each regular RTCP report MUST also contain an ECN summary report (Section 5.2). Reception of subsequent ECN-CE marked packets MUST result in additional early or immediate ECN feedback packets being sent unless no timely feedback is required.

Determination of ECN Support: RTP is a group communication protocol, where members can join and leave the group at any time. This complicates the ECN initiation phase, since the sender must wait until it believes the group membership has stabilised before it can determine if the paths to all receivers support ECN (group membership changes after the ECN initiation phase has completed are discussed in Section 7.3).

An RTP sender shall consider the group membership to be stable after it has been in the session and sending ECT-marked probe packets for at least three RTCP reporting intervals (i.e., after sending its third regularly scheduled RTCP packet), and when a complete RTCP reporting interval has passed without changes to the group membership. ECN initiation is considered successful when the group membership is stable, and all known participants have sent one or more RTCP ECN feedback packets indicating correct receipt of the ECT-marked RTP packets generated by the sender.

As an optimisation, if an RTP sender is initiating ECN usage towards a unicast address, then it MAY treat the ECN initiation as provisionally successful if it receives a single RTCP ECN feedback report indicating successful receipt of the ECT-marked packets, with no negative indications, from a single RTP receiver. After declaring provisional success, the sender MAY generate ECT-marked packets as described in Section 7.3, provided it continues to monitor the RTCP reports for a period of three RTCP reporting intervals from the time the ECN initiation started, to check if there is any other participants in the session. If other participants are detected, the sender MUST fallback to only ECT-marking a small fraction of its RTP packets, while it determines if ECN can be supported following the full procedure described above.

> Note: One use case that requires further consideration is a unicast connection with several SSRCs multiplexed onto the same flow (e.g., an SVC video using SSRC multiplexing for the layers). It is desirable to be able to rapidly negotiate ECN support for such a session, but the optimisation above fails since the multiple SSRCs make it appear that this is a group communication scenario. It's not sufficient to check that all SSRCs map to a common RTCP CNAME to check if they're actually located on the same device, because there are implementations that use the same CNAME for different parts of a distributed implementation.

ECN initiation is considered to have failed at the instant when any RTP session participant sends an RTCP packet that doesn't contain an RTCP ECN feedback report or ECN summary report, but has an RTCP RR with an extended RTP sequence number field that indicates that it should have received multiple (>3) ECT marked RTP packets. This can be due to failure to support the ECN feedback format by the receiver or some middlebox, or the loss of all ECT marked packets. Both indicate a lack of ECN support.

If the ECN negotiation succeeds, this indicates that the path can pass some ECN-marked traffic, and that the receivers support ECN feedback. This does not necessarily imply that the path can robustly convey ECN feedback; Section 7.3 describes the ongoing monitoring that must be performed to ensure the path continues to robustly support ECN.

When a sender or receiver detects ECN failures on paths they should log these to enable follow up and statistics gathering regarding broken paths. The logging mechanism used is implementation dependent.

7.2.2. Detection of ECT using STUN with ICE

This section describes an OPTIONAL method that can be used to avoid media impact and also ensure an ECN capable path prior to media transmission. This method is considered in the context where the session participants are using ICE [RFC5245] to find working connectivity. We need to use ICE rather than STUN only, as the verification needs to happen from the media sender to the address and port on which the receiver is listening.

To minimise the impact of set-up delay, and to prioritise the fact that one has a working connectivity rather than necessarily finding the best ECN capable network path, this procedure is applied after having performed a successful connectivity check for a candidate, which is nominated for usage. At that point, and provided the chosen candidate is not a relayed address, an additional connectivity check is performed, sending the "ECT Check" attribute in a STUN packet that is ECT marked. On reception of the packet, a STUN server supporting this extension will note the received ECN field value, and send a STUN/UDP/IP packet in reply, with the ECN field set to not-ECT, and including an ECN check attribute. A STUN server that doesn’t understand the extension or is incapable of reading the ECN values on incoming STUN packets should follow the STUN specifications rule for unknown comprehension-optional attributes, i.e. ignore the attribute. Which will result in the sender receiving a STUN response but without the ECN Check STUN attribute.

The STUN ECN check attribute contains one field and a flag. The flag indicates whether the echo field contains a valid value or not. The field is the ECN echo field, and when valid contains the two ECN bits from the packet it echoes back. The ECN check attribute is a comprehension optional attribute.

```
 0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Type                  |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Reserved                                      |ECF|V|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: ECN Check STUN Attribute

V: Valid (1 bit) ECN Echo value field is valid when set to 1, and invalid when set 0.

ECF: ECN Echo value field (2 bits) contains the ECN field value of the STUN packet it echoes back when field is valid. If invalid the content is arbitrary.

Reserved: Reserved bits (29 bits) SHALL be set to 0 on transmission, and SHALL be ignored on reception.

This attribute MAY be included in any STUN request to request the ECN field to be echoed back. In STUN requests the V bit SHALL be set to 0. A compliant STUN server receiving a request with the ECN Check attribute SHALL read the ECN field value of the IP/UDP packet the request was received in. Upon forming the response the server SHALL include the ECN Check attribute setting the V bit to valid and include the read value of the ECN field into the ECF field. If the STUN responder was unable to ascertain, due to temporary errors, the ECN value of the STUN request, it SHALL set the V bit in the response to 0. The STUN client may retry immediately.

7.2.3. Leap of Faith ECT initiation method

This method for initiating ECN usage is a leap of faith that assumes that ECN will work on the used path(s). The method is to go directly to "ongoing use of ECN" as defined in Section 7.3. Thus all RTP packets MAY be marked as ECT and the failure detection MUST be used to detect any case when the assumption that the path was ECT capable is wrong. This method is only recommended for controlled environments where the whole path(s) between sender and receiver(s) has been built and verified to be ECT.

If the sender marks all packets as ECT while transmitting on a path that contains an ECN-blocking middlebox, then receivers downstream of that middlebox will not receive any RTP data packets from the sender, and hence will not consider it to be an active RTP SSRC. The sender can detect this and revert to sending packets without ECT marks, since RTCP SR/RR packets from such receivers will either not include a report for sender’s SSRC, or will report that no packets have been received, but this takes at least one RTCP reporting interval. It should be noted that a receiver might generate its first RTCP packet immediately on joining a unicast session, or very shortly after joining a RTP/AVPF session, before it has had chance to receive any data packets. A sender that receives RTCP SR/RR packet indicating lack of reception by a receiver SHOULD therefore wait for a second RTCP report from that receiver to be sure that the lack of reception is due to ECT-marking. Since this recovery process can take several tens of seconds, during which time the RTP session is unusable for media, it is NOT RECOMMENDED that the leap-of-faith ECT initiation method be used in environments where ECN-blocking middleboxes are likely to be present.

7.3. Ongoing Use of ECN Within an RTP Session

Once ECN usage has been successfully initiated for an RTP sender, that sender begins sending all RTP data packets as ECT-marked, and its receivers continue sending ECN feedback information via RTCP packets. This section describes procedures for sending ECT-marked data, providing ECN feedback information via RTCP, responding to ECN feedback information, and detecting failures and misbehaving receivers.

7.3.1. Transmission of ECT-marked RTP Packets

After a sender has successfully initiated ECN usage, it SHOULD mark all the RTP data packets it sends as ECT. The sender SHOULD mark packets as ECT(0) unless the receiver expresses a preference for ECT(1) or random using the "ect" parameter in the "a=ecn-capable-rtp" attribute.

The sender SHALL NOT include ECT marks on outgoing RTCP packets, and SHOULD NOT include ECT marks on any other outgoing control messages (e.g. STUN [RFC5389] packets, DTLS [RFC4347] handshake packets, or ZRTP [I-D.zimmermann-avt-zrtp] control packets) that are multiplexed on the same UDP port. For control packets there might be exceptions, like the STUN based ECN check defined in Section 7.2.2.

7.3.2. Reporting ECN Feedback via RTCP

An RTP receiver that receives a packet with an ECN-CE mark, or that detects a packet loss, MUST schedule the transmission of an RTCP ECN feedback packet as soon as possible (subject to the constraints of [RFC4585] and [RFC3550]) to report this back to the sender unless no timely feedback required. There should be no difference in behavior if ECN-CE marks or packet drops are detected. The feedback RTCP packet sent SHALL consist of at least one ECN feedback packet (Section 5) reporting on the packets received since the last ECN feedback packet, and SHOULD contain an RTCP SR or RR packet. The RTP/AVPF profile in early or immediate feedback mode SHOULD be used where possible, to reduce the interval before feedback can be sent. To reduce the size of the feedback message, reduced size RTCP [RFC5506] MAY be used if supported by the end-points. Both RTP/AVPF and reduced size RTCP MUST be negotiated in the session set-up signalling before they can be used.

Every time a regular compound RTCP packet is to be transmitted, an

ECN-capable RTP receiver MUST include an RTCP XR ECN summary report as described in Section 5.2 as part of the compound packet.

The multicast feedback implosion problem, that occurs when many receivers simultaneously send feedback to a single sender, must also be considered. The RTP/AVPF transmission rules will limit the amount of feedback that can be sent, avoiding the implosion problem but also delaying feedback by varying degrees from nothing up to a full RTCP reporting interval. As a result, the full extent of a congestion situation may take some time to reach the sender, although some feedback should arrive in a reasonably timely manner, allowing the sender to react on a single or a few reports.

> A possible future optimisation might be to define some form of feedback suppression mechanism to reduce the RTCP reporting overhead for group communication using ECN.

7.3.3. Response to Congestion Notifications

The reception of RTP packets with ECN-CE marks in the IP header are a notification that congestion is being experience. The default reaction on the reception of these ECN-CE marked packets MUST be to provide the congestion control algorithm with notification and that it is treated as a packet loss would when it comes to indicating congestion.

We note that there MAY be other reactions to ECN-CE specified in the future. Such an alternative reaction MUST be specified and considered to be safe for deployment under any restrictions specified. A potential example for an alternative reaction could be emergency communications (such as that generated by first responders, as opposed to the general public) in networks where the user has been authorized. A more detailed description of these other reactions, as well as the types of congestion control algorithms used by end-nodes, is outside of the scope of this document.

Depending on the media format, type of session, and RTP topology used, there are several different types of congestion control that can be used.

Sender-Driven Congestion Control: The sender may be responsible for adapting the transmitted bit-rate in response to RTCP ECN feedback. When the sender receives the ECN feedback data it feeds this information into its congestion control or bit-rate adaptation mechanism so that it can react on it as if it was packet losses that was reported. The congestion control algorithm to be used is not specified here, although TFRC [RFC5348] is one example that might be used.

Receiver-Driven Congestion Control: If a receiver driven congestion control mechanism is used, the receiver can react to the ECN-CE marks without contacting the sender. This may allow faster response than sender-driven congestion control in some circumstances. Receiver-driven congestion control is usually implemented by providing the content in a layered way, with each layer providing improved media quality but also increased bandwidth usage. The receiver locally monitors the ECN-CE marks on received packet to check if it experiences congestion at the current number of layers. If congestion is experienced, the receiver drops one layer, so reducing the resource consumption on the path towards itself. For example, if a layered media encoding scheme such as H.264 SVC is used, the receiver may change its layer subscription, and so reduce the bit rate it receives. The receiver MUST still send RTCP XR ECN Summary to the sender, even if it can adapt without contact with the sender, so that the sender can determine if ECN is supported on the network path. The timeliness of RTCP feedback is less of a concern with receiver driven congestion control, and regular RTCP reporting of ECN summary information is sufficient (without using RTP/AVPF immediate or early feedback).

Hybrid: There might be mechanisms that utilize both some receiver behaviors and some sender side monitoring, thus requiring both feedback of congestion events to the sender and taking receiver decisions and possible signalling to the sender. From this solution the congestion control algorithm needs to use the signalling to indicate which functions of ECN that is needed to be used.

Responding to congestion indication in the case of multicast traffic is a more complex problem than for unicast traffic. The fundamental problem is diverse paths, i.e. when different receivers don't see the same path, and thus have different bottlenecks, so the receivers may get ECN-CE marked packets due to congestion at different points in the network. This is problematic for sender driven congestion control, since when receivers are heterogeneous in regards to capacity the sender is limited to transmitting at the rate the slowest receiver can support. This often becomes a significant limitation as group size grows. Also, as group size increases the frequency of reports from each receiver decreases, which further reduces the responsiveness of the mechanism. Receiver-driven congestion control has the advantage that each receiver can choose the appropriate rate for its network path, rather than all having to settle for the lowest common rate.

We note that ECN support is not a silver bullet to improving performance. The use of ECN gives the chance to respond to congestion before packets are dropped in the network, improving the user experience by allowing the RTP application to control how the quality is reduced. An application which ignores ECN congestion experienced feedback is not immune to congestion: the network will eventually begin to discard packets if traffic doesn't respond. It is in the best interest of an application to respond to ECN congestion feedback promptly, to avoid packet loss.

7.4. Detecting Failures

Senders and receivers can deliberately ignore ECN-CE and thus get a benefit over behaving flows (cheating). Nonce [RFC3540] is an addition to TCP that solves this issue as long as the sender acts on behalf of the network. The assumption about the senders acting on the behalf of the network may be reduced due to the nature of peer-to-peer use of RTP. Still a significant portion of RTP senders are infrastructure devices (for example, streaming media servers) that do have an interest in protecting both service quality and the network. Even though there may be cases where nonce can be applicable also for RTP, it is not included in this specification. This as a receiver interested in cheating would simple claim to not support Nonce. It is however worth mention that, as real-time media is commonly sensitive to increased delay and packet loss, it will be in both media sender and receivers interest to minimise the number and duration of any congestion events as they will affect media quality.

RTP sessions can also suffer from path changes resulting in a non-ECN compliant node becoming part of the path. That node may perform either of two actions that has effect on the ECN and application functionality. The gravest is if the node drops packets with any ECN field values other than 00b. This can be detected by the receiver when it receives a RTCP SR packet indicating that a sender has sent a number of packets has not been received. The sender may also detect it based on the receivers RTCP RR packet where the extended sequence number is not advanced due to the failure to receive packets. If the packet loss is less than 100% then packet loss reporting in either the ECN feedback information or RTCP RR will indicate the situation. The other action is to re-mark a packet from ECT or CE to not-ECT. That has less dire results, however, it should be detected so that ECN usage can be suspended to prevent misusing the network.

The ECN feedback packet allows the sender to compare the number of ECT marked packets of different type with the number it actually sent. The number of ECT packets received plus the number of CE marked and lost packets should correspond to the number of sent ECT marked packets unless there is duplication in the network. If this number doesn't agree there are two likely reasons, a translator changing the stream or not carrying the ECN markings forward, or that some node re-marks the packets. In both cases the usage of ECN is broken on the path. By tracking all the different possible ECN field values a sender can quickly detect if some non-compliant behavior is happing on the path.

Thus packet losses and non-matching ECN field value statistics are possible indication of issues with using ECN over the path. The next section defines both sender and receiver reactions to these cases.

7.4.1. Fallback mechanisms

Upon the detection of a potential failure both the sender and the receiver can react to mitigate the situation.

A receiver that detects a packet loss burst MAY schedule an early feedback packet to report this to the sender that includes at least the RTCP RR and the ECN feedback message. Thus speeding up the detection at the sender of the losses and thus triggering sender side mitigation.

A sender that detects high packet loss rates for ECT-marked packets SHOULD immediately switch to sending packets as not-ECT to determine if the losses potentially are due to the ECT markings. If the losses disappear when the ECT-marking is discontinued, the RTP sender should go back to initiation procedures to attempt to verify the apparent loss of ECN capability of the used path. If a re-initiation fails then the two possible actions exist:

1. Periodically retry the ECN initiation to detect if a path change occurs to a path that is ECN capable.

2. Renegotiating the session to disable ECN support. This is a choice that is suitable if the impact of ECT probing on the media quality are noticeable. If multiple initiations has been successful but the following full usage of ECN has resulted in the fallback procedures then disabling of the ECN support is RECOMMENDED.

We foresee the possibility of flapping ECN capability due to several reasons: video switching MCU or similar middleboxes that selects to deliver media from the sender only intermittently; load balancing devices may in worst case result in that some packets take a different network path then the others; mobility solutions that switch underlying network path in a transparent way for the sender or receiver; and membership changes in a multicast group. It is however appropriate to mention that there are also issues such as re-routing of traffic due to a flappy route table or excessive reordering and other issues that are not directly ECN related but nevertheless may cause problems for ECN.

7.4.2. Interpretation of ECN Summary information

This section contains discussion on how you can use the ECN summary report information in detecting various types of ECN path issues. Lets start to review the information the reports provide on a per source (SSRC) basis:

CE Counter: The number of RTP packets received so far in the session with an ECN field set to CE (11b).

ECT (0/1) Counters: The number of RTP packets received so far in the session with an ECN field set to ECT (0) and ECT (1) respectively (10b / 01b).

not-ECT Counter: The number of RTP packets received so far in the session with an ECN field set to not-ECT (00b)

Lost Packets counter: The number of RTP packets that are expected minus the number received.

Extended Highest Sequence number: The highest sequence number seen when sending this report, but with additional bits, to handle disambiguation when wrapping the RTP sequence number field.

The counters will be initiated to zero to provide value for the RTP stream sender from the very first report. After the first report the changes between the latest received and the previous one is determined by simply taking the values of the latest minus the previous one, taking field wrapping into account. This definition is also robust to packet losses, since if one report is missing, the reporting interval becomes longer, but is otherwise equally valid.

In a perfect world the number of not-ECT packets received should be equal to the number sent minus the lost packets counter, and the sum of the ECT(0), ECT(1), and CE counters should be equal to the number of ECT marked packet sent. Two issues may cause a mismatch in these statistics: severe network congestion or unresponsive congestion control might cause some ECT-marked packets to be lost, and packet duplication might result in some packets being received, and counted in the statistics, multiple times (potentially with a different ECN-mark on each copy of the duplicate).

The level of packet duplication included in the report can be estimated from the sum over all of fields counting received packets compared to the number of packets sent. A high level of packet duplication increases the uncertainty in the statistics, making it more difficult to draw firm conclusions about the behaviour of the network. This issue is also present with standard RTCP reception reports.

Detecting clearing of ECN field: If the ratio between ECT and not-ECT transmitted in the reports has become all not-ECT or substantially changed towards not-ECT then this is clearly indication that the path results in clearing of the ECT field.

Dropping of ECT packets: To determine if the packet drop ratio is different between not-ECT and ECT marked transmission requires a mix of transmitted traffic. The sender should compare if the delivery percentage (delivered / transmitted) between ECT and not-ECT is significantly different. Care must be taken if the number of packets are low in either of the categories. One must also take into account the level of CE marking. A CE marked packet would have been dropped unless it was ECT marked. Thus, the packet loss level for not-ECT should be aprroximately equal to the loss rate for ECT when counting the CE marked packets as lost ones. A sender performing this calculation needs to ensure that the difference is statistcally significant.

If erronous behavior is detected, it should be logged to enable follow up and statistics gathering.

8. Processing RTCP ECN Feedback in RTP Translators and Mixers

RTP translators and mixers that support ECN feedback are required to process, and potentially modify or generate, RTCP packets for the translated and/or mixed streams. This includes both downstream RTCP reports generated by the media sender, and also reports generated by the receivers, flowing upstream back towards the sender.

8.1. Fragmentation and Reassembly in Translators

An RTP translator may fragment or reassemble RTP data packets without changing the media encoding, and without reference to the congestion state of the networks it bridges. An example of this might be to combine packets of a voice-over-IP stream coded with one 20ms frame per RTP packet into new RTP packets with two 20ms frames per packet, thereby reducing the header overheads and so stream bandwidth, at the expense of an increase in latency. If multiple data packets are re-encoded into one, or vice versa, the RTP translator MUST assign new sequence numbers to the outgoing packets. Losses in the incoming RTP packet stream may also induce corresponding gaps in the outgoing RTP sequence numbers. An RTP translator MUST rewrite RTCP packets to make the corresponding changes to their sequence numbers, and to reflect the impact of the fragmentation or reassembly. This section describes how that rewriting is to be done for RTCP ECN feedback packets. Section 7.2 of [RFC3550] describes general procedures for other RTCP packet types.

RTCP ECN feedback packets (Section 5.1) contain six fields that are rewritten in an RTP translator that fragments or reassembles packets: the extended highest sequence number, the lost packets counter, the CE counter, and not-ECT counter, the ECT(0) counter, and the ECT(1) counter. The RTCP XR report block for ECN summary information (Section 5.2) includes a subset of these fields excluding the extended highest sequence number and lost packets counter. The procedures for rewriting these fields are the same for both types of RTCP ECN feedback packet.

When receiving an RTCP ECN feedback packet for the translated stream, an RTP translator first determines the range of packets to which the report corresponds. The extended highest sequence number in the RTCP ECN feedback packet (or in the RTCP SR/RR packet contained within the compound packet, in the case of RTCP XR ECN summary reports) specifies the end sequence number of the range. For the first RTCP ECN feedback packet received, the initial extended sequence number of the range may be determined by subtracting the sum of the lost packets counter, the CE counter, the not-ECT counter, the ECT(0) counter and the ECT(1) counter from the extended highest sequence number (this will be inaccurate if there is packet duplication). For subsequent RTCP ECN feedback packets, the starting sequence number may be determined as being one after the extended highest sequence number of the previous RTCP ECN feedback packet received from the same SSRC. These values are in the sequence number space of the translated packets.

Based on its knowledge of the translation process, the translator determines the sequence number range for the corresponding original, pre-translation, packets. The extended highest sequence number in the RTCP ECN feedback packet is rewritten to match the final sequence number in the pre-translation sequence number range.

The translator then determines the ratio, R, of the number of packets in the translated sequence number space (numTrans) to the number of packets in the pre-translation sequence number space (numOrig) such that R = numTrans / numOrig. The counter values in the RTCP ECN feedback report are then scaled by dividing each of them by R. For example, if the translation process combines two RTP packets into one, then numOrig will be twice numTrans, giving R=0.5, and the counters in the translated RTCP ECN feedback packet will be twice those in the original.

The ratio, R, may have a value that leads to non-integer multiples of the counters when translating the RTCP packet. For example, a VoIP translator that combines two adjacent RTP packets into one if they contain active speech data, but passes comfort noise packets unchanged, would have an R values of between 0.5 and 1.0 depending on the amount of active speech. Since the counter values in the translated RTCP report are integer values, rounding will be necessary in this case.

When rounding counter values in the translated RTCP packet, the translator should try to ensure that they sum to the number of RTP packets in the pre-translation sequence number space (numOrig). The translator should also try to ensure that no non-zero counter is rounded to a zero value, since that will lose information that a particular type of event has occurred. It is recognised that it may be impossible to satisfy both of these constraints; in such cases, it is better to ensure that no non-zero counter is mapped to a zero value, since this preserves congestion adaptation and helps the RTCP-based ECN initiation process.

It should be noted that scaling the RTCP counter values in this way is meaningful only on the assumption that the level of congestion in the network is related to the number of packets being sent. This is likely to be a reasonable assumption in the type of environment where RTP translators that fragment or reassemble packets are deployed, as their entire purpose is to change the number of packets being sent to adapt to known limitations of the network, but is not necessarily valid in general.

The rewritten RTCP ECN feedback report is sent from the other side of the translator to that which it arrived (as part of a compound RTCP packet containing other translated RTCP packets, where appropriate).

8.2. Generating RTCP ECN Feedback in Media Transcoders

An RTP translator that acts as a media transcoder cannot directly forward RTCP packets corresponding to the transcoded stream, since those packets will relate to the non-transcoded stream, and will not be useful in relation to the transcoded RTP flow. Such a transcoder will need to interpose itself into the RTCP flow, acting as a proxy for the receiver to generate RTCP feedback in the direction of the sender relating to the pre-transcoded stream, and acting in place of the sender to generate RTCP relating to the transcoded stream, to be sent towards the receiver. This section describes how this proxying is to be done for RTCP ECN feedback packets. Section 7.2 of [RFC3550] describes general procedures for other RTCP packet types.

An RTP translator acting as a media transcoder in this manner does not have its own SSRC, and hence is not visible to other entities at the RTP layer. RTCP ECN feedback packets and RTCP XR report blocks for ECN summary information that are received from downstream relate to the translated stream, and so must be processed by the translator as if it were the original media source. These reports drive the congestion control loop and media adaptation between the translator and the downstream receiver. If there are multiple downstream receivers, a logically separate transcoder instance must be used for each receiver, and must process RTCP ECN feedback and summary reports independently to the other transcoder instances. An RTP translator acting as a media transcoder in this manner MUST NOT forward RTCP ECN feedback packets or RTCP XR ECN summary reports from downstream receivers in the upstream direction.

An RTP translator acting as a media transcoder will generate RTCP reports upstream towards the original media sender, based on the reception quality of the original media stream at the translator. The translator will run a separate congestion control loop and media adaptation between itself and the media sender for each of its downstream receivers, and must generate RTCP ECN feedback packets and RTCP XR ECN summary reports for that congestion control loop using the SSRC of that downstream receiver.

8.3. Generating RTCP ECN Feedback in Mixers

An RTP mixer terminates one-or-more RTP flows, combines them into a single outgoing media stream, and transmits that new stream as a separate RTP flow. A mixer has its own SSRC, and is visible to other participants in the session at the RTP layer.

An ECN-aware RTP mixer must generate RTCP ECN feedback packets and RTCP XR report blocks for ECN summary information relating to the RTP flows it terminates, in exactly the same way it would if it were an RTP receiver. These reports form part of the congestion control loop between the mixer and the media senders generating the streams it is mixing. A separate control loop runs between each sender and the mixer.

An ECN-aware RTP mixer will negotiate and initiate the use of ECN on the mixed flows it generates, and will accept and process RTCP ECN feedback reports and RTCP XR report blocks for ECN relating to those mixed flows as if it were a standard media sender. A congestion control loop runs between the mixer and its receivers, driven in part by the ECN reports received.

An RTP mixer MUST NOT forward RTCP ECN feedback packets or RTCP XR ECN summary reports reports from downstream receivers in the upstream direction.

9. Implementation considerations

To allow the use of ECN with RTP over UDP, the RTP implementation must be able to set the ECT bits in outgoing UDP datagrams, and must be able to read the value of the ECT bits on received UDP datagrams. The standard Berkeley sockets API pre-dates the specification of ECN, and does not provide the functionality which is required for this mechanism to be used with UDP flows, making this specification difficult to implement portably.

10. IANA Considerations

Note to RFC Editor: please replace "RFC XXXX" below with the RFC number of this memo, and remove this note.

10.1. SDP Attribute Registration

Following the guidelines in [RFC4566], the IANA is requested to register one new SDP attribute:

- Contact name, email address and telephone number: Authors of RFCXXXX

- Attribute-name: ecn-capable-rtp

- Type of attribute: media-level

- Subject to charset: no

This attribute defines the ability to negotiate the use of ECT (ECN capable transport). This attribute should be put in the SDP offer if the offering party wishes to receive an ECT flow. The answering party should include the attribute in the answer if it wish to receive an ECT flow. If the answerer does not include the attribute then ECT MUST be disabled in both directions.

10.2. RTP/AVPF Transport Layer Feedback Message

The IANA is requested to register one new RTP/AVPF Transport Layer Feedback Message in the table of FMT values for RTPFB Payload Types [RFC4585] as defined in Section 5.1:

```
Name:           RTCP-ECN-FB
Long name:      RTCP ECN Feedback
Value:          TBA1
Reference:      RFC XXXX
```

10.3. RTCP Feedback SDP Parameter

The IANA is requested to register one new SDP "rtcp-fb" attribute "nack" parameter "ecn" in the SDP ("ack" and "nack" Attribute Values) registry.

```
Value name:     ecn
Long name:      Explicit Congestion Notification
Usable with:    nack
Reference:      RFC XXXX
```

10.4. RTCP XR Report blocks

The IANA is requested to register one new RTCP XR Block Type as defined in Section 5.2:

```
Block Type: TBA2
Name:       ECN Summary Report
Reference:  RFC XXXX
```

10.5. RTCP XR SDP Parameter

The IANA is requested to register one new RTCP XR SDP Parameter "ecn-sum" in the "RTCP XR SDP Parameters" registry.

```
Parameter name      XR block (block type and name)
--------------      ------------------------------------
ecn-sum             TBA2  ECN Summary Report Block
```

10.6. STUN attribute

A new STUN [RFC5389] attribute in the Comprehension-optional range under IETF Review (0x0000 - 0x3FFF) is request to be assigned to the STUN attribute defined in Section 7.2.2. The STUN attribute registry can currently be found at: http://www.iana.org/assignments/stun-parameters/stun-parameters.xhtml.

10.7. ICE Option

A new ICE option "rtp+ecn" is registered in the registry that "IANA Registry for Interactive Connectivity Establishment (ICE) Options" [I-D.ietf-mmusic-ice-options-registry] creates.

11. Security Considerations

The usage of ECN with RTP over UDP as specified in this document has the following known security issues that needs to be considered.

External threats to the RTP and RTCP traffic:

Denial of Service affecting RTCP: For an attacker that can modify the traffic between the media sender and a receiver can achieve either of two things. 1. Report a lot of packets as being Congestion Experience marked, thus forcing the sender into a congestion response. 2. Ensure that the sender disable the usage of ECN by reporting failures to receive ECN by changing the counter fields. The Issue, can also be accomplished by injecting false RTCP packets to the media sender. Reporting a lot of CE marked traffic is likely the more efficient denial of service tool as that may likely force the application to use lowest possible bit-rates. The prevention against an external threat is to integrity protect the RTCP feedback information and authenticate the sender of it.

Information leakage: The ECN feedback mechanism exposes the receivers perceived packet loss, what packets it considers to be ECN-CE marked and its calculation of the ECN-none. This is mostly not considered sensitive information. If considered sensitive the RTCP feedback shall be encrypted.

Changing the ECN bits An on-path attacker that see the RTP packet flow from sender to receiver and who has the capability to change the packets can rewrite ECT into ECN-CE thus forcing the sender or receiver to take congestion control response. This denial of service against the media quality in the RTP session is impossible for en end-point to protect itself against. Only network infrastructure nodes can detect this illicit re-marking. It will be mitigated by turning off ECN, however, if the attacker can modify its response to drop packets the same vulnerability exist.

Denial of Service affecting the session set-up signalling: If an attacker can modify the session signalling it can prevent the usage of ECN by removing the signalling attributes used to indicate that the initiator is capable and willing to use ECN with RTP/UDP. This attack can be prevented by authentication and integrity protection of the signalling. We do note that any attacker that can modify the signalling has more interesting attacks they can perform than prevent the usage of ECN, like inserting itself as a middleman in the media flows enabling wire-tapping also for an off-path attacker.

The following are threats that exist from misbehaving senders or receivers:

Receivers cheating A receiver may attempt to cheat and fail to report reception of ECN-CE marked packets. The benefit for a receiver cheating in its reporting would be to get an unfair bit-rate share across the resource bottleneck. It is far from certain that a receiver would be able to get a significant larger share of the resources. That assumes a high enough level of aggregation that there are flows to acquire shares from. The risk of cheating is that failure to react to congestion results in packet loss and increased path delay.

Receivers misbehaving: A receiver may prevent the usage of ECN in an RTP session by reporting itself as non ECN capable. Thus forcing the sender to turn off usage of ECN. In a point-to-point scenario there is little incentive to do this as it will only affect the receiver. Thus failing to utilise an optimisation. For multi-party session there exist some motivation why a receiver would misbehave as it can prevent also the other receivers from using ECN. As an insider into the session it is difficult to determine if a receiver is misbehaving or simply incapable, making it basically impossible in the incremental deployment phase of ECN for RTP usage to determine this. If additional information about the receivers and the network is known it might be possible to deduce that a receiver is misbehaving. If it can be determined that a receiver is misbehaving, the only response is to exclude it from the RTP session and ensure that is doesn't any longer have any valid security context to affect the session.

Misbehaving Senders: The enabling of ECN gives the media packets a higher degree of probability to reach the receiver compared to not-ECT marked ones on a ECN capable path. However, this is no magic bullet and failure to react to congestion will most likely only slightly delay a buffer under-run, in which its session also will experience packet loss and increased delay. There are some chance that the media senders traffic will push other traffic out of the way without being effected to negatively. However, we do note that a media sender still needs to implement congestion control functions to prevent the media from being badly affected by congestion events. Thus the misbehaving sender is getting a unfair share. This can only be detected and potentially prevented by network monitoring and administrative entities. See Section 7 of [RFC3168] for more discussion of this issue.

We note that the end-point security functions needs to prevent an external attacker from affecting the solution easily are source authentication and integrity protection. To prevent what information leakage there can be from the feedback encryption of the RTCP is also needed. For RTP there exist multiple solutions possible depending on the application context. Secure RTP (SRTP) [RFC3711] does satisfy the requirement to protect this mechanism despite only providing authentication if a entity is within the security context or not. IPsec [RFC4301] and DTLS [RFC4347] can also provide the necessary security functions.

The signalling protocols used to initiate an RTP session also needs to be source authenticated and integrity protected to prevent an external attacker from modifying any signalling. Here an appropriate mechanism to protect the used signalling needs to be used. For SIP/SDP ideally S/MIME [RFC5751] would be used. However, with the limited deployment a minimal mitigation strategy is to require use of SIPS (SIP over TLS) [RFC3261] [RFC5630] to at least accomplish hop-by-hop protection.

We do note that certain mitigation methods will require network functions.

12. Examples of SDP Signalling

This section contain a few different examples of the signalling mechanism defined in this specification in an SDP context. If there is discrepancies between these examples and the specification text, the specification text is what is correct.

12.1. Basic SDP Offer/Answer

This example is a basic offer/answer SDP exchange, assumed done by SIP (not shown). The intention is to establish a basic audio session point to point between two users.

The Offer:

```
v=0
o=jdoe 3502844782 3502844782 IN IP4 10.0.1.4
s=VoIP call
i=SDP offer for VoIP call with ICE and ECN for RTP
b=AS:128
b=RR:2000
b=RS:2500
a=ice-pwd:YH75Fviy6338Vbrhrlp8Yh
a=ice-ufrag:9uB6
a=ice-options:rtp+ecn
t=0 0
m=audio 45664 RTP/AVPF 97 98 99
c=IN IP4 192.0.2.3
a=rtpmap:97 G719/48000/1
a=fmtp:97 maxred=160
a=rtpmap:98 AMR-WB/16000/1
a=fmtp:98 octet-align=1; mode-change-capability=2
a=rtpmap:99 PCMA/8000/1
a=maxptime:160
a=ptime:20
a=ecn-capable-rtp: ice rtp ect=0 mode=setread
a=rtcp-fb:* nack ecn
a=rtcp-fb:* trr-int 1000
a=rtcp-xr:ecn-sum
a=candidate:1 1 UDP 2130706431 10.0.1.4 8998 typ host
a=candidate:2 1 UDP 1694498815 192.0.2.3 45664 typ srflx raddr
   10.0.1.4 rport 8998
```

This SDP offer offers a single media stream with 3 media payload types. It proposes to use ECN with RTP, with the ICE based initilziation as being prefered over the RTP/RTCP one. Leap of faith is not suggested to be used. The offerer is capable of both setting and reading the ECN bits. In addition the RTCP ECN feedback packet is configured and the RTCP XR ECN summary report. ICE is also proposed with two candidates.

The Answer:

```
v=0
o=jdoe 3502844783 3502844783 IN IP4 198.51.100.235
s=VoIP call
i=SDP offer for VoIP call with ICE and ECN for RTP
b=AS:128
b=RR:2000
b=RS:2500
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
a=ice-options:rtp+ecn
t=0 0
m=audio 53879 RTP/AVPF 97 99
c=IN IP4 198.51.100.235
a=rtpmap:97 G719/48000/1
a=fmtp:97 maxred=160
a=rtpmap:99 PCMA/8000/1
a=maxptime:160
a=ptime:20
a=ecn-capable-rtp: ice ect=0 mode=readonly
a=rtcp-fb:* nack ecn
a=rtcp-fb:* trr-int 1000
a=rtcp-xr:ecn-sum
a=candidate:1 1 UDP 2130706431 198.51.100.235 53879 typ host
```

The answer confirms that only one media stream will be used. One RTP Payload type was removed. ECN capability was confirmed, and the initilization method will be ICE. However, the answerer is only capable of reading the ECN bits, which means that ECN can only be used for RTP flowing from the offerer to the answerer. ECT always set to 0 will be used in both directions. Both the RTCP ECN feedback packet and the RTCP XR ECN summary report will be used.

12.2. Declarative Multicast SDP

The below session describes an any source multicast using session with a single media stream.

```
v=0
o=jdoe 3502844782 3502844782 IN IP4 198.51.100.235
s=Multicast SDP session using ECN for RTP
i=Multicasted audio chat using ECN for RTP
b=AS:128
t=3502892703 3502910700
m=audio 56144 RTP/AVPF 97
c=IN IP4 233.252.0.212/127
a=rtpmap:97 g719/48000/1
a=fmtp:97 maxred=160
a=maxptime:160
a=ptime:20
a=ecn-capable-rtp: rtp mode=readonly; ect=0
a=rtcp-fb:* nack ecn
a=rtcp-fb:* trr-int 1500
a=rtcp-xr:ecn-sum
```

In the above example, as this is declarative we need to require certain functionality. As it is ASM the initliziation method that can work here is the RTP/RTCP based one. So that is indicated. The ECN setting and reading capability to take part of this session is at least read. If one is capable of setting that is good, but not required as one can skip using ECN for anything one send oneself. The ECT value is recommended to be set to 0 always. The ECN usage in this session requires both ECN feedback and the XR ECN summary report, so their usage are also indicated.

13. Open Issues

As this draft is under development some known open issues exist and are collected here. Please consider them and provide input.

1. The negotiation and directionality attribute is going to need some consideration for multi-party sessions when readonly capability might be sufficient to enable ECN for all incoming streams. However, it would beneficial to know if no potential sender support setting ECN.

2. Consider initiation optimizations that allows for multi SSRC sender nodes to still have rapid usage of ECN.

3. Should we report congestion in bytes or packets? RTCP usually does this in terms of packets, but there may be an argument that we want to report bytes for ECN. draft-ietf-tsvwg-byte-pkt-congest is extremely unclear on what is the right approach.

4. We have a saturation problem with the packet loss counters. They do need to continue working even if saturation happens due to long sessions where more lost packets than the counters can handle.

14. Acknowledgments

The authors wish to thank the following persons for their reviews and comments: Thomas Belling, Bob Briscoe, Roni Even, Thomas Frankkila, Christian Groves, Cullen Jennings Tom Van Caenegem, Simo Veikkolainen, Lei Zhu, Christer Holmgren.

15. References

15.1. Normative References

[I-D.ietf-mmusic-ice-options-registry]
Westerlund, M. and C. Perkins, "IANA Registry for Interactive Connectivity Establishment (ICE) Options", draft-ietf-mmusic-ice-options-registry-00 (work in progress), January 2011.

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

[RFC2762] Rosenberg, J. and H. Schulzrinne, "Sampling of the Group Membership in RTP", RFC 2762, February 2000.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3611] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, September 2008.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

15.2. Informative References

[I-D.ietf-avt-rtp-no-op]
Andreasen, F., "A No-Op Payload Format for RTP", draft-ietf-avt-rtp-no-op-04 (work in progress), May 2007.

[I-D.zimmermann-avt-zrtp]
Zimmermann, P., Johnston, A., and J. Callas, "ZRTP: Media Path Key Agreement for Unicast Secure RTP", draft-zimmermann-avt-zrtp-22 (work in progress), June 2010.

[RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, October 2000.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC3540] Spring, N., Wetherall, D., and D. Ely, "Robust Explicit Congestion Notification (ECN) Signaling with Nonces", RFC 3540, June 2003.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003.

[RFC3569] Bhattacharyya, S., "An Overview of Source-Specific Multicast (SSM)", RFC 3569, July 2003.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 2006.

[RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security", RFC 4347, April 2006.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, August 2006.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, February 2008.

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.

[RFC5630] Audet, F., "The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)", RFC 5630, October 2009.

[RFC5751] Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010.

[RFC5760] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback", RFC 5760, February 2010.

Authors' Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

Phone: +46 10 714 82 87
Email: magnus.westerlund@ericsson.com

Ingemar Johansson
Ericsson
Laboratoriegrand 11
SE-971 28 Lulea
SWEDEN

Phone: +46 73 0783289
Email: ingemar.s.johansson@ericsson.com

Colin Perkins
University of Glasgow
School of Computing Science
Glasgow G12 8QQ
United Kingdom

Email: csp@csperkins.org

Piers O'Hanlon
University College London
Computer Science Department
Gower Street
London WC1E 6BT
United Kingdom

Email: p.ohanlon@cs.ucl.ac.uk

Ken Carlberg
G11
1600 Clarendon Blvd
Arlington VA
USA

Email: carlberg@g11.org.uk

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 19, 2011
Q. Wu
F. Xia
R. Even
Huawei
February 15, 2011

RTCP Extension for Feedback Suppression Indication
draft-ietf-avtcore-feedback-supression-rtp-00

Abstract

In a large RTP session using the RTCP feedback mechanism defined in RFC 4585, a media source or middlebox may experience transient overload if some event causes a large number of receivers to send feedback at once. This feedback implosion can be mitigated if the device suffering from overload can send a third party loss report message to the receivers to inhibit further feedback. This memo defines RTCP extensions for third party loss report, to suppress NACK and FIR feedback requests. It also defines associated SDP signalling.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on August 19, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction

RTCP feedback messages [RFC4585] allow the receivers in an RTP session to report events and ask for action from the media source (or a delegated feedback target defined in SSM [RFC5760]). There are cases where multiple receivers may initiate the same, or an equivalent message towards the same media source. When the receiver count is large, this behavior may cause transient overload of the media source, the network or both. This is known as a "feedback storm" or a "NACK storm". One common cause of such a feedback storm is receivers utilizing RTP retransmission [RFC4588] as a packet loss recovery technique based, sending feedback using RTCP NACK messages [RFC4585] without proper dithering of the retransmission requests.

Another use case involves video Fast Update requests. A storm of these feedback messages can occur in conversational multimedia scenarios like Topo-Video-switch-MCU [RFC5117]. In this scenario, packet loss may happen on an upstream link of an intermediate network element such as a Multipoint Control Unit(MCU). Poorly designed receivers that blindly issue fast update requests (i.e., Full Intra Request (FIR) described in [RFC5104]), can cause an implosion of FIR requests from receivers to the same media source.

RTCP feedback storms may cause short term overload and, and in extreme cases to pose a possible risk of increasing network congestion on the control channel (e.g. RTCP feedback), the data channel, or both. It is therefore desirable to provide a way of suppressing unneeded feedback.

One approach to this, suggested in [DVB-IPTV], involves sending a NACK message to the other clients (or receiver) in the same group as the sender of NACK. However sending multicast NACK to the group can not prevent large amount of unicast NACK addressed to the same media source or middlebox, for example when the NACK is used as a retransmission request [RFC4588]. Also NACK is defined as a receiver report sent from a receiver observing a packet loss, therefore it only inform others that sender of NACK detected loss while the case the sender of the feedback has received reports that the indicated packets were lost is not covered. This document specifies a new message for this function. It further is more precise in the intended uses and less likely to be confusing to receivers. It tells receivers explicitly that feedback for a particular packet or frame loss is not needed for a period of time and can provide an early indication before the receiver reacts to the loss and invokes its packet loss repair machinery.

2. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Protocol Overview

This document extends the RTCP feedback messages defined in the Audio-Visual Profile with Feedback (AVPF) and define the Third Party Loss Report message. The Third Party Loss Report message informs the receiver in the downstream path of the middlebox that the sender of the Third Party Loss Report has received reports that the indicated packets were lost and asks a receiver to not send feedback messages for particular packets (indicated by their RTP sequence numbers) independent of whether the receiver detected the packet loss or detected a need for a decoder refresh point.

In order to observe packet loss before the receivers perceive it, one or more intermediate nodes may be placed between the media source and the receivers. These intermediates are variously referred to as Distribution servers, MCUs, RTP translator, or RTP mixers, depending on the precise use case. These intermediaries monitor for packet loss upstream of themselves by checking RTP sequence numbers, just as receivers do. Upon observing (or suspecting) an upstream loss, the intermediary may send Loss Party Loss Report message towards the receivers as defined in this specification.

These intermediate nodes need to take into account such factors as the tolerable application delay, the network dynamics, and the media type. When the packet loss is detected upstream of the intermediary and additional latency is tolerable, the intermediate node may itself send a feedback message asking for the suspected lost packet or ask for the correct decoder refresh point. Because it has already provided the necessary feedback toward the source, the intermediate node can be reasonably certain that it will help the situation by sending a Third Party Loss Report message to all the relevant receivers, thereby indicating to the receivers that they should not transmit feedback messages for a period of time.

Alternatively, the media source may directly monitor the amount of feedback requests it receives, and send Third Party Loss Report messages to the receivers.

When a receiver gets such a Third Party Loss Report message, it should refrain from sending a feedback request (e.g., NACK or FIR) for the missing packets reported in the message for a period of time.

A receiver may still have sent a Feedback message according to the AVPF scheduling algorithm of [RFC4585]before receiving a Third Party Loss Report message, but further feedback messages for those sequence numbers will be suppressed by this technique for a period of time. Nodes that do not understand the Third Party Loss Report message will ignore it, and might therefore still send feedback according to the AVPF scheduling algorithm of [RFC4585]. The media source or intermediate nodes cannot assume that the use of a Third Party Loss Report message actually reduces the amount of feedback it receives.

RTCP Third Party Loss Report follows the similar format of message type as RTCP NACK. But unlike RTCP NACK, the third party loss report is defined as an indication that the sender of the feedback has received reports that the indicated packets were lost and conveys the packet receipt/loss events at the sequence number level from the middlebox to the receivers in the downstream path of middlebox while NACK [RFC4585]just indicates that the sender of the NACK observed that these packets were lost. The Third Party Loss Report message can also be generated by RTP middleboxs that has not seen the actual packet loss and sent to the corresponding receivers. Intermediaries downstream of an intermediary detecting loss obviously SHOULD NOT initiate their own additional Third Party Loss Report messages for the same packet sequence numbers. They may either simply forward the Third Party Loss Report message received from upstream, or replace it with a Third Party Loss Report message that reflects the loss pattern they have themselves seen. The Third Party Loss Report does not have the retransmission request [rfc4588] semantics.

Since Third Party Loss Report interacts strongly with repair timing, it has to work together with feedback to not adversely impact the repair of lost source packets. One example is the middle box gets the retransmitted packet by sending a NACK upstream and sent it downstream. This retransmitted packet was lost on the downstream link. In order to deal with this, the downstream receiver can start a timeout in which it expected to get a retransmission packet. When this timeout expires and there is no retransmitted packet or a new third party loss report message, it can take its normal behavior as if there is no current retransmission suppression. In some cases where the loss was detected and repair initiated much closer to the source, the delay for the receiver to recover from packet loss can be reduced through the combination of intermediary feedback to the source and Third Party Loss Report downstream. In all (properly operating) cases, the risk of increasing network congestion is decreased.

4. RTCP Feedback Report Extension

This document registers two new RTCP Feedback messages for Third Party Loss Report. Applications that are employing one or more loss-repair methods MAY use Third Party Loss Report together with their existing loss-repair methods either for every packet they expect to receive, or for an application-specific subset of the RTP packets in a session. In other words, receivers MAY ignore Third Party Loss Report messages, but SHOULD react to them unless they have good reason to still send feedback messages despite having been requested to suppress them.

4.1. Transport Layer Feedback: Third-party Loss Report

This Third Party Loss Report message is an extension to the RTCP Transport Layer Feedback Report and identified by RTCP packet type value PT=RTPFB and FMT=TBD.

The FCI field MUST contain one or more entries of transport layer third party loss Early Indication (TLLEI). Each entry applies to a different media source, identified by its SSRC.

The Feedback Control Information (FCI) for TLLEI uses the similar format of message Types defined in the section 4.3.1.1 of [RFC5104]. The format is shown in Figure 1.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            PID                |             BLP               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Message Format for the Third Party Loss Report

Packet ID (PID): 16 bits

The PID field is used to specify a lost packet. The PID field refers to the RTP sequence number of the lost packet.

bitmask of proceeding lost packets (BLP): 16 bits

The BLP allows for reporting losses of any of the 16 RTP packets immediately following the RTP packet indicated by the PID. The BLP’s definition is identical to that given in [RFC4585].

4.2. Payload Specific Feedback: Third-party Loss Report

This message is an extension to the RTCP Payload Specific Feedback report and identified by RTCP packet type value PT=PSFB and FMT=TBD.

The FCI field MUST contain a Payload Specific Third Party Loss Early Indication (PSLEI) entry. Each entry applies to a different media source, identified by its SSRC.

The Feedback Control Information (FCI) for PSLEI uses the similar format of message Types defined in the section 4.3.1.1 of [RFC5104]. The format is shown in Figure 2.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              SSRC                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Seq nr.   |                    Reserved                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Message Format for the Third Party Loss Report

SSRC (32 bits):

The SSRC value of the media source that is requested to send a decoder refresh point.

Seq nr:8bits Command sequence number. The sequence number space is unique for each pairing of the SSRC of command source and the SSRC of the command target. The sequence number SHALL be increased by 1 modulo 256 for each new request.

Reserved: 24 bits

All bits SHALL be set to 0 by the media source and SHALL be ignored on reception.

5. SDP Signaling

A new feedback value "tplr" needs to be defined for the Third Party Loss Report message to be used with Session Description Protocol (SDP) [RFC4566] using the Augmented Backus-Naur Form (ABNF) [RFC4585].

The "tplr" feedback value SHOULD be used with parameters that indicate the third party loss supported. In this document, we define two such parameter, namely:

- o "tllei" denotes support of transport layer third party loss early indication (fsei).
- o "pslei" denotes support of payload specific third party loss early indication.

In the ABNF for rtcp-fb-val defined in [RFC4585], there is a placeholder called rtcp-fb-id to define new feedback types. "tplr" is defined as a new feedback type in this document, and the ABNF for the parameters for tplr is defined here (please refer to section 4.2 of [RFC4585] for complete ABNF syntax).

```
      rtcp-fb-val        =/ "tplr" rtcp-fb-tplr-param
      rtcp-fb-tplr-param  = SP "tllei";transport layer third party loss early
 indication
                          / SP "pslei";payload specific third party loss earl
y indication
                          / SP token [SP byte-string]
                                    ; for future commands/indications
   byte-string = <as defined in section 4.2 of [RFC4585] >
```

Refer to Section 4.2 of [RFC4585] for a detailed description and the full syntax of the "rtcp-fb" attribute.

6. Example Use Cases

The operation of feedback suppression is similar for all types of RTP sessions and topologies [RFC5117], however the exact messages used and the scenarios in which suppression is employed differ for various use cases. The following sections outline the intended use cases of using Third Party Loss Report for feedback suppression and give an overview of the particular mechanisms.

6.1. Source Specific Multicast (SSM) use case

In SSM RTP sessions as described in [RFC5760], one or more Media Sources send RTP packets to a Distribution Source. The Distribution Source relays the RTP packets to the receivers using a source-specific multicast group.

In order to avoid the forms of Feedback implosion described in section 1,the distribution source should be told that the indicated packets were lost. How the distribution source know the indicated packets were lost is beyond of scope of this document. When upstream link or downstream aggregate link packet loss occurs, the distribution source creates a Third Party Loss Report and sent it to all the RTP receivers, over the multicast channel. Another possibility is when there may be multiple distribution sources placed between the media source and the receivers, the upstream distribution source may inform downstream distribution sources of the detected packet loss using Third Party Loss Report messages. In response, the downstream distribution sources forward Third Party Loss Report received from upstream to all the RTP receivers, over the multicast channel. This Third Party Loss Report message tells the receivers that the sender of the third party loss report has received reports that the indicated packets were lost. The distribution source then can (optionally) ask for the lost packets from the media source on behalf of all the RTP receivers. The lost packets will either be forthcoming from distribution source, or it irretrievably lost such that there is nothing to be gained by the receiver sending a NACK to the media source.

The distribution source must be able to communicate with all group members in order for either mechanism to be effective at suppressing feedback.

As outlined in the [RFC5760], there are two Unicast Feedback models that may be used for reporting, - the Simple Feedback model and the Distribution Source Feedback Summary Model. The RTCP Feedback extension for Third Party Loss Report specified in the Section 4 of this document will work in both Feedback models. Details of operation in each are specified below.

6.1.1. Simple Feedback Model

In the simple Feedback Model, NACKs from the receiver observing the loss will be reflected to the other receivers, and there’s no need for distribution source to create the third-party loss report. The distribution source that has not seen the actual packet loss should pass through any Third Party Loss Report message it receives from the upstream direction.

This RTCP Third Party Loss Report message lets the receivers know that the sender of the Third party Loss Report has received reports that the indicated packets were lost and feedback for this packet loss is not needed and should not be sent to the media source(s). If the media source(s) are part of the SSM group for RTCP packet reflection, the Distribution Source must filter this packet out. If the media source(s) are not part of the SSM group for RTCP packets, the Distribution Source must not forward this RTCP Third Party Loss Report message to the media source(s).

6.1.2. Distribution Source Feedback Summary Model

In the distribution source feedback summary model, there may be multiple distribution sources and the Loss Detection instances are distributed into different distribution sources. In some cases, these Loss Detection instances for the same session can exist at the same time, e.g., one Loss Detection instance is implemented in the upstream distribution source A, a second Loss Detection instance for the same session is part of feedback target A and feedback target B respectively within the distribution source B. The distribution source B is placed in the path between distribution A and downstream receivers. In this section, we focus on this generic case to discuss the distribution Source Feedback Summary Model.

The distribution source A must listen on the RTP channel for data. When the distribution source A observes RTP packets from a media source are not consecutive by checking the sequence number of packets, the distribution source A generates the new RTCP Third Party Loss Report message described in the Section 4, and then send it to receivers in the downstream path via the multicast channel. Note that the distribution source A must use its own SSRC value as packet sender SSRC for transmitting the new RTCP Third Party Loss Report message.

a second detection instance within the Distribution Source B must also listen for RTCP data sent to the RTCP port. Upon receiving the RTCP Third Party Loss Report from the Distribution Source A, the distribution source B needs to check whether it sees upstream third party loss report from distribution source A reporting the same event. If the upstream Third Party Loss Report reports the different event, the distribution source B passes through any Third Party Loss Report message it receives from the upstream direction. If the same event is reported from distribution source A, the distribution source B replaces it with the summary Third Party Loss Report with the information summarization received from two loss detection instances within the Distribution Source B. In order to reduce the processing load at the distribution source, each loss detection instance may provide preliminary summarization report.

During the summary third party loss report creating, the Distribution Source B must use its own SSRC value as packet sender SSRC for transmitting summarization information and MUST perform proper SSRC collision detection and resolution.

The distribution source B may send this new RTCP summary third party loss report described in the Section 4to the group on the multicast RTCP channel and meanwhile send a packet loss request to the media source.

In some case, the distribution source B may receive RTCP NACK messages from the receivers behind the Distribution Source before the distribution source detects the packet loss which may cause potential Feedback implosion. In such case, the distribution source B may filter them out if it already detected the same loss or sent a packet loss request for the missing packet to the media source.

When the host receives the RTCP Third Party Loss Report message, if the host understands this message it will not send packet loss request (e.g., NACK) for the missing packets reported in the message. If it did not understand this new message, the host MAY send packet loss request(e.g., NACK messages) to the specified media source.

6.2. Unicast based Rapid Acquisition of Multicast Stream (RAMS) use case

The typical RAMS architecture [I-D.ietf-avt-rapid-acquisition-for-rtp]may have several Burst/ Retransmission Sources(BRS) behind the multicast source (MS) These BRSes will receive the multicast SSM stream from the media source. If one of the BRSes detects packet loss (i.e., First loss in Figure 3) on its upstream link between the MS and BRS, but the others BRSes have not, as the packet loss took place on SSM tree branch that does not impact the other BRSes. In such case, the BRSes with loss detection functionality support cannot detect packet loss at their upstream link, therefore these BRSes will not create new Third Party Loss Report message and send it to receivers in their downstream path. If the BRS impacted by packet loss has loss detection support, the BRS MAY choose to create new Third Party Loss Report message and send it to the receivers in the downstream link. Note that BRS must use its own SSRC as packet sender SSRC for transmitting the feedback suppress message.

The BRS may also send a NACK upstream to request the retransmitted packet. Upon receiving the retransmitted packet, the BRS sent it downstream. Note that this retransmitted packet may get lost (i.e., second loss in the Figure 3) on the downstream link. In order to deal with this issue, the downstream receiver can start a timeout clock in which it expected to get a retransmission packet. When this timeout expires and there is no retransmitted packet or a new Third Party Loss Report message, it can take its normal behavior as if there is no current retransmission suppression in place.

```
                            First    +------------+            +----------+
                            loss     |Burst and   |Second Loss |          |
                         +-----X-----| Retrans.   |----X------>|          |
                         | Upstream  |Source1(BRS)| Downstream |          |
           Link close    | link 1    +------------+ link 1     |          |
           to multicast  |                                     |          |
           source        |                                     |          |
                |        |                                     |          |
                |        |           +------------+            |   RTP    |
   +---------+  |  +-----++          |Burst and   |            | Receiver |
   |Multicast|  V| |      +----------| Retrans.   |----------->|          |
   | Source  +-----|Router|Upstream  |Source2(BRS)| Downstream |  RTP_Rx  |
   +---------+     |      |link 2    +------------+ link 2     |          |
                   +-----++                                    |          |
                         |                                     |          |
                         |                                     |          |
                         |                                     |          |
                         |           +------------+            |          |
                         |           |Burst and   |            |          |
                         +-----------+ Retrans.   |----------->|          |
                           Upstream  |Source k(BRS| Downstream |          |
                           link k    +------------+ link k     +----------+
```

Figure 3: RAMS Use Case

6.3. RTP transport translator use case

A Transport Translator (Topo-Trn-Translator), as defined in [RFC5117] is typically forwarding the RTP and RTCP traffic between RTP clients, for example converting between multicast and unicast for domains that do not support multicast. The translator can identify packet loss from the upstream and send the Third Party Loss Report message to the unicast receivers. Note that the translator must be a participant in the session and can then use it’s own SSRC as packet sender SSRC for transmitting the Third Party Loss Report message

6.4. Multipoint Control Unit (MCU) use case

In point to multipoint topologies using video switching MCU (Topo-Video-switch-MCU) [RFC5117], the MCU typically forwards a single media stream to each participant, selected from the available input streams. The selection of the input stream is often based on voice activity in the audio-visual conference, but other conference management mechanisms (like presentation mode or explicit floor control) exist as well.

In this case the MCU may detect packet loss from the sender or may decide to switch to a new source. In both cases the receiver may lose synchronization with the video stream and may send a FIR request. If the MCU itself can detect the mis-synchronization of the video, the MCU can send the FIR suppression message to the receivers and send a FIR request to the video source. As suggested in RFC 5117, this topology is better implemented as an Topo-mixer, in which case the mixer's SSRC is used as packet sender SSRC for transmitting Third Party Loss Report message.

7. Security Considerations

The defined messages have certain properties that have security implications. These must be addressed and taken into account by users of this protocol.

Spoofed or maliciously created feedback messages of the type defined in this specification can have the following implications:

Sending Third Party Loss Report with wrong sequence number of lost packet that makes missing RTP packets can not be compensated.

To prevent these attacks, there is a need to apply authentication and integrity protection of the feedback messages. This can be accomplished against threats external to the current RTP session using the RTP profile that combines Secure RTP [RFC3711] and AVPF into SAVPF [RFC5124].

Note that middleboxes that are not visible at the RTP layer that wish to send Third Party Loss Reports on behalf of the media source can only do so if they spoof the SSRC of the media source. This is difficult in case SRTP is in use. If the middlebox is visible at the RTP layer, this is not an issue, provided the middlebox is part of the security context for the session.

Also note that endpoints that receive a Third Party Loss Report would be well-advised to ignore it, unless it is authenticated via SRTCP or similar. Accepting un-authenticated Third Party Loss Report can lead to a denial of service attack, where the endpoint accepts poor quality media that could be repaired.

8. IANA Consideration

New feedback type and New parameters for RTCP Third Party Loss Report are subject to IANA registration. For general guidelines on IANA considerations for RTCP feedback, refer to [RFC4585].

This document assigns one new feedback type value x in the RTCP feedback report registry to "Third Party Loss Report" with the following registrations format:

```
Name:            TPLR
Long Name:       Third Party Loss Report
Value:           TBD
Reference:       This document.
```

This document also assigns the parameter value y in the RTCP TPLR feedback report Registry to " Transport Layer Third Party Loss Early Indication ", with the following registrations format:

```
Name:           TLLEI
Long name:      Transport Layer Third Party Loss Early Indication
Value:          TBD
Reference:      this document.
```

This document also assigns the parameter value z in the RTCP TPLR feedback report Registry to "Payload Specific Third Party Loss Early Indication ", with the following registrations format:

```
Name:           PSLEI
Long name:      Payload Specific Third Party Loss Early Indication
Value:          TBD
Reference:      this document.
```

The contact information for the registrations is:

Qin Wu
sunseawq@huawei.com
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012, China

9. Acknowledgement

The authors would like to thank David R Oran, Ali C. Begen, Colin Perkins,Tom VAN CAENEGEM, Ingemar Johansson S, Bill Ver Steeg, Jonathan Lennox, WeeSan Lee for their valuable comments and suggestions on this document.

10. References

10.1. Normative References

[RFC5760] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast

Sessions with Unicast Feedback", RFC 5760, February 2010.

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

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.

[RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, July 2006.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, February 2008.

10.2. Informative References

[RFC5740] Adamson, B., Bormann, C., Handley, M., and J. Macker, "NACK-Oriented Reliable Multicast (NORM) Transport Protocol", November 2009.

[DVB-IPTV] ETSI Standard, "Digital Video Broadcasting(DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks", ETSI TS 102 034, V1.4.1 , August 2009.

[I-D.ietf-avt-rapid-acquisition-for-rtp]
Steeg, B., Begen, A., Caenegem, T., and Z. Vax, "Unicast-Based Rapid Acquisition of Multicast RTP Sessions", November 2010.

[I-D.hunt-avt-monarch-01]
Hunt, G. and P. Arden, "Monitoring Architectures for RTP", August 2008.

[I-D.ietf-pmol-metrics-framework-02]
Clark, A., "Framework for Performance Metric Development".

Authors' Addresses

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: sunseawq@huawei.com

Frank Xia
Huawei
1700 Alma Dr. Suite 500
Plano, TX 75075
USA

Phone: +1 972-509-5599
Email: xiayangsong@huawei.com

Roni Even
Huawei
14 David Hamelech
Tel Aviv 64953
Israel

Email: even.roni@huawei.com

AVT Core Working Group   V. Singh
Internet-Draft   T. Karkkainen
Intended status: Experimental   J. Ott
Expires: September 15, 2011   S. Ahsan
                                                        Aalto University
                                                               L. Eggert
                                                                   Nokia
                                                          March 14, 2011

Multipath RTP (MPRTP)
draft-singh-avtcore-mprtp-01

Abstract

The Real-time Transport Protocol (RTP) is used to deliver real-time content and, along with the RTP Control Protocol (RTCP), forms the control channel between the sender and receiver. However, RTP and RTCP assume a single delivery path between the sender and receiver and make decisions based on the measured characteristics of this single path. Increasingly, endpoints are becoming multi-homed, which means that they are connected via multiple Internet paths. Network utilization can be improved when endpoints use multiple parallel paths for communication. The resulting increase in reliability and throughput can also enhance the user experience. This document extends the Real-time Transport Protocol (RTP) so that a single session can take advantage of the availability of multiple paths between two endpoints.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

Multi-homed endpoints are becoming common in today's Internet, e.g., devices that support multiple wireless access technologies such as 3G and Wireless LAN. This means that there is often more than one network path available between two endpoints. Transport protocols, such as RTP, have not been designed to take advantage of the availability of multiple concurrent paths and therefore cannot benefit from the increased capacity and reliability that can be achieved by pooling their respective capacities.

Multipath RTP (MPRTP) is an OPTIONAL extension to RTP [1] that allows splitting a single RTP stream into multiple subflows that are transmitted over different paths. In effect, this pools the resource capacity of multiple paths. Multipath RTCP (MPRTCP) is an extension to RTCP, it is used along with MPRTP to report per-path sender and receiver characteristics.

Other IETF transport protocols that are capable of using multiple paths include SCTP [9], MPTCP MPTCP [10] and SHIM6 [11]. However, these protocols are not suitable for realtime communications.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [2].

1.2. Terminology

- Endpoint: host either initiating or terminating an RTP connection.
- Interface: logical or physical component that is capable of acquiring a unique IP address.
- Path: sequence of links between a sender and a receiver. Typically, defined by a set of source and destination addresses.
- Subflow: flow of RTP packets along a specific path, i.e., a subset of the packets belonging to an RTP stream. The combination of all RTP subflows forms the complete RTP stream. Typically, a subflow would map to a unique path, i.e., each combination of IP addresses and port pairs (4-tuple) is a unique subflow.

1.3. Use-cases

The primary use-case for MPRTP is transporting high bit-rate streaming multimedia content between endpoints, where at least one is multi-homed. Such endpoints could be residential IPTV devices that connect to the Internet through two different Internet service providers (ISPs), or mobile devices that connect to the Internet through 3G and WLAN interfaces. By allowing RTP to use multiple paths for transmission, the following gains can be achieved:

- Higher quality: Pooling the resource capacity of multiple Internet paths allows higher bit-rate and higher quality codecs to be used. From the application perspective, the available bandwidth between the two endpoints increases.

- Load balancing: Transmitting one RTP stream over multiple paths can reduce the bandwidth usage, compared to transmitting the same stream along a single path. This reduces the impact on other traffic.

- Fault tolerance: When multiple paths are used in conjunction with redundancy mechanisms (FEC, re-transmissions, etc.), outages on one path have less impact on the overall perceived quality of the stream.

A secondary use-case for MPRTP is transporting Voice over IP (VoIP) calls to a device with multiple interfaces. Again, such an endpoint could be a mobile device with multiple wireless interfaces. In this case, little is to be gained from resource pooling, i.e., higher capacity or load balancing, because a single path should be easily capable of handling the required load. However, using multiple concurrent subflows can improve fault tolerance, because traffic can shift between the subflows when path outages occur. This results in very fast transport-layer handovers that do not require support from signaling.

2. Goals

This section outlines the basic goals that multipath RTP aims to meet. These are broadly classified as Functional goals and Compatibility goals.

2.1. Functional goals

Allow unicast RTP session to be split into multiple subflows in order to be carried over multiple paths. This may prove beneficial in case of video streaming.

- o Increased Throughput: Cumulative capacity of the two paths may meet the requirements of the multimedia session. Therefore, MPRTP MUST support concurrent use of the multiple paths.

- o Improved Reliability: MPRTP SHOULD be able to send redundant packets or re-transmit packets along any available path to increase reliability.

The protocol SHOULD be able to open new subflows for an existing session when new paths appear and MUST be able to close subflows when paths disappear.

2.2. Compatibility goals

MPRTP MUST be backwards compatible; an MPRTP stream needs to fall back to be compatible with legacy RTP stacks if MPRTP support is not successfully negotiated.

- o Application Compatibility: MPRTP service model MUST be backwards compatible with existing RTP applications, i.e., an MPRTP stack MUST be able to work with legacy RTP applications and not require changes to them. Therefore, the basic RTP APIs MUST remain unchanged, but an MPRTP stack MAY provide extended APIs so that the application can configure any additional features provided by the MPRTP stack.

- o Network Compatibility: individual RTP subflows MUST themselves be well-formed RTP flows, so that they are able to traverse NATs and firewalls. This MUST be the case even when interfaces appear after session initiation. Interactive Connectivity Establishment (ICE) [3] MAY be used for discovering new interfaces or performing connectivity checks.

3. RTP Topologies

RFC 5117 [12] describes a number of scenarios using mixers and translators in single-party (point-to-point), and multi-party (point-to-multipoint) scenarios. RFC 3550 [1] (Section 2.3 and 7.x) discuss in detail the impact of mixers and translators on RTP and RTCP packets. MPRTP assumes that if a mixer or translator exists in the network, then either all of the multiple paths or none of the multiple paths go via this component.

4. MPRTP Architecture

In a typical scenario, an RTP session uses a single path. In an

MPRTP scenario, an RTP session uses multiple subflows that each use a different path. Figure 1 shows the difference.

```
+--------------+                Signaling           +--------------+
|              |----------------------------------->|              |
|    Client    |<-----------------------------------|    Server    |
|              |           Single RTP flow          |              |
+--------------+                                    +--------------+

+--------------+              Signaling             +--------------+
|              |----------------------------------->|              |
|    Client    |<-----------------------------------|    Server    |
|              |<-----------------------------------|              |
+--------------+             MPRTP subflows         +--------------+
```

Figure 1: Comparison between traditional RTP streaming and MPRTP

```
+-----------------------+         +-------------------------------+
|      Application      |         |          Application          |
+-----------------------+         +-------------------------------+
|                       |         |             MPRTP             |
+          RTP          +         +- - - - - - - -+- - - - - - - -+
|                       |         |  RTP subflow  |  RTP subflow  |
+-----------------------+         +---------------+---------------+
|          UDP          |         |      UDP      |      UDP      |
+-----------------------+         +---------------+---------------+
|           IP          |         |       IP      |       IP      |
+-----------------------+         +---------------+---------------+
```

Figure 2: MPRTP Architecture

Figure 2 illustrates the differences between the standard RTP stack and the MPRTP stack. MPRTP receives a normal RTP session from the application and splits it into multiple RTP subflows. Each subflow is then sent along a different path to the receiver. To the network, each subflow appears as an independent, well-formed RTP flow. At the receiver, the subflows are combined to recreate the original RTP session. The MPRTP layer performs the following functions:

o Path Management: The layer is aware of alternate paths to the other host, which may, for example, be the peer’s multiple interfaces. So that it is able to send differently marked packets along separate paths. MPRTP also selects interfaces to send and receive data. Furthermore, it manages the port and IP address pair bindings for each subflow.

- o Packet Scheduling: the layer splits a single RTP flow into multiple subflows and sends them across multiple interfaces (paths). The splitting MAY BE done using different path characteristics.

- o Subflow recombination: the layer creates the original stream by recombining the independent subflows. Therefore, the multipath subflows appear as a single RTP stream to applications.

4.1. Relationship of MPRTP with Session Signaling

Session signaling (e.g., SIP [13], RTSP [14]) SHOULD be done over a failover-capable or multipath-capable transport for e.g., SCTP [9] or MPTCP [10] instead of TCP or UDP.

5. Example Media Flow Diagrams

There may be many complex technical scenarios for MPRTP, however, this memo only considers the following two scenarios: 1) a unidirectional media flow that represents the streaming use-case, and 2) a bidirectional media flow that represents a conversational use-case.

5.1. Streaming use-case

In the unidirectional scenario, the receiver (client) initiates a multimedia session with the sender (server). The receiver or the sender may have multiple interfaces and both endpoints are MPRTP-capable. Figure 3 shows this scenario. In this case, host A has multiple interfaces. Host B performs connectivity checks on host A’s multiple interfaces. If the interfaces are reachable, then host B streams multimedia data along multiple paths to host A. Moreover, host B also sends RTCP Sender Reports (SR) for each subflow and host A responds with a standard RTCP Receiver Report (RR) for the overall session and receiver statistics for each subflow. Host B distributes the packets across the subflows based on the individually measured path characteristics.

Alternatively, to reduce media startup time, host B may start streaming multimedia data to host A’s initiating interface and then perform connectivity checks for the other interfaces. This method of updating a single path session to a multipath session is called "multipath session upgrade".

```
           Host A                           Host B
   -----------------------                ----------
   Address A1   Address A2                Address B1
   -----------------------                ----------
       |            Session Setup             |
       |------------------------------------->|  connections at endpoint
       |<-------------------------------------|  may be "preloaded"
       |            |                         |  (e.g., with ICE)
       |            |                         |
       |       (RTP data B1->A1, B1->A2)      |
       |<=====================================|
       |            |<========================|
       |            |                         |
       Note: there may be more scenarios.
```

Figure 3: Unidirectional media flow

5.2. Conversational use-case

In the bidirectional scenario, multimedia data flows in both directions. The two hosts exchange their lists of interfaces with each other and perform connectivity checks. Communication begins after each host finds suitable address, port pairs. Interfaces that receive data send back RTCP receiver statistics for that path (based on the 4-tuple). The hosts balance their multimedia stream across multiple paths based on the per path reception statistics and its own volume of traffic. Figure 4 describes an example of a bidirectional flow.

```
          Host A                                Host B
  -----------------------              -----------------------
  Address A1   Address A2              Address B1   Address B2
  -----------------------              -----------------------
    |            |                       |            |
    |            Session Setup           |            | connections at
    |----------------------------------->|            | the endpoint may
    |<-----------------------------------|            | be "preloaded"
    |            |                       |            | (e.g., ICE)
    |            |                       |            |
    |    (RTP data B1<->A1, B2<->A2)     |            |
    |<===================================|            |
    |            |<===================================|
    |===================================>|            |
    |            |===================================>|
    |            |                       |            |
      Note: the address pairs may have other permutations,
      and they may be symmetric or asymmetric combinations.
```

Figure 4: Bidirectional flow

5.3. Challenges with Multipath Interface Discovery

For some applications, where the user expects immediate playback, e.g., High Definition Media Streaming or IPTV, it may not be possible to perform connectivity checks within the given time bound. In these cases, connectivity checks MAY need to be done ahead of time.

[Open Issue: ICE or any other system would have to be aware of the endpoint's interfaces ahead of time].

6. MPRTP Functional Blocks

This section describes some of the functional blocks needed for MPRTP. We then investigate each block and consider available mechanisms in the next section.

1. Session Setup: Multipath session setup is an upgrade or add-on to a typical RTP session. Interfaces may appear or disappear at anytime during the session. To preserve backward compatibility with legacy applications, a multipath session MUST look like a bundle of individual RTP sessions.

2. Expanding RTP: For a multipath session, each subflow MUST look like an independent RTP flow, so that individual RTCP messages can be generated per subflow. Furthermore, MPRTP splits the single multimedia stream into multiple subflows based on path characteristics (e.g. RTT, loss-rate, receiver rate, bandwidth-delay product etc.) and dynamically adjusts the load on each link.

3. Adding Interfaces: Interfaces on the host need to be regularly discovered and signaled. This can be done at session setup and/or during the session. When discovering and receiving new interfaces, the MPRTP layer needs to select address and port pairs.

4. Expanding RTCP: MPRTP MUST recombine RTCP reports from each path to re-create a single RTCP message to maintain backward compatibility with legacy applications.

5. Maintenance and Failure Handling: In a multi-homed endpoint interfaces may appear and disappear. If this happens at the sender, it has to re-adjust the load on the available links. On the other hand, if this occurs on the receiver, then the multimedia data transmitted by the sender to those interfaces is

lost. This data may be re-transmitted along a different path i.e., to a different interface on the receiver. Furthermore, the receiver has to explicitly signal the disappearance of an interface, or the sender has to detect it. [Open Issue: What happens if the interface that setup the session disappears? does the control channel also failover? re-start the session?]

6. Teardown: The MPRTP layer releases the occupied ports on the interfaces.

7. Available Mechanisms within the Functional Blocks

This section discusses some of the possible alternatives for each functional block mentioned in the previous section.

7.1. Session Setup

MPRTP session can be set up in many possible ways e.g., during handshake, or upgraded mid-session. The capability exchange may be done using out-of-band signaling (e.g., SDP [15] in SIP [13], RTSP [14]) or in-band signaling (e.g., RTP/RTCP header extension). Furthermore, ICE [3] may be used for discovering and performing connectivity checks during session setup.

7.2. Expanding RTP

RTCP [1] is generated per media session. However, with MPRTP, the media sender spreads the RTP load across several interfaces. The media sender SHOULD make the path selection, load balancing and fault tolerance decisions based on the characteristics of each path. Therefore, apart from normal RTP sequence numbers defined in [1], the MPRTP sender MUST add subflow-specific sequence numbers to help calculate fractional losses, jitter, RTT, playout time, etc., for each path and a subflow identifier to associate the characteristics with a path. The RTP header extension for MPRTP is shown in Section 9).

7.3. Adding New Interfaces

When interfaces appear and disappear mid-session, ICE [3] may be used for discovering interfaces and performing connectivity checks. However, MPRTP may require a capability re-negotiation (using SDP) to include all these new interfaces. This method is referred to as out-of-band multipath advertisement.

Alternatively, when new interfaces appear, the interface advertisements may be done in-band using RTP/RTCP extensions. The endpoints perform connectivity checks (see Figure 5 for more details). If the connectivity packets are received by the peers, then multimedia data can flow between the new address, port pairs.

7.4. Expanding RTCP

To provide accurate per path information an MPRTP endpoint MUST send (SR/RR) report for each unique subflow along with the overall session RTCP report. Therefore, the additional subflow reporting affects the RTCP bandwidth and the RTCP reporting interval for each subflow. RTCP report scheduling for each subflow may cause a problem for RTCP recombination and reconstruction in cases when 1) RTCP for a subflow is lost, and 2) RTCP for a subflow arrives later than the other subflows. (There may be other cases as well.)

The sender distributes the media across different paths using the per path RTCP reports. However, this document doesn’t cover algorithms for congestion control or load balancing.

7.5. Checking and Failure Handling

[Note: If the original interface that setup the session disappears then does the session signaling failover to another interface? Can we recommend that SIP/RTSP be run over MPTCP, SCTP].

8. MPRTP Protocol

To enable a quick start of a multimedia session, a multipath session MUST be upgraded from a single path session. Therefore, no explicit changes are needed in multimedia session setup and the session can be setup as before.

```
           Host A                                   Host B
   -----------------------                  -----------------------
   Address A1   Address A2                  Address B1   Address B2
   -----------------------                  -----------------------
       |            |                           |             |
       |            |      (1)                  |             |
       |--------------------------------------->|             |
       |<---------------------------------------|             |
       |            |      (2)                  |             |
       |<=======================================|             |
       |<=======================================|     (3)     |
       |            |      (4)                  |             |
       |<=======================================|             |
       |<=======================================|             |
       |<=======================================|             |
       |            |      (5)                  |             |
       |- - - - - - - - - - - - - - - - - - - ->|             |
       |<=======================================|     (6)     |
       |<=======================================|             |
       |            |<========================================|
       |<=======================================|             |
       |            |<========================================|

   Key:
   |  Interface
   ---> Signaling Protocol
   <=== RTP Packets
   - -> RTCP Packet
```

Figure 5: MPRTP New Interface

8.1. Overview

The bullet points explain the different steps shown in Figure 5 for upgrading a standard single path multimedia session to multipath session.

(1) The first two interactions between the hosts represents the standard session setup. This may be SIP or RTSP.

(2) Following the setup, like in a conventional RTP scenario, host B using interface B1 starts to stream data to host A at interface A1.

(3) Host B is an MPRTP-capable media sender and becomes aware of another interface B2.

(4) Host B advertises the multiple interface addresses using an RTCP header extensions.

(5) Host A is an MPRTP-capable media receiver and becomes aware of another interface A2. It advertises the multiple interface addresses using an RTCP extension.

Side note, even if an MPRTP-capable host has only one interface, it SHOULD respond to the advertisement with its single interface.

(6) Each host receives information about the additional interfaces and performs the connectivity tests (not shown in figure). If the paths are reachable then the host starts to stream the multimedia content using the additional paths.

8.1.1. Subflow or Interface advertisement

To advertise the multiple interfaces, an MPRTP-capable endpoint MUST add the MPRTP Interface Advertisement defined in Figure 6 with the RTCP Sender Report (SR). Each unique address is encapsulated in an Interface Advertisement block and contains the IP address, RTP and RTCP port addresses. The Interface Advertisement blocks are ordered based on a decreasing priority level. On receiving the MPRTP Interface Advertisement, an MPRTP-capable receiver MUST respond with its own set of interfaces.

If the sender and receiver have only one interface, then the endpoints MUST respond with the default IP, RTP port and RTCP port addresses. If an endpoint receives an RTCP report without the MPRTP Interface Advertisement, then the endpoint MUST assume that the other endpoint is not MPRTP capable.

8.1.2. Path selection

After MPRTP support has been discovered and interface advertisements have been exchanged, the sender MUST initiate connectivity checks to determine which interface pairs offer valid paths between the sender and the receiver. Each combination of IP addresses and port pairs (4-tuple) is a unique subflow. An endpoint MUST associate a Subflow ID to each unique subflow.

To initiate a connectivity check, the endpoints send an RTP packet using the appropriate MPRTP extension header (See Figure 10), associated Subflow ID and no RTP payload. The receiving endpoint replies to each connectivity check with an RTCP packet with the appropriate packet type (See Figure 7) and Subflow ID. After the endpoint receives the reply, the path is considered a valid candidate for sending data. An endpoint MAY choose to do any number of connectivity checks for any interface pairs at any point in a session.

[Open Issue: How should the endpoint adjust the RTCP Reporting interval/schedule the RTCP packet on receiving a connectivity check containing a new Subflow ID? Editor: One option is send immediately as defined in [4]. Another option is the RTCP timing defined in [16].]

8.1.3. Opening subflows

The sender MAY open any number of subflows from the set of candidate subflows after performing connectivity checks. To use the subflow, the sender simply starts sending the RTP packets with an MPRTP extension shown in Figure 9. The MPRTP extension carries a mapping of a subflow packet to the aggregate flow. Namely, sequence numbers and timestamps associated with the subflow.

An endpoint MAY use all or a subset of candidate subflows for sending media packets. To avoid redoing the connectivity checks the endpoint MAY send keep-alive MPRTP packets (see Section 9.2.3) to the passive subflows to keep the NAT bindings alive.

[Open Issue: How to differentiate between Passive and Active connections? Editor: Active paths get "regular flow" of media packets while passive paths are for failover of active paths.]

[Open Issue: How to keep a passive connection alive, if not actively used? Alternatively, what is the maximum timeout? Editor: keep-alive for ICE/NAT bindings should not be less than 15 seconds [3].]

8.2. RTP Transmission

The MPRTP layer SHOULD associate an RTP packet with a subflow based on a scheduling strategy. The scheduling strategy may either choose to augment the paths to create higher throughput or use the alternate paths for enhancing resilience or error-repair. Due to the changes in path characteristics, an MPRTP sender can change its scheduling strategy during an ongoing session. The MPRTP sender MUST also populate the subflow specific fields described in the MPRTP extension header (see Section 9.2.1).

8.3. Playout Considerations at the Receiver

A media receiver, irrespective of MPRTP support or not, should be able to playback the media stream because the received RTP packets are compliant to [1], i.e., a non-MPRTP receiver will ignore the MPRTP header and still be able to playback the RTP packets. However, the variation of jitter and loss per path may affect proper playout. By calculating optimum skew across all paths, the receiver can compensate for the jitter by modifying the playout delay (adaptive playout) of the received RTP packets.

8.4. Subflow-specific RTCP Statistics and RTCP Aggregation

Aggregate RTCP provides the overall media statistics and follows the standard RTCP defined in RFC3550 [1]. However, subflow specific RTCP provides the per path media statistics because the aggregate RTCP report may not provide sufficient per path information to an MPRTP scheduler. Specifically, the scheduler should be aware of each path’s RTT and loss-rate, which an aggregate RTCP cannot provide. The sender/receiver MUST use non-compound RTCP reports defined in RFC5506 [5] to transmit the aggregate and subflow-specific RTCP reports. Also, each subflow and the aggregate RTCP report MUST follow the timing rules defined in [4].

The RTCP reporting interval is locally implemented and the scheduling of the RTCP reports may depend on the the behavior of each path. For instance, the RTCP interval may be different for a passive path than an active path to keep port bindings alive. Additionally, an endpoint may decide to share the RTCP reporting bit rate equally across all its paths or schedule based on the receiver rate on each path.

8.5. RTCP Transmission

The sender sends an RTCP SR on each active path. For each SR the receiver gets, it echoes one back to the same IP address-port pair that sent the SR. The receiver tries to choose the symmetric path and if the routing is symmetric then the per-path RTT calculations will work out correctly. However, even if the paths are not symmetric, the sender would at maximum, under-estimate the RTT of the path by a factor of half of the actual path RTT.

9. Packet Formats

In this section we define the protocol structures described in the previous sections.

9.1. RTCP Extension for Interface advertisement

This sub-section defines the RTCP header extension for in-band interface advertisement by the receiver, instead of relying on ICE or in situations when the interface appears after SDP session establishment.

The interface advertisement SHOULD immediately follow the Receiver Report. If the Receiver Report is not present, then it MUST be appended to the Sender Report.

The endpoint MUST advertise all its interfaces when a new interface appears. Furthermore, an endpoint MUST advertise all its interfaces when it receives an Interface Advertisement.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|reserved | PT=MP_IA=210  |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      SSRC of packet sender                    |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                  SSRC_1 (SSRC of first source)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         MPR_Type=0x00         |           block length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface #1 Advertisement block              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface #2 Advertisement block              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #... Advertisement block             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface #m Advertisement block              |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Figure 6: MPRTP Interface Advertisement. (appended to SR/RR)

MP_IA: 8 bits

Contains the constant 210 to identify this as an interface advertisement.

length: 16 bits

As described for the RTCP packet (see Section 6.4.1 of the RTP specification [1]), the length of this is in 32-bit words minus one, including the header and any padding.

MPR_Type: 16-bits

The MPRR_Type field corresponds to the type of MPRTP RTCP packet. Namely:

```
+---------------+------------------------------------------------+
|    MPR_Type   | Use                                            |
|     Value     |                                                |
+---------------+------------------------------------------------+
|      0x00     | Interface Advertisement                        |
|      0x01     | Connectivity Check. For this case the length is |
|               | set to 0                                       |
|      TBD      | Keep Alive Packet.                             |
+---------------+------------------------------------------------+
```

Figure 7: RTP header extension values for MPRTP (MPR_Type)

block length: 16-bits

The 16-bit length field is the length of the encapsulated advertisement blocks in 32-bit word length not including the MPR_Type and length fields. The value zero indicates there is no data following.

Interface Advertisement block: variable size

Defined later in 9.1.1.

9.1.1. Interface Advertisement block

This block describes a method to represent IPv4, IPv6 and generic DNS-type addresses in a block format. It is based on the sub-reporting block in RFC 5760 [6].

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Type={0,1,2}  |     Length    |          Subflow ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           RTCP Port           |           RTCP Port           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Address                           |
+                                                               +
:                                                               :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: Interface Advertisement block during path discovery

Type: 8 bits

The Type corresponds to the type of address. Namely:

0: IPv4 address

1: IPv6 address

2: DNS name

Length: 8 bits

The length of the Interface Advertisement block in bytes.

For an IPv4 address, this should be 9 (i.e., 5 octets for the header and 4 octets for IPv4 address).

For an IPv6 address, this should be 21.

For a DNS name, the length field indicates the number of octets making up the string plus the 5 byte header.

RTP Port: 2 octets

The port number to which the sender sends RTP data. A port number of 0 is invalid and MUST NOT be used.

RTCP Port: 2 octets

The port number to which receivers send feedback reports. A port number of 0 is invalid and MUST NOT be used.

Address: 4 octets (IPv4), 16 octets (IPv6), or n octets (DNS name)

The address to which receivers send feedback reports. For IPv4 and IPv6, fixed-length address fields are used. A DNS name is an arbitrary-length string. The string MAY contain Internationalizing Domain Names in Applications (IDNA) domain names and MUST be UTF-8 encoded [7].

9.2. MPRTP RTP Header Extension

The MPRTP header extension is used to 1) distribute a single RTP stream over multiple subflows, 2) perform connectivity checks on the advertised interfaces, and 3) keep-alive passive interfaces (paths).

The header conforms to the 2-byte RTP header extension defined in [8]. The header extension contains a 16-bit length field that counts the number of 32-bit words in the extension, excluding the four-octet extension header (therefore zero is a valid length, see Section 5.3.1 of [1] for details).

To signal the use of the above RTP header extensions in SDP, the following URI MUST be used: urn:ietf:params:rtp-hdrext:mprtp.

9.2.1. MPRTP RTP Extension for a Subflow

The RTP header for each subflow is defined below:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|1|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|      0x10     |     0x00      |        len=N-1 words          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    H-Ext ID   |    length     |          Subflow ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Subflow-specific Seq Number  |    Pad (0)    |   Pad (0)     |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                          RTP payload                          |
|                              ....                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9: MPRTP header for subflow

H-Ext ID and length: 8-bits each

The field corresponds to the type of MPRTP packet. Namely:

H-Ext ID Value	Use
0x00	Subflow RTP Header. For this case the Length is set to 6
0x01	Connectivity Check. For this case the length is set to 0
TBD	Keep Alive Packet.

Figure 10: RTP header extension values for MPRTP (H-Ext ID)

length

The 8-bit length field is the length of extension data in bytes not including the H-Ext ID and length fields. The value zero indicates there is no data following.

Subflow ID: Identifier of the subflow. Every RTP packet belonging to the same subflow carries the same unique subflow identifier.

Flow-Specific Sequence Number (FSSN): Sequence of the packet in the subflow. Each subflow has its own strictly monotonically increasing sequence number space.

9.2.2. MPRTP RTP Extension for Connectivity Checks

[Open Issue: What sequence number to use for the RTP session? Alternative 1: An MPRTP receiver MUST NOT send the packet with H-Ext ID=0x01 to the decoder and ignore these packets from RTCP calculation. Alternative 2: Instead of sending an RTP packet the sender transmits a modified STUN packet.]

9.2.3. MPRTP RTP Extension for Keep-alive Packets

[Editor: Waiting for the progress on RTCP guidelines for the RTP keep alive packet [16].

9.3. MPRTP Extension for Subflow Reporting (MPRTCP)

The MPRTP RTCP header extension is used to 1) provide RTCP feedback per subflow to determine the characteristics of each path, 2) perform connectivity check on the other endpoint’s interfaces, and 3) to keep alive a passive connection.

9.3.1. MPRTCP Generic Extension

When sending a report for a specific subflow the sender or receiver MUST add only the reports associated with that 4-tuple. Each subflow is reported independently using the following MPRTCP Feedback header.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|reserved |   PT=SFR=211  |           length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ H
|                         SSRC of sender                        | D
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ R
|         Subflow ID #1         |           reserved            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                    Subflow-specific reports                   |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|reserved |   PT=SFR=211  |           length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ H
|                         SSRC of sender                        | D
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ R
|         Subflow ID #2         |           reserved            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                    Subflow-specific reports                   |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 11: MPRTCP Generic Feedback Header

Subflow ID: 16 bits

Subflow identifier is the value associated with the subflow the endpoint is reporting about. If it is a sender it MUST use the Subflow ID associated with the 4-tuple. If it is a receiver it MUST use the Subflow ID received in the Subflow-specific Sender Report.

length: 16 bits

The length of this RTCP packet in 32-bit words minus one, including the header and any padding. It MUST contain at least one subflow report, for e.g., Sender Subflow Report, Receiver Subflow Report, or Subflow Extension Reports, etc.

Subflow-specific reports: variable

Subflow-specific report contains all the reports associated with the Subflow ID. For a sender, it MUST include the Subflow-specific Sender Report (SSR). For a receiver, it MUST include Subflow-specific Receiver Report (SRR). Additionally, if the receiver supports subflow-specific extension reports then it MUST append them to the SRR.

9.3.2. MPRTCP for Subflow-specific SR, RR and XR

[Editor: inside the context of subflow specific reports can we reuse the payload type code for Sender Report (PT=200), Receiver Report (PT=201), Extension Report (PT=207). Transport and Payload specific RTCP messages are session specific and SHOULD be used as before.]

Example:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |V=2|P|reserved |   PT=SFR=211  |             length=9          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ H
   |                         SSRC of sender                        | D
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ R
   |         Subflow ID #1         |             reserved          |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |V=2|P|    RC   |   PT=SR=200   |             length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         SSRC of sender                        |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |              NTP timestamp, most significant word             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             NTP timestamp, least significant word             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         RTP timestamp                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    subflow’s packet count                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     subflow’s octet count                     |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |V=2|P|reserved |   PT=SFR=211  |             length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ H
   |                         SSRC of sender                        | D
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ R
   |         Subflow ID #2         |             reserved          |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |V=2|P|    RC   |   PT=RR=201   |             length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     SSRC of packet sender                     |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   | fraction lost |       cumulative number of packets lost       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           extended highest sequence number received           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      interarrival jitter                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         last SR (LSR)                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   delay since last SR (DLSR)                  |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |             Subflow specific extension reports                |
   |                             ....                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: Example of reusing RTCP SR and RR inside an MPRTCP header

(Bi-directional use-case).

10. SDP Considerations

The packet formats specified in this document define extensions for RTP and RTCP. The use of MPRTP is left to the discretion of the sender and receiver.

A participant of a media session MAY use SDP to signal that it supports MPRTP. Not providing this information may/will make the sender or receiver ignore the header extensions. However, MPRTP MAY be used by either sender or receiver without prior signaling.

```
mprtp-attrib = "a=" "mprtp" [":"
        mprtp-optional-parameter]
        CRLF   ; flag to enable MPRTP
```

The literal 'mprtp' MUST be used to indicate support for MPRTP. Generally, senders and receivers SHOULD indicate this capability if they support MPRTP and would like to use it in the specific media session being signaled. However, it is possible for an MPRTP sender to stream data using multiple paths to a non-MPRTP client.

Currently, there are no extensions defined for the literal 'mprtp' but we provide the opportunity to extend it using the mprtp-optional-parameter.

10.1. Increased Throughput

The MPRTP layer MAY choose to augment paths to increase throughput. If the desired media rate exceeds the current media rate, the endpoints MUST renegotiate the application specific ("b=AS:") [17] bandwidth.

10.2. Increased Reliability

TBD

10.3. MPRTP using preloaded interfaces from ICE

TBD

11. IANA Considerations

This document defines a new SDP attribute, "mprtp", within the existing IANA registry of SDP Parameters.

TBD.

12. Security Considerations

All drafts are required to have a security considerations section. See RFC 3552 [18] for a guide.

13. Acknowledgements

Varun Singh, Saba Ahsan, and Teemu Karkkainen are supported by Trilogy (http://www.trilogy-project.org), a research project (ICT-216372) partially funded by the European Community under its Seventh Framework Program. The views expressed here are those of the author(s) only. The European Commission is not liable for any use that may be made of the information in this document.

14. References

14.1. Normative References

[1] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

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

[3] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[4] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

[5] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.

[6] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback", RFC 5760, February 2010.

[7] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[8] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

14.2. Informative References

[9] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[10] Ford, A., Raiciu, C., Handley, M., Barre, S., and J. Iyengar, "Architectural Guidelines for Multipath TCP Development", draft-ietf-mptcp-architecture-05 (work in progress), January 2011.

[11] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, June 2009.

[12] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.

[13] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[14] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, "Real Time Streaming Protocol 2.0 (RTSP)", draft-ietf-mmusic-rfc2326bis-27 (work in progress), March 2011.

[15] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[16] Marjou, X. and A. Sollaud, "Application Mechanism for keeping alive the Network Address Translator (NAT) mappings associated to RTP/RTCP flows.", draft-ietf-avt-app-rtp-keepalive-10 (work in progress), March 2011.

[17] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[18] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

Authors' Addresses

Varun Singh
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: varun@comnet.tkk.fi
URI: http://www.netlab.tkk.fi/˜varun/

Teemu Karkkainen
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: teemuk@comnet.tkk.fi

Joerg Ott
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: jo@comnet.tkk.fi

Saba Ahsan
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: sahsan@cc.hut.fi

Lars Eggert
Nokia Research Center
P.O. Box 407
Nokia Group 00045
Finland

Phone: +358 50 48 24461
Email: lars.eggert@nokia.com
URI: http://research.nokia.com/people/lars_eggert

AVT T. VanCaenegem
Internet-Draft Alcatel-Lucent
Intended status: Standards Track March 07, 2011
Expires: September 8, 2011

RTCP FB NACK storm suppression and its impact on retransmission in RTP SSM sessions with unicast FB
draft-vancaenegem-avtcore-fb-supp-and-retransm-00

Abstract

This document discusses how RTCP Feedback storm suppression negatively affects retransmission efficacy for Source Specific Multicast sessions with unicast feedback architectures and proposes some recommendations by means of additional signaling (e.g. RSI message and new attribute parameters) and a small AVPF FB suppression rule modification resulting in a overall better system where the FB suppression can be maintained but with a optimised retransmission efficacy.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on September 8, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

RTCP Feedback (FB) storm suppression is efficiently realised by means of the AVPF algorithm implemented at RTP receivers participating to multi-party RTP sessions, defined in [RFC4585]. In RTP multi-party sessions, a single event may impact many or even all RTP receivers. RTP receivers that react on a packet loss event by sending RTCP FB NACK messages, must follow the [RFC4585] AVPF timing rules which include a suppression rule: a RTP receiver receiving the same FB message as the one it intends to send, must discard its own FB message. This results in FB storm suppression or mitigation. The AVPF FB storm suppression mechanism is introduced to protect the network, server(s) and indirectly all the receivers and works well in most RTP topologies, including SSM with unicast feedback. However, such RTCP feedback storm suppression does result in decreased visibility on the status of RTP receivers, and hence impacts monitoring service and also services triggered by the reception of such RTCP FB messages, such as the packet loss recovery service by means of RTP retransmission . RTP retransmissions are requested from RTP receivers by RTCP FB NACK messages, reporting RTP packet loss. The internet draft "draft-wu-avt-retransmission-supression-rtp" discusses FB storm suppression, and proposes a new RTCP message that is called "third party loss" message that can be taken advantage to counter FB storms for various considered architectures of various RTP multi-party sessions. However, the usability of such message in the context of the AVPF suppression algorithm is not clearly addressed and the possible interference with a packet loss recovery service by means of RTP retransmission is omitted in draft "draft-wu-avt-retransmission-supression-rtp". For instance, in the transport translator scenario that is addressed in the draft, all RTCP messages must normally be forwarded, and hence every RTCP NACK/ FIR FB message from one receiver will be sent to all other receivers, where the [RFC4585] FB suppression rule will kick-in. Hence the "third party loss" message does not bring substantial value.

The present draft discusses also FB storm suppression, but focuses on RTCP SSM architectures with unicast feedback target(s). It describes how the goals of FB NACK storm suppression and the goal of retransmission in terms of service enhancement are often in conflict with each other. To that extent it discusses several packet loss event use cases for RTCP SSM with unicast FB -for both modes defined in [RFC5760] including the SSM architecture with multiple FTs and makes proposals to reconcile suppression and (retransmission) service fulfillment for SSM with unicast feedback, taking into account the minimisation of network bandwidth and server resources. In certain scenarios/architectures, the "third party loss" message from "draft-wu-avt-retransmission-supression-rtp" can be leveraged.

2. Requirements Notation

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 [RFC2119].

3. Considered Architecture

In this draft we consider the Source Specific Multicast (SSM) with unicast feedback architecture as defined in [RFC5760] defining one or several media senders, a Distribution Source (DS)-sourcing the SSM-, one or several Feedback Targets (FT) that may be co-joint with the DS and the SSM RTP receivers that provide unicast feedback to a FT. On top, similar as defined in [I-D.ietf-avt-rapid-acquisition-for-rtp] , also a Retransmission Source is considered that is co-joint with the Feedback Target, and together constitute the Retransmission Server (RS). It is assumed that the receivers support sending RTCP NACK FB messages . Two models for SSM with unicast FB have been defined in [RFC5760]:

- o In distribution source feedback summary model, the unicast RTCP Receiver Report messages from the SSM RTP receivers are default aggregated by the DS and their information is transmitted as Receiver Summary Information (RSI) messages in the SSM session. The RTCP FB packets are default terminated by the DS. However, the DS may also aggregate or forward RTCP FB packets and transmit them on the SSM, when this is explicitly signaled. Note that from the RTP perspective, the DS is an RTP receiver generating its own RTCP RR as well as other RTCP packets and sending them to the receiver group and media senders.

- o In simple feedback model the DS must reflect all RTCP messages (hence including RTCP FB) received in unicast via the FT from the SSM RTP receivers.

In the remainder of this draft, both models will be considered. It must be noted that for large group of receivers in a SSM with unicast feedback session, the feedback summary model is the most useful one, as the simple feedback model would result in significant reflected RTCP messaging overhead in the network and for all the SSM receivers, from bandwidth resources and processing overhead point of view. We also make in this draft a distinction between two topologies for SSM with unicast feedback with retransmission server capability :

- o a topology where there is one DS and a single FT, and where the FT is combined with a Retransmission Source function. The FT/RS could be joint or disjoint from the DS, but this is not really relevant in the discussion. Because a retransmission packet is in general a response on a NACK directed to a FT, combining the FT and the RS in a single entity is a logical choice. In the remainder of this draft, the co-location of FT and RS is assumed, and they represent together the retransmission server, in agreement with [I-D.ietf-avt-rapid-acquisition-for-rtp].

o a topology where we have multiple FTs that are disjoint from the DS, and where each FT is combined with a Retransmission Source function. Also here is assumed that the FT is co-located with the Retransmission Source, being together the Retransmission Server.

The interference of the [RFC4585] FB suppression mechanism with the client's ability for receiving retransmissions from the RS is discussed first for SSM with unicast feedback and single FT/RS, followed by a discussion for an SSM architecture with multiple FT/RS.

4. Feedback suppression in combination with retransmission for SSM with unicast feedback with single FT/RS

In the SSM architecture with single FT/RS that is either co-joint or separate from the DS, a FB storm can always be prevented or mitigated because SSM RTP receivers implementing AVPF, must adhere to the suppression rules defined in [RFC4585]. It is explored in this and the following sections how this impacts a RTP retransmission packet loss repair service for a given packet loss event at a SSM RTP receiver.

4.1. Packet Loss Upstream of the Distribution Source

As an first example of a packet loss event triggering FB suppression, a packet drop event somewhere along the data path between a media sender and the DS is considered. All SSM RTP receivers will notice this packet loss, including the DS itself (in the RTP stream between the media sender and the DS).

- o In the simple feedback model, all RTCP messages are relayed back to all receivers and the media source(s). Hence, the first RTCP NACK(s) sent by a SSM RTP receiver or a subset of the SSM RTP receivers, will be relayed by the DS to all SSM receivers, which will make the other RTP SSM receivers refrain from sending a NACK, as determined by the RTP receiver’s AVPF RTCP scheduling algorithm. This will limit the amount of RTCP FB traffic from the SSM receivers both to the DS and to the media source(s), and avoid a FB storm.

- o In the feedback summary model, the DS is an RTP receiver generating its own receiver reports and sending these to the receiver group and to the media senders. For the given use case example, the DS can generate its own RTCP FB message and send it to the SSM group. All SSM receivers supporting and implementing AVPF will adhere to the FB suppression rule defined in [RFC4585], and hence a FB storm is avoided. The DS can choose to send a FB NACK multiple times for redundancy reasons, as long as it complies to the AVPF RTCP scheduling algorithm.

Consider now that the FT is also a retransmission server which can respond with retransmissions when receiving a RTCP FB NACK from a RTP receiver- provided this server has access to the original RTP packet. Note that for the considered packet loss use case, the retransmission server will also detect the packet loss. In both SSM models when considering large groups of receivers, at least one but not more than a few receivers will send a NACK FB. However still all receivers will be impacted by the packet loss and desire a retransmission. A fundamental aspect of combining retransmission service with FB suppression mechanism, is that retransmissions may be sent to receivers that are unsollicited. An unsollicited retransmission can be defined as a retransmission received by a receiver which was not requested by this same receiver via an RTCP FB NACK message or any other explicit signaling from that receiver. Sollicited retransmission is a retransmission provided to a receiver which was explicitly requested by that same receiver. For the given packet loss event, when the RS is capable of recovering the lost packet (the way this is achieved is not relevant nor discussed here), it can provide the retransmission to all RTP receivers. This retransmission can be performed either in a each separate unicast RTP retransmission sessions to each receiver or - in a single SSM RTP session that is session muxed with the original RTP SSM.

Providing the retransmission over SSM has the advantage that

- o the retransmission packet must be transmitted by the DS/RS only once, saving both network and server resources

- o because the retransmission is not explicitly sollicited by means of a NACK, it may happen that the unsollicited retransmission packet when transmitted in unicast is blocked by any intermediate gateway on the path between the retransmission server and the RTP receiver.

When a packet loss repair service is announced as a retransmission server-sourced SSM retransmission session, a RTP receiver that joins this RTP SSM retransmission session via IGMP/MLP, implicitly indicates it is willing to accept retransmissions over this SSM, that are unsollicited. When there is no SSM retransmission session in place and signaled to the receivers, a RS can of course still send unsollicited retransmissions in those unicast retransmission sessions that are established as per [I-D.ietf-avt-ports-for-ucast-mcast-rtp]. It is implementation-specific whether RTP receivers choose to ignore received unsollicited retransmission packets (in the same way as RTP receivers may ignore retransmission packets for which the receiver did send a NACK FB message, i.e. sollicited retransmission)

It should be noted that when an SSM RTP receiver is involved in both a unicast retransmission session and a SSM retransmission session sourced by the same retransmission server, a retransmission of a packet transmitted in the original SSM may be sent in the unicast retransmission session, in the multicast retransmission session or both. The retransmission server SHOULD send unsollicited retransmissions over the retransmission SSM session when such session is available. A retransmission server that receives a RTCP FB NACK and decides to provide a retransmission, should (also) send that retransmission in the unicast retransmission session to the receiver that sent the RTCP FB NACK (when such a unicast retransmission session is available and established as described in [I-D.ietf-avt-ports-for-ucast-mcast-rtp].

In conclusion, note that in this packet loss event use case:

- all SSM RTP receivers were impacted by the packet loss and detected this packet loss
- all receivers behave compliant to [RFC4585] in terms of RTCP transmission scheduling and suppression rules
- no RTCP FB storms occur
- all SSM RTP receivers can receive the retransmission either in a dedicated retransmission SSM or in separate unicast retransmission sessions established by the RTP receivers.

Whether the retransmission server does provide a retransmission and to which RTP receiver (when using unicast retransmission) is governed by the retransmission server policy.

4.2. Packet Loss Downstream of the Distribution Source

As a second example packet loss event triggering feedback suppression, consider a packet drop event in the SSM tree downstream of the DS/FT, which may impact just one SSM RTP receiver but can possibly also impact a large set of SSM RTP receivers (all those that are downstream of the SSM tree branch where the packet loss event occured). The DS/RS in general does not know a particular RTP packet got lost untill it starts receiving RTCP FB NACK(s) from one or more SSM RTP receivers. Note that for the considered packet loss event use case, the RS will have in its cache the missing packet as the original packet got dropped downstream of the DS/RS. The feedback suppression will, depending on where the packet was lost, possibly interfere with the packet loss repair service based on RTP retransmission , as explained below for the two SSM feedback models.

4.2.1. Simple feedback model

Consider the simple feedback model where a retransmission server is in place that is co-located with the DS. Assume that multiple RTP receivers observe the same packet loss in the RTP SSM, which is most likely caused by a single packet loss drop in a branch of the SSM tree connecting to the impacted receivers. Even though the retransmission server may be capable of providing a retransmission to all impacted SSM RTP receivers, even when each receiver individually transmits a RTCP FB NACK, some receivers may not have a chance to receive a retransmission. This is due to the fact that the DS is supposed to reflect all RTCP FB messages. Hence because of the RTCP FB transmission suppression algorithm, the RS will not know which (other) SSM receivers experienced the same packet loss.

There are two ways to make sure that all impacted receivers do get a retransmission:

o FB suppression is enabled by having the DS reflecting any RTCP FB message received, but the RS does send a unsollicited retransmission to all SSM receivers, each time a RTCP FB NACK is received. This solution is not desirable as it provides retransmissions to SSM receivers which are -in the large majority of possible cases- unneeded and results in a waste of network and also a waste of server resources when retransmissions are provided over unicast.

o FB suppression is disabled because the DS does not forward/reflect RTCP FB packets down the SSM. All SSM RTP receivers impacted by the loss (ranging from one to all of the SSM RTP receivers) will send a RTCP FB NACK. Only when the storm of RTCP FB packets has no detrimental impact, the RS can respond to each NACK with a retransmission packet in each unicast retransmission session -or, alternatively, the retransmission is provided over a dedicated retransmission SSM.

In summary: the DS/RS is capable of preventing a FB storm by reflecting the received RTCP FB messages down the SSM with the disadvantage of having no visibility on which receiver has detected which missing packets in the SSM. Alternatively, the DS takes the risk of being confronted with a FB storm by not forwarding the RTCP FB messages, where in general each SSM receiver that detected the packet loss event, can be paired with a unicast retransmission. The second option is in conflict with the forwarding requirement defined in [RFC5760]. It is also in disagreement with the first use case (packet loss upstream of the DS) where a simple reflection behaviour does result in efficient FB suppression, without withholding the impacted receivers from receiving a (unsollicited) retransmission.

The general recommended solution addressing packet loss event use cases 1 and 2, is therefor to allow "selective" reflection (or "selective" termination) in the simple feedback model for RTCP FB messages. It allows feedback suppression but still giving visisbility to the DS on which are the impacted receivers, and providing reasonable guarantees on a efficient retransmission service to all receivers.

With "selective" RTCP FB reflection, the DS will in general not reflect RTCP FB messages received from SSM receivers except in the following two cases:

o the DS/RS itself is subject to packet loss and will reflect any RTCP FB NACK received from the downstream SSM RTP receivers reporting this same packet loss.

o the DS (selectively) reflects received RTCP FB NACK, when the RS itself was not impacted directly by the packet loss but a certain threshold for incoming RTCP FB NACK packets has been reached, all pertaining to the same original packet in the SSM. This threshold is based on the total amount of receivers reporting to the FT/RS, and can be adjusted dynamically, but this is a metric internal to the FT/DS.

Note that there is maximum efficiency in the retransmission operation that may occur after reflecting the RTCP FB NACK in these two exception cases, if the retransmission takes place over a (retransmission) SSM RTP session.

The proposal is to define an additional parameter for the "rtcp-unicast" SDP attribute indicating SSM sessions with unicast feedback operated in simple feedback mode, named "selective reflection". Its meaning is that RTCP FB messages may not be reflected by the FT/DS, but instead terminated. All other RTCP reports are reflected, as imposed by [RFC5760].

4.2.2. Feedback summary model

For the considered use case of packet loss downstream of the DS/RS, similar as discussed for the simple feedback model discussion, feedback suppression is enabled by having the DS selectively forwarding the received RTCP FB messages: e.g. when the number of received RTCP FB NACKs pertaining to the same RTP packet loss crosses a certain threshold, the DS fowards such a RTCP FB NACK.

"Selective Forwarding" is therefore proposed as a new parameter for the processing attribute in the rsi-rule in the SDP for the summary feedback model, that is allowed ONLY for RTCP FB packets:

Alternatively the DS/RS always terminates RTCP FB messages, but prevents FB storms, in the following two ways:

o for packet loss events taking place upstream of the DS, the DS simply sends itself a RTCP FB NACK

o for packet loss events taking place downstream of the DS, the reception of RTCP FB NACKs may trigger the transmission of a new

RTCP FB packet by the DS, named "3rd party NACK" which has the same semantics as a RTCP FB NACK, as defined in draft "draft-wu-avt-retransmission-supression-rtp"

A SSM RTP receiver , receiving this message in the SSM, shall treat it the same way as a RTCP FB NACK received from another SSM RTP receiver and hence SHALL NOT send a RTCP FB message. The DS needs to carefully evaluate when to send or not send such a "3rd party NACK", as discussed previously in this section.

5. Feedback suppression and retransmission for SSM with unicast feedback with multiple and disjoint FTs

A specific case of SSM with unicast FB, is where there are multiple FTs disjoint from the DS. Similar as before, in the considered architectures, each FT is combined with a retransmission source, constituting a retransmission server [[I-D.ietf-avt-rapid-acquisition-for-rtp]]. Note that the RS (=FT+ BRSource) are generally not positioned in the direct SSM path between the DS and the SSM RTP receivers. This architecture provides a scalable solution for SSM with a large population of receivers, because it is able to distribute RTCP feedback processing loads across different entities in different parts of the network. It is an architecture that is well suited for IPTV networks of large service providers, where the DS is the head-end sourcing the SSM that carry broadcast streams over IP.

[RFC5760] indicates that for the simple FB model where the FT(s) are disjoint from the DS, the FT must forward all RTCP packets to the DS.

[RFC5760] indicates that for the summary FB model where the FT(s) are disjoint from the DS, the following:

- The Feedback Target(s) MAY simply forward all RTCP packets incoming from the RTP receivers to the Distribution Source
- The Feedback Target(s) MAY also perform aggregation of incoming RTCP packets and send only aggregated information to the Distribution Source.
- If the Feedback Target performs summarization functions, it MUST also act as a receiver and choose a unique SSRC for its own reporting towards the Distribution Source.

The discussion on how FB suppression and retransmissions can be efficiently combined for the SSM with single FT topology -as discussed above- remains applicable and valid for the SSM with multiple (disjoint) FTs topology, but there is an additional aspect that should be addressed, and a third example packet loss event use case visualises this.

The considered topology is a DS with two disjoint FT/RS entities, FT/RS 1 and FT/RS 2 , where each FT receives RTCP messages from a separate group of SSM RTP receivers. The assumption is that a RTP packet (with Sequence Number N) in the original SSM got dropped in the network upstream of the FT 1 (and hence impacting FT 1, as well as all the SSM receivers that report to FT 1). Because FT 2 does not get the original SSM packets from the DS via the router where the packet loss took place, the FT 2 does receive the packet with SN N and so do the SSM receivers reporting to FT 2.

The FT 1 and its reporting SSM receivers experience a situation that is discussed in the first use case for the SSM with single FT topology. Because the packet loss event impacts all SSM receivers reporting to FT 1, it is paramount that those receivers in general suppress sending a RTCP FB NACK. Hence having the FT 1 forwarding the first received RTCP FB NACK(s) from a SSM RTP receiver to the DS -which then reflects/forwards the FB NACK over the original SSM, is the correct thing to do from that point of view. However, the reflection /forwarding of the FB NACK by the DS means that also the SSM RTP receivers reporting to the FT 2 will suppress sending an RTCP FB NACK for packet N in the SSM, even if they detect the same packet loss - but which is not caused by the packet loss event impacting FT/RS 2 and all its reporting SSM RTP receivers. This means there is a discrepancy between the network reach of the suppression (covering all SSM receivers) and the actual network subdomain that was commonly impacted by the packet loss. The RS 2 will in general not know whether there are any SSM receivers -reporting to FT 2- that missed RTP packet with RTP SN N because of a different packet loss event .

Note also that the unsollicited retransmission by RS 1 -following the packet loss with SN N detection -can remain confined to the subdomain impacted by the loss, when the FT is co-located with the RS (using either unicast retransmission sessions or a SSM retransmission session sourced by the RS).

SSM with multiple disjoint unicast FTs hence may result in efficient feedback storm supression across all SSM RTP receivers, but this also prevents any SSM RTP receiver from sending a RTCP FB NACK for detected packet loss, even when no FB storm was imminent for the subdomain covered by a particular FT. A solution for maximising the retransmission service fulfillment may be for the DS to also act as RS and always send retransmissions requested by a particular FT, over a separate retransmission SSM to all SSM RTP receivers. However, this unnecessarily loads the network and requires all the SSM RTP receivers to receive both an original/primary SSM and a retransmission SSM.

A more optimised solution is to keep both the FB suppression and retransmission within the same local "subdomain". This can be enabled by adding a rule to the AVPF FB suppression algorithm, that makes the suppression mechanism "selective" at the SSM RTP receivers side. The proposed overall solution to enable such selective receiver FB storm suppression algorithm, is accomplished in three steps:

o The SSM RTP receivers first learn the SSRC identifier of the FT where the FT either acts as a translator in the original SSM session (simple feedback model) or acts as a SSM RTP receiver. There are several ways through which this can happen:

  * "in-band" : applies only for the feedback summary model, where by means of a new RSI message the DS provides a listing of all deployed FTs with the corresponding SSRC for each of these FTs.

  * "out-of-band" for either the feedback summary or simple feedback model of SSM operation, by advertising the FT’s SSRC as a media attribute for the FT in the SSM RTP session description [RFC5576] .

  The "out-of-band" signaling mechanism requires the application signaling to know/learn the SSRCs deployed by the FTs prior to signaling this information to the SSM RTP receivers (those receivers that are not acting as FT).

o In the feedback summary model, the FT does not forward RTCP FB NACK messages as received from the SSM RTP receivers to the DS. Instead the FT sends a RTCP FB NACK message using its own SSRC when the FT/RS itself directly detected the common packet loss event. Alternatively, the FT sends the RTCP message "3rd party NACK" ["draft-wu-avt-retransmission-supression-rtp"] using its own SSRC when it senses a local FB storm is imminent but when the RS itself was not subject to the packet loss. In the simple feedback model, the FT can act as translator in the SSM session and send the new RTCP FB message "3rd party NACK" using its own SSRC. Alternatively and similar as described for the feedback summary model, when the DS advertises itself as FT towards the RSs that host the FTs, the RS sends as SSM RTP receiver to the DS a RTCP FB NACK or RTCP FB "3rd party NACK", depending on whether it detected the reported packet loss itself or not.

o The DS forwards the RTCP FB NACK messages or RTCP FB "3rd party loss" messages received from any of the FTs in the SSM session, down on the original SSM session. The feedback suppression can then remain localised, by having an SSM RTP receiver only activating feedback suppression when the "SSRC of packet sender" field value in the received RTCP FB message(s) matches with the SSRC that is used by the local FT to which it reports.

Note that when the DS sends a third party loss report or NACK RTCP FB message using its own SSM SSRC, all the SSM RTP receivers (including the FTs) will abstain from sending a RTCP FB message, enabling a FB storm suppression across the whole SSM network domain. This occurs e.g. when a packet loss event took place between a media sender and the DS.

6. Security Considerations

No dedicated security measures must be considered other than the ones defined in [RFC4585] and [RFC5760].

7. IANA Considerations

The following contact information shall be used for all registrations in this document:

"tom.van_caenegem@alcatel-lucent.com"

7.1. Registration of SDP Attributes

TBC.

8. Acknowledgments

9. References

9.1. Normative References

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

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

[RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, June 2009.

[RFC5760] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback", RFC 5760, February 2010.

9.2. Informative References

[I-D.ietf-avt-ports-for-ucast-mcast-rtp]
Begen, A., Wing, D., and T. VanCaenegem, "Port Mapping Between Unicast and Multicast RTP Sessions", draft-ietf-avt-ports-for-ucast-mcast-rtp-11 (work in progress), January 2011.

[I-D.ietf-avt-rapid-acquisition-for-rtp]
Steeg, B., Begen, A., Caenegem, T., and Z. Vax, "Unicast-Based Rapid Acquisition of Multicast RTP Sessions", draft-ietf-avt-rapid-acquisition-for-rtp-17 (work in progress), November 2010.

[RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, July 2006.

Author's Address

Tom VanCaenegem
Alcatel-Lucent
Copernicuslaan 50
Antwerpen,   2018
Belgium

Email: Tom.Van_Caenegem@alcatel-lucent.com