IETF RMCAT Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2017

Z. Sarker
Ericsson AB
C. Perkins
University of Glasgow
V. Singh
callstats.io
M. Ramalho
Cisco Systems

July 07, 2016

RTP Control Protocol (RTCP) Feedback for Congestion Control
draft-dt-rmcat-feedback-message-00

Abstract

This document describes a feedback message intended to enable congestion control for interactive real-time traffic. The RTP Media Congestion Avoidance Techniques (RMCAT) Working Group formed a design team to analyze feedback requirements from various congestion control algorithms and to design a generic feedback message to help ensure interoperability across those algorithms. The feedback message is designed for a sender-based congestion control, which means the receiver of the media will send necessary feedback to the sender of the media to perform the congestion control at the sender.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

For interactive real-time traffic the typical protocol choice is Realtime Transport Protocol (RTP) over User Datagram Protocol (UDP). RTP does not provide any guarantee of Quality of Service (QoS), reliable or timely delivery and expects the underlying transport protocol to do so. UDP alone certainly does not meet that expectation. However, RTP Control Protocol (RTCP) provides a mechanism to periodically send transport and media metrics to the media sender which can be utilized and extended for the purposes of RMCAT congestion control. For a congestion control algorithm which operates at the media sender, RTCP messages can be transmitted from the media receiver back to the media sender to enable congestion control. In the absence of standardized messages for this purpose, the congestion control algorithm designers have designed proprietary RTCP messages that convey only those parameters required for their respective designs. As a direct result, the different congestion control (a.k.a. rate adaptation) designs are not interoperable. To enable algorithm evolution as well as interoperability across designs (e.g., different rate adaptation algorithms), it is highly desirable to have generic congestion control feedback format.

To help achieve interoperability for unicast RTP congestion control, this memo proposes a common RTCP feedback format that can be used by NADA [I-D.ietf-rmcat-nada], SCReAM [I-D.ietf-rmcat-scream-cc], Google Congestion Control [I-D.ietf-rmcat-gcc] and Shared Bottleneck Detection [I-D.ietf-rmcat-sbd], and hopefully future RTP congestion control algorithms as well.

[Editor’s Note: consider removing this part of the section in the later versions] In preparing this memo, we have considered the following:

- o What are the feedback requirements for the proposed RTP congestion control candidate solution?
- o Can we design a feedback message that is future proof, and general enough to meet the needs of algorithms that have yet to be defined?
- o Can we use existing RTCP Extended Report (XR) blocks and/or RTCP Feedback Messages? If not, what is the rationale behind new XR blocks and/or RTCP feedback messages?
- o What will be the wire format of the generic feedback message?

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 [RFC2119].

In addition the terminology defined in [RFC3550], [RFC3551], [RFC3611], [RFC4585], and [RFC5506] applies.

3. Feedback Message

The design team analyzed the feedback requirements from the different proposed candidate in RMCAT WG. The analysis showed some commonalities between the proposed solution candidate and some can be derived from other information. The design team has agreed to have following packet information block in the feedback message to satisfy different requirement analyzed.

- o Packet Identifier : RTP sequence number. The RTP packet header includes an incremental packet sequence number that the sender needs to correlate packets sent at the sender with packets received at the receiver.

- Packet Arrival Time : Arrival time stamp at the receiver of the media. The sender requires the arrival time stamp of the respective packet to determine delay and jitter the packet had experienced during transmission. In a sender based congestion control solution the sender requires to keep track of the sent packets - usually packet sequence number, packet size and packet send time. With the packet arrival time the sender can detect the delay and jitter information. Along with packet loss and delay information the sender can estimate the available bandwidth and thus adapt to the situation.

- Packet Explicit Congestion Notification (ECN) Marking : If ECN [RFC3168] is used, it is necessary to report on the 2-bit ECN mark in received packets, indicating for each packet whether it is marked not-ECT, ECT(0), ECT(1), or ECN-CE. If the path on which the media traffic traversing is ECN capable then the sender can use the Congestion Experienced (ECN-CE) marking information for congestion control. It is important that the receiver sends the ECN-CE marking information of the packet back to the sender to take the advantages of ECN marking. Note that how the receiver gets the ECN marking information at application layer is out of the scope of this design team. Additional information for ECN use with RTP can be found at [RFC6679].

The feedback messages can have one or more of the above information blocks. For RTCP based feedback message the packet information block will be grouped by Synchronization Source (SSRC) identifier.

As a practical matter, we note that host Operating System (OS) process interruptions can occur at inopportune times. Thus, the recording of the sent times at the sender and arrival times at the receiver should be made with deliberate care. This is because the time duration of host OS interruptions can be significant relative to the precision desired in the one-way delay estimates. Specifically, the send time should be recorded at the latest opportunity prior to outputting the media packet at the sender (e.g., socket or RTP API) and the arrival time at the receiver (e.g., socket or RTP API) should be recorded at the earliest opportunity available to the receiver.

3.1. RTCP Packet format

The feedback is over RTCP [RFC3550] and it is described as a stand alone RTCP packet for now, suitable for use in regular RTCP reports. [FIXME: this is simply a placeholder for now. We can design different wire format of this packet with different efficiency in mind. This doc will contain a very simple format. The optimized versions will be discussed in the group and finally the selected one will replace this simple format in future. This section will describe a new RTP/AVPF transport feedback message and a new RTCP XR block report]

```
     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=RC2F    |   SSRC count  |      Block Length = TBD       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        SSRC of Source                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  Report Timestamp (32bits)                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   SSRC of 1st media source                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          begin_seq            |             end_seq           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |L|ECN|  Arrival time offset    | ...                           .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    .                                                               .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   SSRC of nth media source                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          begin_seq            |             end_seq           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |L|ECN|  Arrival time offset    | ...                           |
    .                                                               .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The XR Discard RLE report block uses the same format as specified for the loss and duplicate report blocks in [RFC3611]. The fields "block length", "begin_seq", and "end_seq" have the same semantics and representation as defined in [RFC3611]

Block Type (BT, 8 bits): The RMCAT congestion control feedback Report Block is identified by the constant RC2F. [Note to RFC Editor: Please replace RC2F with the IANA provided RTCP XR block type for this block.]

SSRC Count (8 bits): field describes the number of SSRCs reported by this report block. The number should at least be 1.

SSRC of source (32 bits): The SSRC of the RTP source being reported upon by this report block. This report block MAY report on multiple SSRC

Report Timestamp (RTS, 32 bits): represents the timestamp when this report was generated. [FIXME: It is derived from which clock?]

Each sequence number between the begin_seq and end_seq (both inclusive) is represented by a packet metric block of 16-bits that contains the L, ECN, and ATO metrics. [FIXME: if an odd number of reports are included, i.e., end_seq - begin_seq is odd OPTION 1: pad to a 32 bit boundary? How do we mark for padding? OPTION 2: just report on higher than received RTP packet. In both cases the 16bits are set to zero. A short note on modulo operations for the sequence number may be made here?]

L (1 bit): is a boolean to indicate if the packet was received. 0 represents that the packet was not yet received and all the subsequent bits (ECN and ATO) are also set to 0. 1 represent the packet was received and the subsequent bits in the block need to be parsed.

ECN (2 bits): is the echoed ECN mark of the packet (00 if not received or if ECN is not used).

Arrival time offset (ATO, 13 bits): it the relative arrival time of the RTP packets at the receiver before this feedback report was generated measured in milliseconds. It is calculated by subtracting the reception timestamp of the RTP packet denoted by this 16bit block and the timestamp (RTS) of this report.

[FIXME: Should the timestamp of the RTP packets and the RTS be same? Needs more information if we go down this path.] [FIXME: should reserve 0xFFF to mean anything greater than 0xFFE.]

The above packet format is expressed as an RTCP XR report block when reported with regular RTCP reports. However, the same block information will need a new RTP/AVPF feedback message if reported more frequently than regular RTCP report.

4. Feedback Frequency and Overhead

There is a trade-off between speed and accuracy of reporting, and the overhead of the reports. [I-D.ietf-rmcat-rtp-cc-feedback] discusses this trade-off, and the possible rates of feedback.

It is a general understanding that the congestion control algorithms will work better with more frequent feedback - per packet feedback.

However, RTCP bandwidth and transmission rules put some upper limits on how frequently the RTCP feedback messages can be send from the media receiver to the media sender. It has been shown [I-D.ietf-rmcat-rtp-cc-feedback] that in most cases a per frame feedback is a reasonable assumption on how frequent the RTCP feedback messages can be transmitted. The design team also have noted that even if a higher frequency of feedback is desired it is not viable if the feedback messages starts to compete against the media traffic on the feedback path during congestion period. Analyzing the feedback interval requirement [feedback-requirements] it can be seen that the candidate algorithms can perform with a feedback interval range of 50-200ms. A value within this range need to be negotiated at session setup.

5. Design Rationale

The primary function of RTCP Sender Report (SR) / Receiver Report (RR) is to report the reception quality of media. The regular SR / RR reports contain information about observed jitter, fractional packet loss and cumulative packet loss. The original intent of this information was to assist flow and congestion control mechanisms. Even though it is possible to do congestion control based on information provided in the SR/RR reports it is not sufficient to design an efficient congestion control algorithm for interactive real-time communication. An efficient congestion control algorithm requires more fine grain information on per packet (see Section 3) to react to the congestion or to avoid funder congestion on the path.

Codec Control Message for AVPF [RFC5104] defines Temporary Maximum Media Bit Rate (TMMBR) message which conveys a temporary maximum bitrate limitation from the receiver of the media to the sender of the media. Even though it is not designed to replace congestion control, TMMBR has been used as a means to do receiver based congestion control where the session bandwidth is high enough to send frequent TMMBR messages especially with reduced sized reports [RFC5506]. This requires the receiver of the media to analyze the data reception, detect congestion level and recommend a maximum bitrate suitable for current available bandwidth on the path with an assumption that the sender of the media always honors the TMMBR message. This requirement is completely opposite of the sender based congestion control approach. Hence, TMMBR cannot be as a signaling means for a sender based congestion control mechanism. However, TMMBR should be viewed a complimentary signaling mechanism to establish receiver’s current view of acceptable maximum bitrate which a sender based congestion control should honor.

There are number of RTCP eXtended Report (XR) blocks have been defined for reporting of delay, loss and ECN marking. It is possible to combine several XR blocks to report the loss and ECN marking at the cost of overhead and complexity. However, there is no existing RTCP XR block to report packet arrival time.

Considering the issues discussed here it is rational to design a new congestion control feedback signaling mechanism for sender based congestion control algorithm.

6. Acknowledgements

This document is an outcome of RMCAT design team discussion. We would like to thank all participants specially Xiaoquing Zhu, Stefan Holmer, David, Ingemar Johansson and Randell Jesup for their valuable contribution to the discussions and to the document.

7. IANA Considerations

7.1. RTP/AVPF Transport Layer Feedback Message

TBD

7.2. RTCP XR Report Blocks

TBD

8. Security Considerations

There is a risk of causing congestion if an on-path attacker modifies the feedback messages in such a manner to make available bandwidth greater than it is in reality. [More on security consideration TBD.]

9. References

9.1. Normative References

[I-D.ietf-rmcat-rtp-cc-feedback]
Perkins, C., "Using RTP Control Protocol (RTCP) Feedback for Unicast Multimedia Congestion Control", draft-ietf-rmcat-rtp-cc-feedback-00 (work in progress), July 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <http://www.rfc-editor.org/info/rfc3551>.

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <http://www.rfc-editor.org/info/rfc3611>.

[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, DOI 10.17487/RFC4585, July 2006, <http://www.rfc-editor.org/info/rfc4585>.

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <http://www.rfc-editor.org/info/rfc5506>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <http://www.rfc-editor.org/info/rfc6679>.

9.2. Informative References

[feedback-requirements] "RMCAT Feedback Requirements", <://www.ietf.org/proceedings/95/slides/slides-95-rmcat-1.pdf>.

[I-D.ietf-rmcat-gcc] Holmer, S., Lundin, H., Carlucci, G., Cicco, L., and S. Mascolo, "A Google Congestion Control Algorithm for Real-Time Communication", draft-ietf-rmcat-gcc-01 (work in progress), October 2015.

[I-D.ietf-rmcat-nada]
Zhu, X., Pan, R., Ramalho, D., Cruz, S., Jones, P., Fu, J., D'Aronco, S., and C. Ganzhorn, "NADA: A Unified Congestion Control Scheme for Real-Time Media", draft-ietf-rmcat-nada-02 (work in progress), March 2016.

[I-D.ietf-rmcat-sbd]
Hayes, D., Ferlin, S., Welzl, M., and K. Hiorth, "Shared Bottleneck Detection for Coupled Congestion Control for RTP Media.", draft-ietf-rmcat-sbd-04 (work in progress), March 2016.

[I-D.ietf-rmcat-scream-cc]
Johansson, I. and Z. Sarker, "Self-Clocked Rate Adaptation for Multimedia", draft-ietf-rmcat-scream-cc-05 (work in progress), June 2016.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <http://www.rfc-editor.org/info/rfc5104>.

Authors' Addresses

Zaheduzzaman Sarker
Ericsson AB
Luleae
Sweden

Phone: +46107173743
Email: zaheduzzaman.sarker@ericsson.com

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

Email: csp@csperkins.org

Varun Singh
Nemu Dialogue Systems Oy
Runeberginkatu 4c A 4
Helsinki 00100
Finland

Email: varun.singh@iki.fi
URI: http://www.callstats.io/

Michael A. Ramalho
Cisco Systems, Inc.
6310 Watercrest Way Unit 203
Lakewood Ranch, FL 34202
USA

Phone: +1 919 476 2038
Email: mramalho@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2016

E. Berger
S. Nandakumar
M. Zanaty
Cisco Systems

March 21, 2016

Frame Marking RTP Header Extension
draft-ietf-avtext-framemarking-01

Abstract

This document describes a Frame Marking RTP header extension used to convey information about video frames that is critical for error recovery and packet forwarding in RTP middleboxes or network nodes. It is most useful when media is encrypted, and essential when the middlebox or node has no access to the media encryption keys. It is also useful for codec-agnostic processing of encrypted or unencrypted media, while it also supports extensions for codec-specific information.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Many widely deployed RTP topologies used in modern voice and video conferencing systems include a centralized component that acts as an RTP switch. It receives voice and video streams from each participant, which may be encrypted using SRTP [RFC3711], or extensions that provide participants with private media via end-to-end encryption that excludes the switch. The goal is to provide a set of streams back to the participants which enable them to render the right media content. In a simple video configuration, for example, the goal will be that each participant sees and hears just the active speaker. In that case, the goal of the switch is to receive the voice and video streams from each participant, determine the active speaker based on energy in the voice packets, possibly using the client-to-mixer audio level RTP header extension, and select the corresponding video stream for transmission to participants; see Figure 1.

In this document, an "RTP switch" is used as a common short term for the terms "switching RTP mixer", "source projecting middlebox", "source forwarding unit/middlebox" and "video switching MCU" as discussed in [I-D.ietf-avtcore-rtp-topologies-update].

```
+---+      +------------+      +---+
| A |<---->|            |<---->| B |
+---+      |            |      +---+
           |   RTP      |
+---+      |  Switch    |      +---+
| C |<---->|            |<---->| D |
+---+      +------------+      +---+
```

Figure 1: RTP switch

In order to properly support switching of video streams, the RTP switch typically needs some critical information about video frames in order to start and stop forwarding streams.

- o Because of inter-frame dependencies, it should ideally switch video streams at a point where the first frame from the new speaker can be decoded by recipients without prior frames, e.g switch on an intra-frame.
- o In many cases, the switch may need to drop frames in order to realize congestion control techniques, and needs to know which frames can be dropped with minimal impact to video quality.
- o Furthermore, it is highly desirable to do this in a way which is not specific to the video codec. Nearly all modern video codecs share common concepts around frame types.
- o It is also desirable to be able to do this for SRTP without requiring the video switch to decrypt the packets. SRTP will encrypt the RTP payload format contents and consequently this data is not usable for the switching function without decryption, which may not even be possible in the case of end-to-end encryption of private media.

A comprehensive discussion of SFU considerations around codec agnostic selective forwarding of RTP media is described in [I-D.draft-aboba-avtcore-sfu-rtp]

By providing meta-information about the RTP streams outside the encrypted media payload an RTP switch can do selective forwarding without decrypting the payload. This document provides a solution to this problem.

2. Solution

The solution uses RTP header extensions as defined in [RFC5285]. A subset of meta-information from the video stream is provided as an RTP header extension to allow an RTP switch to do generic selective forwarding of video streams encoded with potentially different video codecs.

2.1. Mandatory Extension

The following information are extracted from the media payload and sent in the Frame Marking RTP header extension.

- S: Start of Frame (1 bit) - MUST be 1 in the first packet in a frame within a layer; otherwise MUST be 0.
- E: End of Frame (1 bit) - MUST be 1 in the last packet in a frame within a layer; otherwise MUST be 0.
- I: Independent Frame (1 bit) - MUST be 1 for frames that can be decoded independent of prior frames, e.g. intra-frame, VPx keyframe, H.264 IDR [RFC6184], H.265 CRA/BLA; otherwise MUST be 0.
- D: Discardable Frame (1 bit) - MUST be 1 for frames that can be dropped, and still provide a decodable media stream; otherwise MUST be 0.
- B: Base Layer Sync (1 bit) - MUST be 1 if this frame only depends on the base layer; otherwise MUST be 0.
- TID: Temporal ID (3 bits) - The base temporal layer starts with 0, and increases with 1 for each higher temporal layer/sub-layer.
- LID: Layer ID (8 bits) - Identifies the spatial and quality layer encoded.

The layer information contained in TID and LID convey useful aspects of the layer structure that can be utilized in selective forwarding. Without further information about the layer structure, these identifiers can only be used for relative priority of layers. They convey a layer hierarchy with TID=0 and LID=0 identifying the base layer. Higher values of TID identify higher temporal layers with higher frame rates. Higher values of LID identify higher spatial or quality layers with higher resolutions and bitrates.

With further information, for example, possible future RTCP SDES items that convey full layer structure information, it may be possible to map these TIDs and LIDs to specific frame rates, resolutions and bitrates. Such additional layer information may be useful to forwarding decisions in the RTP switch, but is beyond the scope of this memo. The relative layer information is still useful for many selective forwarding decisions even without such additional layer information.

The Frame Marking RTP header extension is encoded using the one-byte header as described in [RFC5285] as shown below.

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=2 |  L=1  |S|E|I|D|B| TID |   LID         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.2. Layer ID Mappings

2.2.1. H265 LID Mapping

The following shows H265-LayerID (6 bits) mapped to the generic LID field.

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=2 |  L=1  |S|E|I|D|B| TID |0|0|  LayerID  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.2.2. VP9 LID Mapping

The following shows VP9 Layer encoding information (4 bits for spatial and quality) mapped to the generic LID field.

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=2 |  L=1  |S|E|I|D|B| TID |0|0|0|0| RS| RQ|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.2.3. VP8 LID Mapping

The following shows the header extension for VP8 that contains no layer information.

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=2 |  L=1  |S|E|I|D|B| TID |0|0|0|0|0|0|0|0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.2.4. H264-SVC LID Mapping

The following shows H264-SVC Layer encoding information (3 bits for spatial and 4 bits quality) mapped to the generic LID field.

```
  0                   1                   2
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  ID=2 |  L=1  |S|E|I|D|B| TID |0| DID |  QID  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.2.5. H264 (AVC) LID Mapping

The following shows the header extension for H264 (AVC) that contains no layer information.

```
  0                   1                   2
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  ID=2 |  L=1  |S|E|I|D|B| TID |0|0|0|0|0|0|0|0|
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.3. Signaling information

The URI for declaring this header extension in an extmap attribute is "urn:ietf:params:rtp-hdrext:framemarking". It does not contain any extension attributes.

An example attribute line in SDP:

```
a=extmap:3 urn:ietf:params:rtp-hdrext:framemarking
```

2.4. Considerations on use

The header extension values MUST represent what is already in the RTP payload.

When a RTP switch needs to discard a received video frame due to congestion control considerations, it is RECOMMENDED that it preferably drop frames marked with the "discardable" bit.

When a RTP switch wants to forward a new video stream to a receiver, it is RECOMMENDED to select the new video stream from the first switching point (I bit set) and forward the same. A RTP switch can request a media source to generate a switching point for H.264 by sending Full Intra Request (RTCP FIR) as defined in [RFC5104], for example.

3. Security Considerations

In the Secure Real-Time Transport Protocol (SRTP) [RFC3711], RTP header extensions are authenticated but not encrypted. When header extensions are used some of the payload type information are exposed and is visible to middle boxes. The encrypted media data is not exposed, so this is not seen as a high risk exposure.

4. Acknowledgements

Many thanks to Bernard Aboba, Jonathan Lennox for their inputs.

5. IANA Considerations

This document defines a new extension URI to the RTP Compact HeaderExtensions sub-registry of the Real-Time Transport Protocol (RTP) Parameters registry, according to the following data:

Extension URI: urn:ietf:params:rtp-hdrext:framemarkinginfo
Description: Frame marking information for video streams
Contact: espeberg@cisco.com
Reference: RFC XXXX

Note to RFC Editor: please replace RFC XXXX with the number of this RFC.

6. References

6.1. Normative References

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

6.2. Informative References

[I-D.ietf-avtcore-rtp-topologies-update]
Westerlund, M. and S. Wenger, "RTP Topologies", draft-ietf-avtcore-rtp-topologies-update (work in progress), April 2013.

[I-D.draft-aboba-avtcore-sfu-rtp]
Aboba, B., "Codec-Independent Selective Forwarding", raft-aboba-avtcore-sfu-rtp-00 (work in progress), July 2015.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <http://www.rfc-editor.org/info/rfc3711>.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <http://www.rfc-editor.org/info/rfc5104>.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <http://www.rfc-editor.org/info/rfc5285>.

[RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <http://www.rfc-editor.org/info/rfc6184>.

Authors' Addresses

Espen Berger
Cisco Systems

Phone: +47 98228179
Email: espeberg@cisco.com

Suhas Nandakumar
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
US

Email: snandaku@cisco.com

Mo Zanaty
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
US

Email: mzanaty@cisco.com

Payload Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2017

J. Lennox
D. Hong
Vidyo
J. Uberti
S. Holmer
M. Flodman
Google

July 8, 2016

The Layer Refresh Request (LRR) RTCP Feedback Message
draft-ietf-avtext-lrr-03

Abstract

This memo describes the RTCP Payload-Specific Feedback Message "Layer Refresh Request" (LRR), which can be used to request a state refresh of one or more substreams of a layered media stream. It also defines its use with several RTP payloads for scalable media formats.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This memo describes an RTCP [RFC3550] Payload-Specific Feedback Message [RFC4585] "Layer Refresh Request" (LRR). It is designed to allow a receiver of a layered media stream to request that one or more of its substreams be refreshed, such that it can then be decoded by an endpoint which previously was not receiving those layers, without requiring that the entire stream be refreshed (as it would be if the receiver sent a Full Intra Request (FIR); [RFC5104] see also [I-D.ietf-avtext-avpf-ccm-layered]).

The feedback message is applicable both to temporally and spatially scaled streams, and to both single-stream and multi-stream scalability modes.

2. Conventions, Definitions and Acronyms

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].

2.1. Terminology

A "Layer Refresh Point" is a point in a scalable stream after which a decoder, which previously had been able to decode only some (possibly none) of the available layers of stream, is able to decode a greater number of the layers.

For spatial (or quality) layers, layer refresh typically requires that a spatial layer be encoded in a way that references only lower-layer subpictures of the current picture, not any earlier pictures of that spatial layer. Additionally, the encoder must promise that no earlier pictures of that spatial layer will be used as reference in the future.

In a layer refresh, however, other layers than the ones requested for refresh may still maintain dependency on earlier content of the stream. This is the difference between a layer refresh and a Full Intra Request [RFC5104]. This minimizes the coding overhead of refresh to only those parts of the stream that actually need to be refreshed at any given time.

An illustration of spatial layer refresh of an enhancement layer is shown below.

```
 ... <--  S1  <--  S1       S1  <--  S1  <-- ...
           |        |        |        |
          \/       \/       \/       \/
 ... <--  S0  <--  S0  <--  S0  <--  S0  <-- ...

           1        2        3        4
```

In this illustration, frame 3 is a layer refresh point for spatial layer S1; a decoder which had previously only been decoding spatial layer S0 would be able to decode layer S1 starting at frame 3.

Figure 1

An illustration of spatial layer refresh of a base layer is shown below.

```
     ... <--  S1  <--  S1  <--  S1  <--  S1  <-- ...
               |        |        |        |
              \/       \/       \/       \/
     ... <--  S0  <--  S0       S0  <--  S0  <-- ...

               1        2        3        4
```

In this illustration, frame 3 is a layer refresh point for spatial layer S0; a decoder which had previously not been decoding the stream at all could decode layer S0 starting at frame 3.

Figure 2

For temporal layers, layer refresh requires that the layer be "temporally nested", i.e. use as reference only earlier frames of a lower temporal layer, not any earlier frames of this temporal layer, and also promise that no future frames of this temporal layer will reference frames of this temporal layer before the refresh point. In many cases, the temporal structure of the stream will mean that all frames are temporally nested, in which case decoders will have no need to send LRR messages for the stream.

An illustration of temporal layer refresh is shown below.

```
        ...  <----- T1  <------ T1          T1  <------ ...
                   /           /           /
                 |_          |_          |_
     ... <--  T0  <------ T0  <------ T0  <------ T0  <--- ...

               1     2     3     4     5     6     7
```

In this illustration, frame 6 is a layer refresh point for temporal layer T1; a decoder which had previously only been decoding temporal layer T0 would be able to decode layer T1 starting at frame 6.

Figure 3

An illustration of an inherently temporally nested stream is shown below.

```
                   T1          T1          T1
                  /           /           /
                |_          |_          |_
     ... <--  T0  <------ T0  <------ T0  <------ T0  <--- ...

              1     2     3     4     5     6     7
```

In this illustration, the stream is temporally nested in its ordinary structure; a decoder receiving layer T0 can begin decoding layer T1 at any point.

Figure 4

3. Layer Refresh Request

A layer refresh frame can be requested by sending a Layer Refresh Request (LRR), which is an RTCP payload-specific feedback message [RFC4585] asking the encoder to encode a frame which makes it possible to upgrade to a higher layer. The LRR contains one or two tuples, indicating the layer the decoder wants to upgrade to, and (optionally) the currently highest layer the decoder can decode.

The specific format of the tuples, and the mechanism by which a receiver recognizes a refresh frame, is codec-dependent. Usage for several codecs is discussed in Section 4.

LRR follows the model of the Full Intra Request (FIR) [RFC5104](Section 3.5.1) for its retransmission, reliability, and use in multipoint conferences.

The LRR message is identified by RTCP packet type value PT=PSFB and FMT=TBD. The FCI field MUST contain one or more LRR entries. Each entry applies to a different media sender, identified by its SSRC.

3.1. Message Format

The Feedback Control Information (FCI) for the Layer Refresh Request consists of one or more FCI entries, the content of which is depicted in Figure 5. The length of the LRR feedback message MUST be set to 2+3*N, where N is the number of FCI entries.

```
  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.       |C| Payload Type| Reserved                      |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | RES     | TTID| TLID          | RES     | CTID| CLID (opt)    |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5

SSRC (32 bits) The SSRC value of the media sender that is requested to send a layer refresh point.

Seq nr. (8 bits) 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 command. A repetition SHALL NOT increase the sequence number. The initial value is arbitrary.

C (1 bit) A flag bit indicating whether the "Current Layer Index" field is present in the FCI. If this bit is false, the sender of the LRR message is requesting refresh of all layers up to and including the target layer.

Payload Type (7 bits) The RTP payload type for which the LRR is being requested. This gives the context in which the target layer index is to be interpreted.

Reserved (RES) (16 bits / 5 bits / 5 bits) All bits SHALL be set to 0 by the sender and SHALL be ignored on reception.

Target Temporal Layer ID (TTID) (3 bits) The temporal ID of the target layer for which the receiver wishes a refresh point.

Target Layer ID (TLID) (8 bits) The layer ID of the target layer for which the receiver wishes a refresh point. Its format is dependent on the payload type field.

Current Temporal Layer ID (CTID) (3 bits) If C is 1, the ID of the current temporal layer being decoded by the receiver. This message is not requesting refresh of layers at or below this layer. If C is 0, this field SHALL be set to 0 by the sender and SHALL be ignored on reception.

Current Layer ID (CLID) (8 bits) If C is 1, the layer ID of the current layer being decoded by the receiver. This message is not requesting refresh of layers at or below this layer. If C is 0, this field SHALL be set to 0 by the sender and SHALL be ignored on reception.

Note: the syntax of the TTID, TLID, CTID, and TLID fields are designed to match the TID and LID fields in [I-D.ietf-avtext-framemarking].

3.2. Semantics

Within the common packet header for feedback messages (as defined in section 6.1 of [RFC4585]), the "SSRC of packet sender" field indicates the source of the request, and the "SSRC of media source" is not used and SHALL be set to 0. The SSRCs of the media senders to which the LRR command applies are in the corresponding FCI entries. A LRR message MAY contain requests to multiple media senders, using one FCI entry per target media sender.

Upon reception of LRR, the encoder MUST send a decoder refresh point (see section Section 2.1) as soon as possible.

The sender MUST consider congestion control as outlined in section 5 of [RFC5104], which MAY restrict its ability to send a layer refresh point quickly.

4. Usage with specific codecs

In order for LRR to be used with a scalable codec, the format of the target layer and current target layer fields needs to be specified for that codec’s RTP packetization. New RTP packetization specifications for scalable codecs SHOULD define how this is done. (The VP9 payload [I-D.ietf-payload-vp9], for instance, has done so.) If the payload also specifies how it is used with the Frame Marking RTP Header Extension [I-D.ietf-avtext-framemarking], the syntax MUST be defined in the same manner as the TID and LID fields in that header.

4.1. H264 SVC

H.264 SVC [RFC6190] defines temporal, dependency (spatial), and quality scalability modes.

```
+---------------+---------------+
|0|1|2|3|4|5|6|7|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RES     | TID |R|  DID  | QID |
+---------------+---------------+
```

Figure 6

Figure 6 shows the format of the layer index field for H.264 SVC streams. The "R" and "RES" fields MUST be set to 0 on transmission and ignored on reception. See [RFC6190] Section 1.1.3 for details on the DID, QID, and TID fields.

A dependency or quality layer refresh of a given layer in H.264 SVC can be identified by the "I" bit (idr_flag) in the extended NAL unit header, present in NAL unit types 14 (prefix NAL unit) and 20 (coded scalable slice). Layer refresh of the base layer can also be identified by its NAL unit type of its coded slices, which is "5" rather than "1". A dependency or quality layer refresh is complete once this bit has been seen on all the appropriate layers (in decoding order) above the current layer index (if any, or beginning from the base layer if not) through the target layer index.

Note that as the "I" bit in a PACSI header is set if the corresponding bit is set in any of the aggregated NAL units it describes; thus, it is not sufficient to identify layer refresh when NAL units of multiple dependency or quality layers are aggregated.

In H.264 SVC, temporal layer refresh information can be determined from various Supplemental Encoding Information (SEI) messages in the bitstream.

Whether an H.264 SVC stream is scalably nested can be determined from the Scalability Information SEI message’s temporal_id_nesting flag. If this flag is set in a stream’s currently applicable Scalability Information SEI, receivers SHOULD NOT send temporal LRR messages for that stream, as every frame is implicitly a temporal layer refresh point. (The Scalability Information SEI message may also be available in the signaling negotiation of H.264 SVC, as the sprop-scalability-info parameter.)

If a stream’s temporal_id_nesting flag is not set, the Temporal Level Switching Point SEI message identifies temporal layer switching points. A temporal layer refresh is satisfied when this SEI message is present in a frame with the target layer index, if the message’s delta_frame_num refers to a frame with the requested current layer index. (Alternately, temporal layer refresh can also be satisfied by a complete state refresh, such as an IDR.) Senders which support receiving LRR for non-temporally-nested streams MUST insert Temporal Level Switching Point SEI messages as appropriate.

4.2. VP8

The VP8 RTP payload format [RFC7741] defines temporal scalability modes. It does not support spatial scalability.

```
+---------------+---------------+
|0|1|2|3|4|5|6|7|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RES     | TID | RES           |
+---------------+---------------+
```

Figure 7

Figure 7 shows the format of the layer index field for VP8 streams. The "RES" fields MUST be set to 0 on transmission and be ignored on reception. See [RFC7741] Section 4.2 for details on the TID field.

A VP8 layer refresh point can be identified by the presence of the "Y" bit in the VP8 payload header. When this bit is set, this and all subsequent frames depend only on the current base temporal layer. On receipt of an LRR for a VP8 stream, A sender which supports LRR MUST encode the stream so it can set the Y bit in a packet whose temporal layer is at or below the target layer index.

Note that in VP8, not every layer switch point can be identified by the Y bit, since the Y bit implies layer switch of all layers, not just the layer in which it is sent. Thus the use of LRR with VP8 can result in some inefficiency in transmision. However, this is not expected to be a major issue for temporal structures in normal use.

4.3. H265

The initial version of the H.265 payload format [RFC7798] defines temporal scalability, with protocol elements reserved for spatial or other scalability modes (which are expected to be defined in a future version of the specification).

```
+---------------+---------------+
|0|1|2|3|4|5|6|7|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RES     | TID |RES|  LayerId  |
+---------------+---------------+
```

Figure 8

Figure 8 shows the format of the layer index field for H.265 streams. The "RES" fields MUST be set to 0 on transmission and ignored on reception. See [RFC7798] Section 1.1.4 for details on the LayerId and TID fields.

H.265 streams signal whether they are temporally nested, using the vps_temporal_id_nesting_flag in the Video Parameter Set (VPS), and the sps_temporal_id_nesting_flag in the Sequence Parameter Set (SPS). If this flag is set in a stream's currently applicable VPS or SPS, receivers SHOULD NOT send temporal LRR messages for that stream, as every frame is implicitly a temporal layer refresh point.

If a stream's sps_temporal_id_nesting_flag is not set, the NAL unit types 2 to 5 inclusively identify temporal layer switching points. A layer refresh to any higher target temporal layer is satisfied when a NAL unit type of 4 or 5 with TID equal to 1 more than current TID is seen. Alternatively, layer refresh to a target temporal layer can be incrementally satisfied with NAL unit type of 2 or 3. In this case, given current TID = TO and target TID = TN, layer refresh to TN is satisfied when NAL unit type of 2 or 3 is seen for TID = T1, then TID = T2, all the way up to TID = TN. During this incremental process, layer refresh to TN can be completely satisfied as soon as a NAL unit type of 2 or 3 is seen.

Of course, temporal layer refresh can also be satisfied whenever any Intra Random Access Point (IRAP) NAL unit type (with values 16-23, inclusively) is seen. An IRAP picture is similar to an IDR picture in H.264 (NAL unit type of 5 in H.264) where decoding of the picture can start without any older pictures.

In the (future) H.265 payloads that support spatial scalability, a spatial layer refresh of a specific layer can be identified by NAL units with the requested layer ID and NAL unit types between 16 and 21 inclusive. A dependency or quality layer refresh is complete once NAL units of this type have been seen on all the appropriate layers (in decoding order) above the current layer index (if any, or beginning from the base layer if not) through the target layer index.

5. Usage with different scalability transmission mechanisms

Several different mechanisms are defined for how scalable streams can be transmitted in RTP. The RTP Taxonomy [RFC7656] Section 3.7 defines three mechanisms: Single RTP Stream on a Single Media Transport (SRST), Multiple RTP Streams on a Single Media Transport (MRST), and Multiple RTP Streams on Multiple Media Transports (MRMT).

The LRR message is applicable to all these mechanisms. For MRST and MRMT mechanisms, the "media source" field of the LRR FCI is set to the SSRC of the RTP stream containing the layer indicated by the Current Layer Index (if "C" is 1), or the stream containing the base encoded stream (if "C" is 0). For MRMT, it is sent on the RTP session on which this stream is sent. On receipt, the sender MUST refresh all the layers requested in the stream, simultaneously in decode order.

6. Security Considerations

All the security considerations of FIR feedback packets [RFC5104] apply to LRR feedback packets as well. Additionally, media senders receiving LRR feedback packets MUST validate that the payload types and layer indices they are receiving are valid for the stream they are currently sending, and discard the requests if not.

7. SDP Definitions

Section 7 of [RFC5104] defines SDP procedures for indicating and negotiating support for codec control messages (CCM) in SDP. This document extends this with a new codec control command, "lrr", which indicates support of the Layer Refresh Request (LRR).

Figure 9 gives a formal Augmented Backus-Naur Form (ABNF) [RFC5234] showing this grammar extension, extending the grammar defined in [RFC5104].

```
rtcp-fb-ccm-param =/ SP "lrr"    ; Layer Refresh Request
```

Figure 9: Syntax of the "lrr" ccm

The Offer-Answer considerations defined in [RFC5104] Section 7.2 apply.

8. IANA Considerations

This document defines a new entry to the "Codec Control Messages" subregistry of the "Session Description Protocol (SDP) Parameters" registry, according to the following data:

Value name: lrr

Long name: Layer Refresh Request Command

Usable with: ccm

Reference: RFC XXXX

This document also defines a new entry to the "FMT Values for PSFB Payload Types" subregistry of the "Real-Time Transport Protocol (RTP) Parameters" registry, according to the following data:

Name: LRR

Long Name: Layer Refresh Request Command

Value: TBD

Reference: RFC XXXX

9. References

9.1. Normative References

[I-D.ietf-avtext-framemarking]
Berger, E., Nandakumar, S., and M. Zanaty, "Frame Marking RTP Header Extension", draft-ietf-avtext-framemarking-01 (work in progress), March 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[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, DOI 10.17487/RFC4585, July 2006, <http://www.rfc-editor.org/info/rfc4585>.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <http://www.rfc-editor.org/info/rfc5104>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <http://www.rfc-editor.org/info/rfc6190>.

[RFC7741] Westin, P., Lundin, H., Glover, M., Uberti, J., and F. Galligan, "RTP Payload Format for VP8 Video", RFC 7741, DOI 10.17487/RFC7741, March 2016, <http://www.rfc-editor.org/info/rfc7741>.

[RFC7798] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S., and M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <http://www.rfc-editor.org/info/rfc7798>.

9.2. Informative References

[I-D.ietf-avtext-avpf-ccm-layered]
Wenger, S., Lennox, J., Burman, B., and M. Westerlund, "Using Codec Control Messages in the RTP Audio-Visual Profile with Feedback with Layered Codecs", draft-ietf-avtext-avpf-ccm-layered-01 (work in progress), May 2016.

[I-D.ietf-payload-vp9]
Uberti, J., Holmer, S., Flodman, M., Lennox, J., and D. Hong, "RTP Payload Format for VP9 Video", draft-ietf-payload-vp9-02 (work in progress), March 2016.

[RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <http://www.rfc-editor.org/info/rfc7656>.

Authors' Addresses

Jonathan Lennox
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: jonathan@vidyo.com

Danny Hong
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: danny@vidyo.com

Justin Uberti
Google, Inc.
747 6th Street South
Kirkland, WA 98033
USA

Email: justin@uberti.name

Stefan Holmer
Google, Inc.
Kungsbron 2
Stockholm 111 22
Sweden

Email: holmer@google.com

Magnus Flodman
Google, Inc.
Kungsbron 2
Stockholm 111 22
Sweden

Email: mflodman@google.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2017

A. Roach
Mozilla
S. Nandakumar
Cisco Systems
P. Thatcher
Google

July 06, 2016

RTP Stream Identifier Source Description (SDES)
draft-ietf-avtext-rid-05

Abstract

This document defines and registers two new RTCP SDES items. One, named RtpStreamId, is used for unique identification of RTP streams. The other, RepairedRtpStreamId, can be used to identify which stream a redundancy RTP stream is to be used to repair.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

RTP sessions frequently consist of multiple streams, each of which is identified at any given time by its SSRC; however, the SSRC associated with a stream is not guaranteed to be stable over its lifetime. Within a session, these streams can be tagged with a number of identifiers, including CNAMEs and MSIDs [I-D.ietf-mmusic-msid]. Unfortunately, none of these have the proper ordinality to refer to an individual stream; all such identifiers can appear in more than one stream at a time. While approaches that use unique Payload Types (PTs) per stream have been used in some applications, this is a semantic overloading of that field, and one for which its size is inadequate: in moderately complex systems that use PT to uniquely identify every potential combination of codec configuration and unique stream, it is possible to simply run out of values.

To address this situation, we define a new RTCP SDES identifier, RtpStreamId, that uniquely identifies a single RTP stream. A key motivator for defining this identifier is the ability to differentiate among different encodings of a single Source Stream that are sent simultaneously (i.e., simulcast). This need for unique identification extends to dependent streams (e.g., where layers used by a layered codec are transmitted on separate streams).

At the same time, when redundancy RTP streams are in use, we also need an identifier that connects such streams to the RTP stream for which they are providing redundancy. For this purpose, we define an additional SDES identifier, RepairedRtpStreamId. This identifier can appear only in packets associated with a redundancy RTP stream. They carry the same value as the RtpStreamId of the RTP stream that the redundant RTP stream is correcting.

2. Terminology

In this document, the terms "source stream", "encoded stream," "RTP stream", "source RTP stream", "dependent stream", "received RTP stream", and "redundancy RTP stream" are used as defined in [RFC7656].

3. Usage of RtpStreamId and RepairedRtpStreamId in RTP and RTCP

The RTP fixed header includes the payload type number and the SSRC values of the RTP stream. RTP defines how you de-multiplex streams within an RTP session; however, in some use cases, applications need further identifiers in order to effectively map the individual RTP Streams to their equivalent payload configurations in the SDP.

This specification defines two new RTCP SDES items [RFC3550]. The first item is 'RtpStreamId', which is used to carry RTP stream identifiers within RTCP SDES packets. This makes it possible for a receiver to associate received RTP packets (identifying the RTP stream) with a media description having the format constraint specified. The second is 'RepairedRtpStreamId', which can be used in redundancy RTP streams to indicate the RTP stream repaired by a redundancy RTP stream.

To be clear: the value carried in a RepairedRtpStreamId will always match the RtpStreamId value from another RTP stream in the same session. For example, if a source RTP stream is identified by RtpStreamId "A", then any redundancy RTP stream that repairs that source RTP stream will contain a RepairedRtpStreamId of "A" (if this mechanism is being used to perform such correlation). These redundant RTP streams may also contain their own unique RtpStreamId.

This specification also uses the RTP header extension for RTCP SDES items [I-D.ietf-avtext-sdes-hdr-ext] to allow carrying RtpStreamId and RepairedRtpStreamId values in RTP packets. This allows correlation at stream startup, or after stream changes where the use of RTCP may not be sufficiently responsive. This speed of response is necessary since, in many cases, the stream cannot be properly processed until it can be identified.

3.1. RTCP 'RtpStreamId' SDES Extension

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|RtpStreamId=TBD|     length    | RtpStreamId                 ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The RtpStreamId payload is UTF-8 encoded and is not null-terminated.

RFC EDITOR NOTE: Please replace TBD with the assigned SDES identifier value.

3.2. RTCP 'RepairedRtpStreamId' SDES Extension

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Repaired...=TBD|     length    | RepairRtpStreamId           ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The RepairedRtpStreamId payload is UTF-8 encoded and is not null-terminated.

RFC EDITOR NOTE: Please replace TBD with the assigned SDES identifier value.

3.3. RTP 'RtpStreamId' and 'RepairedRtpStreamId' Header Extensions

Because recipients of RTP packets will typically need to know which streams they correspond to immediately upon receipt, this specification also defines a means of carrying RtpStreamId and RepairedRtpStreamId identifiers in RTP extension headers, using the technique described in [I-D.ietf-avtext-sdes-hdr-ext].

As described in that document, the header extension element can be encoded using either the one-byte or two-byte header, and the identification-tag payload is UTF-8 encoded, as in SDP.

As the identifier is included in an RTP header extension, there should be some consideration given to the packet expansion caused by the identifier. To avoid Maximum Transmission Unit (MTU) issues for the RTP packets, the header extension's size needs to be taken into account when the encoding media. Note that set of header extensions included in the packet needs to be padded to the next 32-bit boundary [RFC5285].

In many cases, a one-byte identifier will be sufficient to distinguish streams in a session; implementations are strongly encouraged to use the shortest identifier that fits their purposes. Implementors are warned, in particular, not to include any information in the identifier that is derived from potentially user-identifying information, such as user ID or IP address. To avoid identification of specific implementations based on their pattern of tag generation, implementations are encouraged to use a simple scheme that starts with the ASCII digit "1", and increments by one for each subsequent identifier.

4. IANA Considerations

4.1. New RtpStreamId SDES item

RFC EDITOR NOTE: Please replace RFCXXXX with the RFC number of this document.

RFC EDITOR NOTE: Please replace TBD with the assigned SDES identifier value.

This document adds the RtpStreamId SDES item to the IANA "RTCP SDES item types" registry as follows:

```
Value:          TBD
Abbrev.:        RtpStreamId
Name:           RTP Stream Identifier
Reference:      RFCXXXX
```

4.2. New RepairRtpStreamId SDES item

RFC EDITOR NOTE: Please replace RFCXXXX with the RFC number of this document.

RFC EDITOR NOTE: Please replace TBD with the assigned SDES identifier value.

This document adds the RepairedRtpStreamId SDES item to the IANA "RTCP SDES item types" registry as follows:

```
Value:          TBD
Abbrev.:        RepairedRtpStreamId
Name:           Repaired RTP Stream Identifier
Reference:      RFCXXXX
```

4.3. New RtpStreamId Header Extension URI

RFC EDITOR NOTE: Please replace RFCXXXX with the RFC number of this document.

This document defines a new extension URI in the RTP SDES Compact Header Extensions sub-registry of the RTP Compact Header Extensions registry sub-registry, as follows

Extension URI: urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
Description: RTP Stream Identifier Contact: adam@nostrum.com
Reference: RFCXXXX

The SDES item does not reveal privacy information about the user or the session contents. It serves only to bind the identity of a stream to corresponding data in a session description.

4.4. New RepairRtpStreamId Header Extension URI

RFC EDITOR NOTE: Please replace RFCXXXX with the RFC number of this document.

This document defines a new extension URI in the RTP SDES Compact Header Extensions sub-registry of the RTP Compact Header Extensions registry sub-registry, as follows

Extension URI: urn:ietf:params:rtp-hdrext:sdes:repair-rtp-sream-id
Description: RTP Repaired Stream Identifier Contact: adam@nostrum.com
Reference: RFCXXXX

The SDES item does not reveal privacy information about the user or the session contents. It serves only to bind redundancy stream to the streams they provide repair data for.

5. Security Considerations

The actual identifiers used for RtpStreamIds (and therefore RepairedRtpStreamIds) are expected to be opaque. As such, they are not expected to contain information that would be sensitive, were it observed by third-parties.

6. Acknowledgements

Many thanks for review and input from Cullen Jennings, Magnus Westerlund, Colin Perkins, Peter Thatcher, Jonathan Lennox, and Paul Kyzivat.

7. References

7.1. Normative References

[I-D.ietf-avtext-sdes-hdr-ext]
Westerlund, M., Burman, B., Even, R., and M. Zanaty, "RTP Header Extension for RTCP Source Description Items", draft-ietf-avtext-sdes-hdr-ext-07 (work in progress), June 2016.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <http://www.rfc-editor.org/info/rfc5285>.

[RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <http://www.rfc-editor.org/info/rfc7656>.

7.2. Informative References

[I-D.ietf-mmusic-msid]
Alvestrand, H., "WebRTC MediaStream Identification in the Session Description Protocol", draft-ietf-mmusic-msid-14 (work in progress), June 2016.

Authors' Addresses

Adam Roach
Mozilla

Email: adam@nostrum.com

Suhas Nandakumar
Cisco Systems

Email: snandaku@cisco.com

Peter Thatcher
Google

Email: pthatcher@google.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 12, 2016

M. Westerlund
B. Burman
Ericsson
R. Even
Huawei Technologies
M. Zanaty
Cisco Systems
June 10, 2016

RTP Header Extension for RTCP Source Description Items
draft-ietf-avtext-sdes-hdr-ext-07

Abstract

Source Description (SDES) items are normally transported in RTP control protocol (RTCP). In some cases it can be beneficial to speed up the delivery of these items. Mainly when a new source (SSRC) joins an RTP session and the receivers need this source's identity, relation to other sources, or its synchronization context, all of which may be fully or partially identified using SDES items. To enable this optimization, this document specifies a new RTP header extension that can carry SDES items.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This specification defines an RTP header extension [RFC3550][RFC5285] that can carry RTCP source description (SDES) items. Normally the SDES items are carried in their own RTCP packet type [RFC3550]. By including selected SDES items in a header extension the determination of relationship and synchronization context for new RTP streams (SSRCs) in an RTP session can be optimized. Which relationship and what information depends on the SDES items carried. This becomes a complement to using only RTCP for SDES Item delivery.

It is important to note that not all SDES items are appropriate to transmit using RTP header extensions. Some SDES items performs binding or identifies synchronization context with strict timeliness requirements, while many other SDES items do not have such requirements. In addition, security and privacy concerns for the SDES item information need to be considered. For example, the Name and Location SDES items are highly sensitive from a privacy perspective and should not be transported over the network without strong security. No use case has identified where this information is required at the same time as the first RTP packets arrive. A few seconds delay before such information is available to the receiver appears acceptable. Therefore only appropriate SDES items will be registered for use with this header extension, such as CNAME.

First, some requirements language and terminology are defined. The following section motivates why this header extension is sometimes required or at least provides a significant improvement compared to waiting for regular RTCP packet transmissions of the information. This is followed by a specification of the header extension and usage recommendations. Next, a sub-space of the header-extension URN is defined to be used for existing and future SDES items, and then the appropriate existing SDES items are registered.

2. Definitions

2.1. 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 RFC 2119 [RFC2119].

2.2. Terminology

This document uses terminology defined in "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources" [RFC7656]. In particular the following definitions:

Media Source

RTP Stream

Media Encoder

Participant

3. Motivation

Source Description (SDES) items are associated with a particular SSRC and thus RTP stream. The source description items provide various meta data associated with the SSRC. How important it is to have this data no later than when receiving the first RTP packets depends on the item itself. The CNAME item is one item that is commonly needed either at reception of the first RTP packet for this SSRC, or at least by the time the first media can be played out. If it is not available, the synchronization context cannot be determined and thus any related streams cannot be correctly synchronized. Thus, this is a valuable example for having this information early when a new RTP stream is received.

The main reason for new SSRCs in an RTP session is when media sources are added. This can be either because an end-point is adding a new actual media source, or additional participants in a multi-party session are added to the session. Another reason for a new SSRC can be an SSRC collision that forces both colliding parties to select new SSRCs.

For the case of rapid media synchronization, one may use the RTP header extension for Rapid Synchronization of RTP Flows [RFC6051]. This header extension carries the clock information present in the RTCP sender report (SR) packets. It however assumes that the CNAME binding is known, which can be provided via signaling [RFC5576] in some cases, but not all. Thus an RTP header extension for carrying SDES items like CNAME is a powerful combination to enable rapid synchronization in all cases.

The Rapid Synchronization of RTP Flows specification does provide an analysis of the initial synchronization delay for different sessions depending on number of receivers as well as on session bandwidth (Section 2.1 of [RFC6051]). These results are applicable also for other SDES items that have a similar time dependency until the information can be sent using RTCP. These figures can be used to determine the benefit of reducing the initial delay before information is available for some use cases.

Rapid Synchronization of RTP Flows [RFC6051] also discusses the case of late joiners, and defines an RTCP Feedback format to request synchronization information, which is another potential use case for SDES items in RTP header extension. It would for example be natural to include CNAME SDES item with the header extension containing the NTP formatted reference clock to ensure synchronization.

The ongoing work on bundling SDP media descriptions [I-D.ietf-mmusic-sdp-bundle-negotiation] has identified a new SDES item that can benefit from timely delivery. A corresponding RTP SDES compact header extension is therefore also defined and registered in that document:

MID: This is a media description identifier that matches the value of the Session Description Protocol (SDP) [RFC4566] a=mid attribute [RFC5888], to associate RTP streams multiplexed on the same transport with their respective SDP media description.

4. Specification

This section first specifies the SDES item RTP header extension format, followed by some usage considerations.

4.1. SDES Item Header Extension

An RTP header extension scheme allowing for multiple extensions is defined in "A General Mechanism for RTP Header Extensions" [RFC5285]. That specification defines both short and long item headers. The short headers (One-byte) are restricted to 1 to 16 bytes of data, while the long format (Two-byte) supports a data length of 0 to 255 bytes. Thus the RTP header extension formats are capable of supporting any SDES item from a data length perspective.

The ID field, independent of short or long format, identifies both the type of RTP header extension and, in the case of the SDES item header extension, the type of SDES item. The mapping is done in signaling by identifying the header extension and SDES item type using a URN, which is defined in the IANA consideration (Section 5) for the known SDES items appropriate to use.

4.1.1. One-Byte Format

The one-byte header format for an SDES item extension element consists of the one-byte header (defined in Section 4.2 of [RFC5285]), which consists of a 4-bit ID followed by a 4-bit length field (len) that identifies the number of data bytes (len value +1) following the header. The data part consists of len+1 bytes of UTF-8 [RFC3629] text. The type of text and its mapping to the SDES item type is determined by the ID field value.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID   |  len  | SDES Item text value ...                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1

4.1.2. Two-Byte Format

The two-byte header format for an SDES item extension element consists of the two-byte header (defined in Section 4.3 of [RFC5285]), which consists of an 8-bit ID followed by an 8-bit length field (len) that identifies the number of data bytes following the header. The data part consists of len bytes of UTF-8 text. The type of text and its mapping to the SDES item type is determined by the ID field value.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      ID       |      len      |  SDES Item text value ...     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2

4.2. Usage of the SDES Item Header Extension

This section discusses various usage considerations; which form of header extension to use, the packet expansion, and when to send SDES items in header extension.

4.2.1. One or Two Byte Headers

The RTP header extensions for SDES items MAY use either the one-byte or two-byte header formats, depending on the text value size for the used SDES items and the requirement from any other header extensions used. The one-byte header SHOULD be used when all non SDES item header extensions supports the one-byte format and all SDES item text values contain at most 16 bytes. Note that the RTP header extension specification does not allow mixing one-byte and two-byte headers for the same RTP stream (SSRC), so if the value size of any of the SDES items value requires the two-byte header, then all other header extensions MUST also use the two-byte header format.

For example using CNAMEs that are generated according to "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)" [RFC7022], using short term persistent values, and if 96-bit random values prior to base64 encoding are sufficient, then they will fit into the one-byte header format.

An RTP middlebox needs to take care choosing between one-byte headers and two-byte headers when creating the first packets for an outgoing stream (SSRC) with header extensions. First of all it needs to consider all the header extensions that may potentially be used. Secondly, it needs to know the size of the SDES items that are going to be included, and use two bytes headers if any are longer than 16 bytes. An RTP middlebox that forwards a stream, i.e., not mixing it or combing it with other streams, may be able to base its choice on the header size in incoming streams. This is assuming that the middlebox does not modify the stream or add additional header extensions to the stream it sends, in which case it needs to make its own decision.

4.2.2. MTU and Packet Expansion

The RTP packet size will clearly increase when a header extension is included. How much depends on the type of header extensions and their data content. The SDES items can vary in size. There are also some use-cases that require transmitting multiple SDES items in the same packet to ensure that all relevant data reaches the receiver. An example of that is when both CNAME, a MID, and the rapid time synchronization extension from RFC 6051 are needed. Such a combination is quite likely to result in at least 16+3+8 bytes of data plus the headers, which will be another 7 bytes for one-byte headers, plus two bytes of header padding to make the complete header extension 32-bit word aligned, thus in total 36 bytes.

If the packet expansion cannot be taken into account when producing the RTP payload, it can cause an issue. An RTP payload that is created to meet a particular IP level Maximum Transmission Unit (MTU), taking the addition of IP/UDP/RTP headers but not RTP header extensions into account, could exceed the MTU when the header extensions are present, thus resulting in IP fragmentation. IP fragmentation is known to negatively impact the loss rate due to middleboxes unwilling or not capable of dealing with IP fragments, as well as increasing the target surface for other types of packet losses.

As this is a real issue, the media encoder and payload packetizer should be flexible and be capable of handling dynamically varying payload size restrictions to counter the packet expansion caused by header extensions. If that is not possible, some reasonable worst case packet expansion should be calculated and used to reduce the RTP payload size of all RTP packets the sender transmits.

4.2.3. Transmission Considerations

The general recommendation is to only send header extensions when needed. This is especially true for SDES items that can be sent in periodic repetitions of RTCP throughout the whole session. Thus, the different usages (Section 4.2.4) have different recommendations. First some general considerations for getting the header extensions delivered to the receiver:

1. The probability for packet loss and burst loss determine how many repetitions of the header extensions will be required to reach a targeted delivery probability, and if burst loss is likely, what distribution would be needed to avoid getting all repetitions of the header extensions lost in a single burst.

2. If a set of packets are all needed to enable decoding, there is commonly no reason for including the header extension in all of these packets, as they share fate. Instead, at most one instance of the header extension per independently decodable set of media data would be a more efficient use of the bandwidth.

3. How early the SDES item information is needed, from the first received RTP data or only after some set of packets are received, can guide if the header extension(s) should be in all of the first N packets or be included only once per set of packets, for example once per video frame.

4. The use of RTP level robustness mechanisms, such as RTP retransmission [RFC4588], or Forward Error Correction, e.g., [RFC5109] may treat packets differently from a robustness perspective, and SDES header extensions should be added to packets that get a treatment corresponding to the relative importance of receiving the information.

As a summary, the number of header extension transmissions should be tailored to a desired probability of delivery taking the receiver population size into account. For the very basic case, N repetitions of the header extensions should be sufficient, but may not be optimal. N is selected so that the header extension target delivery probability reaches 1-P^N, where P is the probability of packet loss. For point to point or small receiver populations, it might also be possible to use feedback, such as RTCP, to determine when the information in the header extensions has reached all receivers and stop further repetitions. Feedback that can be used includes the RTCP XR Loss RLE report block [RFC3611], which will indicate successful delivery of particular packets. If the RTP/AVPF Transport Layer Feedback Messages for generic NACK [RFC4585] is used, it can indicate the failure to deliver an RTP packet with the header extension, thus indicating the need for further repetitions. The normal RTCP report blocks can also provide an indicator of successful delivery, if no losses are indicated for a reporting interval covering the RTP packets with the header extension. Note that loss of an RTCP packet reporting on an interval where RTP header extension packets were sent, does not necessarily mean that the RTP header extension packets themselves were lost.

4.2.4. Different Usages

4.2.4.1. New SSRC

A new SSRC joins an RTP session. As this SSRC is completely new for everyone, the goal is to ensure, with high probability, that all RTP session participants receives the information in the header extension. Thus, header extension transmission strategies that allow some margins in the delivery probability should be considered.

4.2.4.2. Late Joiner

In a multi-party RTP session where one or a small number of receivers join a session where the majority of receivers already have all necessary information, the use of header extensions to deliver relevant information should be tailored to reach the new receivers. The trigger to send header extensions can for example either be RTCP from new receiver(s) or an explicit request like the Rapid Resynchronization Request defined in [RFC6051]. In centralized topologies where an RTP middlebox is present, it can be responsible for transmitting the known information, possibly stored, to the new session participant only, and not repeat it to all the session participants.

4.2.4.3. Information Change

If the SDES information is tightly coupled with the RTP data, and the SDES information needs to be updated, then the use of the RTP header extension is superior to RTCP. Using the RTP header extension ensures that the information is updated on reception of the related RTP media, ensuring synchronization between the two. Continued use of the old SDES information can lead to undesired effects in the application. Thus, header extension transmission strategies with high probability of delivery should be chosen.

4.2.5. SDES Items in RTCP

The RTP header extension information, i.e., SDES items, can and will be sent also in RTCP. Therefore, it is worth making some reflections on this interaction. As an alternative to the header extension, it is possible to schedule a non-regular RTCP packet transmission containing important SDES items, if one uses an RTP/AVPF-based RTP profile. Depending on which mode one’s RTCP feedback transmitter is working on, extra RTCP packets may be sent as immediate or early packets, enabling more timely SDES information delivery.

There are however two aspects that differ between using RTP header extensions and any non-regular transmission of RTCP packets. First, as the RTCP packet is a separate packet, there is no direct relation and also no fate sharing between the relevant media data and the SDES information. The order of arrival for the packets will matter. With a header-extension, the SDES items can be ensured to arrive if the media data to play out arrives. Secondly, it is difficult to determine if an RTCP packet is actually delivered. This, as the RTCP packets lack both sequence number and a mechanism providing feedback on the RTCP packets themselves.

4.2.6. Update Flaps

The SDES item may arrive both in RTCP and in RTP header extensions, potentially causing the value to flap back and forth at the time of updating. There are at least two reasons for these flaps. The first one is packet reordering, where a pre-update RTP or RTCP packet with an SDES item is delivered to the receiver after the first RTP/RTCP packet with the updated value. The second reason is the different code-paths for RTP and RTCP in implementations. An update to the sender’s SDES item parameter can take a different time to propagate to the receiver than the corresponding media data. For example, an RTCP packet with the SDES item included that may have been generated prior to the update can still reside in a buffer and be sent unmodified. The update of the item’s value can at the same time cause RTP packets to be sent including the header extension, prior to the RTCP packet being sent.

However, most of these issues can be avoided by the receiver performing some checks before updating the receiver’s stored value. To handle flaps caused by reordering, SDES items received in RTP packets with the same or a lower extended sequence number than the last change MUST NOT be applied, i.e., discard items that can be determined to be older than the current one. For compound RTCP packets, which will contain a Sender Report (SR) packet (assuming an active RTP sender), the receiver can use the RTCP SR Timestamp field to determine at what approximate time it was transmitted. If the timestamp is earlier than the last received RTP packet with a header extension carrying an SDES item, and especially if carrying a previously used value, the SDES item in the RTCP SDES packet can be ignored. Note that media processing and transmission pacing can easily cause the RTP header timestamp field as well as the RTCP SR timestamp field to not match with the actual transmission time.

4.2.7. RTP Header Compression

When robust header compression (ROHC) [RFC5225] is used with RTP, the RTP header extension [RFC5285] data itself is not part of what is being compressed and thus does not impact header compression performance. The extension indicator (X) bit in the RTP header is however compressed. It is classified as rarely changing, which may no longer be true for all RTP header extension usage, in turn leading to lower header compression efficiency.

5. IANA Considerations

This section makes the following requests to IANA:

- o Create a new sub-registry reserved for RTCP SDES items with the URN sub-space "urn:ietf:params:rtp-hdrext:sdes:" in the RTP Compact Header Extensions registry.

- o Register the SDES items appropriate for use with the RTP header extension defined in this document.

RFC-editor note: Please replace all occurrences of RFCXXXX with the RFC number this specification receives when published.

5.1. Registration of an SDES Base URN

IANA is requested to register the below entry in the RTP Compact Header Extensions registry:

```
Extension URI: urn:ietf:params:rtp-hdrext:sdes
Description:   Reserved as base URN for RTCP SDES items that are also
               defined as RTP Compact header extensions.
Contact:       Authors of [RFCXXXX]
Reference:     [RFCXXXX]
```

The reason to register a base URN for an SDES sub-space is that the name represents an RTCP Source Description item, where a specification is strongly recommended [RFC3550].

5.2. Creation of an SDES Sub-Registry

IANA is requested to create a sub-registry to the RTP Compact Header Extensions registry, with the same basic requirements, structure and layout as the RTP Compact Header Extensions registry.

- o Registry name: RTP SDES Compact Header Extensions

- o Specification: RFCXXXX and RFCs updating RFCXXXX

- o Information required: Same as for RTP Header Extensions [RFC5285] registry o Review process: Same as for RTP Header Extensions [RFC5285] registry, with the following requirements added to the expert review:

1. Any registration using an Extension URI that starts with "urn:ietf:params:rtp-hdrext:sdes:" (Section 5.1) MUST also have a registered Source Description item in the "RTP SDES item types" registry.

2. A security and privacy consideration for the SDES item MUST be provided with the registration.

3. Information MUST be provided on why this SDES item requires timely delivery, motivating it to be transported in a header extension rather than as RTCP only.

o Size and format of entries: Same as for RTP Header Extensions [RFC5285] registry.

o Initial assignments: See Section 5.3 below.

5.3. Registration of SDES Item

It is requested that the following SDES item is registered in the newly formed RTP SDES Compact Header Extensions registry:

```
Extension URI: urn:ietf:params:rtp-hdrext:sdes:cname
Description:   Source Description: Canonical End-Point Identifier
               (SDES CNAME)
Contact:       Authors of [RFCXXXX]
Reference:     [RFCXXXX]
```

6. Security Considerations

Source Description items may contain data that are sensitive from a security perspective. There are SDES items that are or may be sensitive from a user privacy perspective, like CNAME, NAME, EMAIL, PHONE, LOC and H323-CADDR. Some may contain sensitive information, like NOTE and PRIV, while others may be sensitive from profiling implementations for vulnerability or other reasons, like TOOL. The CNAME sensitivity can vary depending on how it is generated and what persistence it has. A short term CNAME identifier generated using a random number generator [RFC7022] may have minimal security implications, while a CNAME of the form user@host has privacy concerns, and a CNAME generated from a MAC address has long term tracking potentials.

In RTP sessions where any type of confidentiality protection is enabled for RTCP, the SDES item header extensions MUST also be protected. This implies that to provide confidentiality, users of SRTP need to implement and use encrypted header extensions per [RFC6904]. SDES items carried as RTP header extensions MUST then use commensurate strength algorithms and SHOULD use the same cryptographic primitives (algorithms, modes) as applied to RTCP packets carrying corresponding SDES items. If the security level is chosen to be different for an SDES item in RTCP and RTP header extension, it is important to motivate the exception, and to consider the security properties as the worst in each aspect for the different configurations. It is worth noting that the current Secure RTP (SRTP) [RFC3711] only provides protection for the next trusted RTP/RTCP hop, which is not necessarily end-to-end.

The general RTP header extension mechanism [RFC5285] does not itself contain any functionality that is a significant risk for a denial of service attack, neither from processing nor storage requirements. The extension for SDES items defined in this document, can potentially be a risk. The risk depends on the received SDES item and its content. If the SDES item causes the receiver to perform a large amount of processing, create significant storage structures, or emit network traffic, such a risk does exist. The CNAME SDES item in the RTP header extension is only a minor risk, as reception of a CNAME item will create an association between the stream carrying the SDES item and other RTP streams with the same SDES item. This usually results in time synchronizing the media streams, thus some additional processing is performed. However, the application’s media quality is likely more affected by an erroneous or changing association and media synchronization, than the application quality impact caused by the additional processing.

As the SDES items are used by the RTP based application to establish relationships between RTP streams or between an RTP stream and information about the originating participant, there SHOULD be strong integrity protection and source authentication of the header extensions. If not, an attacker can modify the SDES item value to create erroneous relationship bindings in the receiving application. For information regarding options for securing RTP, see [RFC7201].

7. Acknowledgments

The authors likes to thank the following individuals for feedback and suggestions: Colin Perkins, Ben Campbell, and Samuel Weiler.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <http://www.rfc-editor.org/info/rfc5285>.

[RFC6904] Lennox, J., "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, DOI 10.17487/RFC6904, April 2013, <http://www.rfc-editor.org/info/rfc6904>.

8.2. Informative References

[I-D.ietf-mmusic-sdp-bundle-negotiation] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-30 (work in progress), June 2016.

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <http://www.rfc-editor.org/info/rfc3611>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <http://www.rfc-editor.org/info/rfc3629>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <http://www.rfc-editor.org/info/rfc3711>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <http://www.rfc-editor.org/info/rfc4566>.

[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, DOI 10.17487/RFC4585, July 2006, <http://www.rfc-editor.org/info/rfc4585>.

[RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, DOI 10.17487/RFC4588, July 2006, <http://www.rfc-editor.org/info/rfc4588>.

[RFC5109] Li, A., Ed., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, DOI 10.17487/RFC5109, December 2007, <http://www.rfc-editor.org/info/rfc5109>.

[RFC5225] Pelletier, G. and K. Sandlund, "RObust Header Compression Version 2 (ROHCv2): Profiles for RTP, UDP, IP, ESP and UDP-Lite", RFC 5225, DOI 10.17487/RFC5225, April 2008, <http://www.rfc-editor.org/info/rfc5225>.

[RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, DOI 10.17487/RFC5576, June 2009, <http://www.rfc-editor.org/info/rfc5576>.

[RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, DOI 10.17487/RFC5888, June 2010, <http://www.rfc-editor.org/info/rfc5888>.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, DOI 10.17487/RFC6051, November 2010, <http://www.rfc-editor.org/info/rfc6051>.

[RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla, "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)", RFC 7022, DOI 10.17487/RFC7022, September 2013, <http://www.rfc-editor.org/info/rfc7022>.

[RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, DOI 10.17487/RFC7201, April 2014, <http://www.rfc-editor.org/info/rfc7201>.

[RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <http://www.rfc-editor.org/info/rfc7656>.

Authors' Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Stockholm
Sweden

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

Bo Burman
Ericsson
Gronlandsgatan 31
Stockholm 16480
Sweden

Email: bo.burman@ericsson.com

Roni Even
Huawei Technologies
Tel Aviv
Israel

Email: roni.even@mail01.huawei.com

Mo Zanaty
Cisco Systems
7100 Kit Creek
RTP, NC 27709
USA

Email: mzanaty@cisco.com

AVTEXT Working Group
INTERNET-DRAFT
Intended Status: Standards Track
Expires: December 30, 2016

J. Xia
R. Even
R. Huang
Huawei
L. Deng
China Mobile
June 28, 2016

RTP/RTCP extension for RTP Splicing Notification
draft-ietf-avtext-splicing-notification-08

Abstract

Content splicing is a process that replaces the content of a main multimedia stream with other multimedia content, and delivers the substitutive multimedia content to the receivers for a period of time. The splicer is designed to handle RTP splicing and needs to know when to start and end the splicing.

This memo defines two RTP/RTCP extensions to indicate the splicing related information to the splicer: an RTP header extension that conveys the information in-band and an RTCP packet that conveys the information out-of-band.

Status of this Memo

Copyright and License Notice

Table of Contents

1 Introduction

Splicing is a process that replaces some multimedia content with other multimedia content and delivers the substitutive multimedia content to the receivers for a period of time. In some predictable splicing cases, e.g., advertisement insertion, the splicing duration needs to be inside of the specific, pre-designated time slot. Certain timing information about when to start and end the splicing must be first acquired by the splicer in order to start the splicing. This document refers to this information as the Splicing Interval.

[SCTE35] provides a method that encapsulates the Splicing Interval inside the MPEG2-TS layer in cable TV systems. When transported in RTP, an middle box designed as the splicer to decode the RTP packets and search for the Splicing Interval inside the payloads is required. The need for such processing increases the workload of the middle box and limits the number of RTP sessions the middle box can support.

The document defines an RTP header extension [RFC5285] used by the main RTP sender to provide the Splicing Interval by including it in the RTP packets.

However, the Splicing Interval conveyed in the RTP header extension might not reach the splicer successfully. Any splicing un-aware middlebox on the path between the RTP sender might strip this RTP header extension.

To increase robustness against such case, the document also defines a new RTCP packet type to carry the same Splicing Interval to the splicer. Since RTCP is also unreliable and may not be so immediate as the in-band way, it’s only considered as a complement to the RTP header extension.

1.1 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].

In addition, we define following terminologies:

Main RTP sender:

The sender of RTP packets carrying the main RTP stream.

Splicer:

An intermediary node that inserts substitutive content into a main

RTP stream. The splicer sends substitutive content to the RTP receiver instead of the main content during splicing. It is also responsible for processing RTCP traffic between the RTP sender and the RTP receiver.

Splicing-In Point

A virtual point in the RTP stream, suitable for substitutive content entry, typically in the boundary between two independently decodable frames.

Splicing-Out Point

A virtual point in the RTP stream, suitable for substitutive content exit, typically in the boundary between two independently decodable frames.

Splicing Interval:

The NTP-format timestamps, representing the main RTP sender wallclock time, for the Splicing-In point and Splicing-Out point per [RFC6828] allowing the splicer to know when to start and end the RTP splicing.

Substitutive RTP Sender:

The sender of RTP packets carrying the RTP stream that will replace the content in the main RTP stream.

2 Overview

2.1 Overview of RTP Splicing

RTP Splicing is intended to replace some multimedia content with certain substitutive multimedia content, and then forward it to the receivers for a period of time. This process is authorized by the main RTP sender that offers a specific time window for inserting the substitutive multimedia content in the main content. A typical usage is that IPTV service provider uses its own regional advertising content to replace national advertising content, the time window of which is explicitly indicated by the IPTV service provider.

The splicer is a middlebox handling RTP splicing. It receives main content and substitutive content simultaneously but only chooses to send one of them to the receiver at any point of time. When RTP splicing begins, the splicer sends the substitutive content to the receivers instead of the main content. When RTP splicing ends, the splicer switches back to sending the main content to the receivers.

This implies that the receiver is explicitly configured to receive the traffic via the splicer, and will return any RTCP feedback to it in the presence of the splicer.

The middlebox working as the splicer can be implemented as either an RTP mixer or as an RTP translator. If implemented as an RTP mixer, [RFC6828] specifies how the splicer can use its own SSRC, sequence number space, and timing model when generating the output stream to receivers, using the CSRC list to indicate whether the original or substitutive content is being delivered. The splicer, on behalf of the content provider, can omit the CSRC list from the RTP packets it generates. This simplifies the design of the receivers, since they don’t need to parse the CSRC list, but makes it harder to determine when the splicing is taking place (it requires inspection of the RTP payload data, rather than just the RTP headers). A splicer working as an RTP mixer splits the flow between the sender and receiver into two, and requires separate control loops, for RTCP and congestion control, see Section 4.4 of [RFC6828].

A splicer implemented as an RTP translator [RFC3550] will forward the RTP packets from the original and substitutive senders with their SSRCs intact, but will need to rewrite RTCP sender report packets to account for the splicing. In this case, the congestion control loops run between original sender and receiver, and between the substitutive sender and receiver. The splicer needs to ensure that the RTCP feedback message from the receiver are passed to the right sender to let the congestion control work.

2.2 Overview of Splicing Interval

To handle splicing on the RTP layer at the reserved time slots set by the main RTP sender, the splicer must first know the Splicing Interval from the main RTP sender before it can start splicing. The splicer can be a mixer as described in [RFC6828].

When a new splicing is forthcoming, the main RTP sender needs to send the Splicing Interval to the splicer. The Splicing Interval SHOULD be sent by RTP header extension or RTCP extension message more than once to mitigate the possible packet loss. To enable the splicer to get the substitutive content before the splicing starts, the main RTP sender MUST send the Splicing Interval far ahead. For example, the main RTP sender can estimate when to send the Splicing Interval based on the round-trip time (RTT) following the mechanisms in section 6.4.1 of [RFC3550] when the splicer sends RTCP RR to the main sender.

The substitutive sender also needs to learn the Splicing Interval from the main RTP sender in advance, and thus estimates when to transfer the substitutive content to the splicer. The Splicing

Interval could be transmitted from the main RTP sender to the substitutive content using some out-of-band mechanisms, for example, a proprietary mechanism to exchange the Splicing Interval, or the substitutive sender is implemented together with the main RTP sender inside a single device. To ensure the Splicing Interval is valid for both the main RTP sender and the substitutive RTP sender, the two senders MUST share a common reference clock so that the splicer can achieve accurate splicing. The requirements for the common reference clock (e.g. resolution, skew) depend on the codec used by the media content.

In this document, the main RTP sender uses a pair of NTP-format timestamps, to indicate when to start and end the splicing to the splicer: the timestamp of the first substitutive RTP packet at the splicing in point, and the timestamp of the first main RTP packet at the splicing out point.

When the substitutive RTP sender gets the Splicing Interval, it must prepare the substitutive stream. The main and the substitutive content providers MUST ensure that the RTP timestamp of the first substitutive RTP packet that would be presented to the receivers corresponds to the same time instant as the former NTP-format timestamp in the Splicing Interval. To enable the splicer to know the first substitutive RTP packet it needs to send, the substitutive RTP sender MUST send the substitutive RTP packet ahead of the Splicing In point, allowing the splicer to find out the timestamp of this first RTP packet in the substitutive RTP stream, e.g., using a prior RTCP SR (Sender Report) message.

When the splicing will end, the main content provider and the substitutive content provider MUST ensure the RTP timestamp of the first main RTP packet that would be presented on the receivers corresponds to the same time instant as the latter NTP-format timestamp in the Splicing Interval.

3 Conveying Splicing Interval in RTP/RTCP extensions

This memo defines two backwards compatible RTP extensions to convey the Splicing Interval to the splicer: an RTP header extension and an RTCP splicing notification message.

3.1 RTP Header Extension

The RTP header extension mechanism defined in [RFC5285] can be adapted to carry the Splicing Interval consisting of a pair of NTP-format timestamps.

This RTP header extension carries the 7 octets splicing-out NTP-format timestamp (lower 24-bit part of the Seconds of a NTP-format timestamp and the 32 bits of the Fraction of a NTP-format timestamp as defined in [RFC5905]), followed by the 8 octets splicing-in NTP-format timestamp (64-bit NTP-format timestamp as defined in [RFC5905]). The top 8 bits of the splicing-out NTP timestamp are inferred from the top 8 bits of the splicing-in NTP timestamp, under the assumption that the splicing-out time is after the splicing-in time, and the splicing interval is less than 2^25 seconds. Therefore, if the value of 7 octets splicing-out NTP-format timestamp is smaller than the value of 7 lower octets splicing-in NTP-format, it implies a wrap of the 56-bit splicing-out NTP-format timestamp which means the top 8-bit value of the 64-bit splicing-out is equal to the top 8-bit value of splicing-in NTP Timestamp plus 0x01. Otherwise, the top 8 bits of splicing-out NTP timestamp is equal to the top 8 bits of splicing-in NTP Timestamp.

This RTP header extension can be encoded using either the one-byte or two-byte header defined in [RFC5285]. Figure 1 and 2 show the splicing interval header extension with each of the two header formats.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+E
|   ID  | L=14  | OUT NTP timestamp format - Seconds (bit 8-31) |x
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+t
|       OUT NTP timestamp format - Fraction (bit 0-31)          |e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+n
|        IN  NTP timestamp format - Seconds (bit 0-31)          |s
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+i
|        IN  NTP timestamp format - Fraction (bit 0-31)         |o
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+n
```

Figure 1: Splicing Interval
Using the One-Byte Header Format

```
        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
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+E
      |   ID          |    L=15       | OUT NTP timestamp  - Seconds  |x
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+t
      |Out Secds(cont)|      OUT NTP timestamp format - Fraction      |e
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+n
      |Out Fract(cont)|      IN  NTP timestamp format - Seconds       |s
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+i
      | In Secds(cont)|     IN  NTP timestamp format - Fraction       |o
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+n
      | In Fract(cont)| 0 (pad)       |              ...
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Splicing Interval
Using the Two-Byte Header Format

Since the inclusion of an RTP header extension will reduce the efficiency of RTP header compression, it is RECOMMENDED that the main sender inserts the RTP header extensions into only a number of RTP packets, instead of all the RTP packets, prior to the splicing in.

After the splicer obtains the RTP header extension and derives the Splicing Interval, it generates its own stream and is not allowed to include the RTP header extension in outgoing packets to reduce header overhead.

3.2 RTCP Splicing Notification Message

In addition to the RTP header extension, the main RTP sender includes the Splicing Interval in an RTCP splicing notification message. Whether or not the timestamps are included in the RTP header extension, the main RTP sender MUST send the RTCP splicing notification message. This provide robustness in the case where a middlebox strips RTP header extensions. The main RTP sender MUST make sure the splicing information contained in the RTCP splicing notification message consistent with the information included in the RTP header extensions.

The RTCP splicing notification message is a new RTCP packet type. It has a fixed header followed by a pair of NTP-format timestamps:

```
         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=TBA     |             length            |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             SSRC                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             IN NTP Timestamp (most significant word)          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             IN NTP Timestamp (least significant word)         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             OUT NTP Timestamp (most significant word)         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             OUT NTP Timestamp (least significant word)        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: RTCP Splicing Notification Message

The RSI packet includes the following fields:

Length: 16 bits

As defined in [RFC3550], the length of the RTCP packet in 32-bit words minus one, including the header and any padding.

SSRC: 32 bits

The SSRC of the Main RTP Sender.

Timestamp: 64 bits

Indicates the wallclock time when this splicing starts and ends. The full-resolution NTP-format timestamp is used, which is a 64-bit, unsigned, fixed-point number with the integer part in the first 32 bits and the fractional part in the last 32 bits. This format is same as the NTP timestamp field in the RTCP Sender Report (Section 6.4.1 of [RFC3550]).

The RTCP splicing notification message can be included in the RTCP compound packet together with RTCP SR generated at the main RTP sender, and hence follows the compound RTCP rules defined in Section 6.1 in [RFC3550].

If the use of non-compound RTCP [RFC5506] was previously negotiated between the sender and the splicer, the RTCP splicing notification message may be sent as non-compound RTCP packets. In some cases that the mapping from RTP timestamp to NTP timestamp changes, e.g., clock drift happening before the splicing event, it may be required to send RTCP SR or even updated Splicing Interval information timely to update the timestamp mapping for accurate splicing.

Since the RTCP splicing notification message is intentionally sent by the main RTP sender to the splicer, the splicer is not allowed to forward this message to the receivers so as to avoid their useless processing and additional RTCP bandwidth consumption in the downstream.

4 Reducing Splicing Latency

When splicing starts or ends, the splicer outputs the multimedia content from another sender to the receivers. Given that the receivers must first acquire certain information ([RFC6285] refers to this information as Reference Information) to start processing the multimedia data, either the main RTP sender or the substitutive sender SHOULD provide the Reference Information together with its multimedia content to reduce the delay caused by acquiring the Reference Information. The methods by which the Reference Information is distributed to the receivers is out of scope of this memo.

Another latency element is synchronization caused delay. The receivers must receive enough synchronization metadata prior to synchronizing the separate components of the multimedia streams when splicing starts or ends. Either the main RTP sender or the substitutive sender SHOULD send the synchronization metadata early enough so that the receivers can play out the multimedia in a synchronized fashion. The main RTP sender or the substitutive sender can estimate when to send the synchronization metadata based on, for example, the round-trip time (RTT) following the mechanisms in section 6.4.1 of [RFC3550] when the splicer sends RTCP RR to the main sender or the substitutive sender. The main RTP sender and the substitutive sender can also be coordinated by some proprietary out-of-band mechanisms to decide when and whom to send the metadata. If both send the information, the splicer SHOULD pick one based on the current situation, e.g., choosing main RTP sender when synchronizing the main media content while choosing the information from the substitutive sender when synchronizing the spliced content. The mechanisms defined in [RFC6051] are RECOMMENDED to be adopted to reduce the possible synchronization delay.

5 Failure Cases

This section examines the implications of losing RTCP splicing notification message and the other failure case, e.g., the RTP header extension is stripped on the path.

Given that there may be a splicing un-aware middlebox on the path between the main RTP sender and the splicer, the main and the substitutive RTP senders can use one heuristic to verify whether or not the Splicing Interval reaches the splicer.

The splicer can be implemented to have its own SSRC, and send RTCP reception reports to the senders of the main and substitutive RTP streams. This allows the senders to detect problems on the path to the splicer. Alternatively, it is possible to implement the splicer such that it has no SSRC, and does not send RTCP reports; this prevents the senders from being able to monitor the quality to the path to the splicer.

If the splicer has an SSRC and sends its own RTCP reports, it can choose not to pass RTCP reports it receives from the receivers to the senders. This will stop the senders from being able to monitor the quality of the paths from the splicer to the receivers.

A splicer that has an SSRC can choose to pass RTCP reception reports from the receivers back to the senders, after modifications to account for the splicing. This will allow the senders the monitor the quality of the paths from the splicer to the receivers. A splicer that does not have its own SSRC has to forward and translation RTCP reports from the receiver, otherwise the senders will not see any receivers in the RTP session.

If the splicer is implemented following [RFC6828], it will have its own SSRC and will send its own RTCP reports, and will forward translated RTCP reports from the receivers.

Upon the detection of a failure, the splicer can communicate with the main sender and the substitutive sender in some out of band signaling ways to fall back to the payload specific mechanisms it supports, e.g., MPEG-TS splicing solution defined in [SCTE35], or just abandon the splicing.

6 Session Description Protocol (SDP) Signaling

This document defines the URI for declaring this header extension in an extmap attribute to be "urn:ietf:params:rtp-hdrext:splicing-interval".

This document extends the standard semantics defined in SDP Grouping Framework [RFC5888] with a new semantic: SPLICE to represent the relationship between the main RTP stream and the substitutive RTP stream. Only 2 m-lines are allowed in the SPLICE group. The main RTP stream is the one with the extended extmap attribute, and the other one is substitutive stream. A single m-line MUST NOT be included in different SPLICE groups at the same time. The main RTP sender provides the information about both main and substitutive sources.

The extended SDP attribute specified in this document is applicable for offer/answer content [RFC3264] and do not affect any rules when negotiating offer and answer. When used with multiple m-lines, substitutive RTP MUST be applied only to the RTP packets whose SDP m-line is in the same group with the substitutive stream using SPLICE and has the extended splicing extmap attribute. This semantic is also applicable for BUNDLE cases.

The following examples show how SDP signaling could be used for splicing in different cases.

6.1 Declarative SDP

```
v=0
o=xia 1122334455 1122334466 IN IP4 splicing.example.com
s=RTP Splicing Example
t=0 0
a=group:SPLICE 1 2
m=video 30000 RTP/AVP 100
i=Main RTP Stream
c=IN IP4 233.252.0.1/127
a=rtpmap:100 MP2T/90000
a=extmap:1 urn:ietf:params:rtp-hdrext:splicing-interval
a=mid:1
m=video 30002 RTP/AVP 100
i=Substitutive RTP Stream
c=IN IP4 233.252.0.2/127
a=sendonly
a=rtpmap:100 MP2T/90000
a=mid:2
```

Figure 3: Example SDP for a single-channel splicing scenario

The splicer receiving the SDP message above receives one MPEG2-TS stream (payload 100) from the main RTP sender (with multicast destination address of 233.252.0.1) on port 30000, and/or receives another MPEG2-TS stream from the substitutive RTP sender (with multicast destination address of 233.252.0.2) on port 30002. But at a particular point in time, the splicer only selects one stream and outputs the content from the chosen stream to the downstream receivers.

6.2 Offer/Answer without BUNDLE

SDP Offer - from main RTP sender

```
v=0
o=xia 1122334455 1122334466 IN IP4 splicing.example.com
s=RTP Splicing Example
t=0 0
a=group:SPLICE 1 2
m=video 30000 RTP/AVP 31 100
i=Main RTP Stream
c=IN IP4 splicing.example.com
a=rtpmap:31 H261/90000
a=rtpmap:100 MP2T/90000
a=extmap:1 urn:ietf:params:rtp-hdrext:splicing-interval
a=sendonly
a=mid:1
m=video 40000 RTP/AVP 31 100
i=Substitutive RTP Stream
c=IN IP4 substitutive.example.com
a=rtpmap:31 H261/90000
a=rtpmap:100 MP2T/90000
a=sendonly
a=mid:2
```

SDP Answer - from splicer

```
v=0
o=xia 1122334455 1122334466 IN IP4 splicer.example.com
s=RTP Splicing Example
t=0 0
a=group:SPLICE 1 2
m=video 30000 RTP/AVP 100
i=Main RTP Stream
c=IN IP4 splicer.example.com
a=rtpmap:100 MP2T/90000
a=extmap:1 urn:ietf:params:rtp-hdrext:splicing-interval
a=recvonly
a=mid:1
m=video 40000 RTP/AVP 100
i=Substitutive RTP Stream
c=IN IP4 splicer.example.com
a=rtpmap:100 MP2T/90000
a=recvonly
a=mid:2
```

6.3 Offer/Answer with BUNDLE: All Media are spliced

In this example, the bundled audio and video media have their own substitutive media for splicing:

1. An Offer, in which the offerer assigns a unique address and a substitutive media to each bundled "m="line for splicing within the BUNDLE group.

2. An answer, in which the answerer selects its own BUNDLE address, and leave the substitutive media untouched.

SDP Offer - from main RTP sender

```
v=0
o=alice 1122334455 1122334466 IN IP4 splicing.example.com
s=RTP Splicing Example
c=IN IP4 splicing.example.com
t=0 0
a=group:SPLICE foo 1
a=group:SPLICE bar 2
a=group:BUNDLE foo bar
m=audio 10000 RTP/AVP 0 8 97
a=mid:foo
b=AS:200
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 iLBC/8000
a=extmap:1 urn:ietf:params:rtp-hdrext:splicing-interval
a=sendonly
m=video 10002 RTP/AVP 31 32
a=mid:bar
b=AS:1000
a=rtpmap:31 H261/90000
a=rtpmap:32 MPV/90000
a=extmap:2 urn:ietf:params:rtp-hdrext:splicing-interval
a=sendonly
m=audio 20000 RTP/AVP 0 8 97
i=Substitutive audio RTP Stream
c=IN IP4 substitive.example.com
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 iLBC/8000
a=sendonly
a=mid:1
m=video 20002 RTP/AVP 31 32
i=Substitutive video RTP Stream
```

```
c=IN IP4 substitive.example.com
a=rtpmap:31 H261/90000
a=rtpmap:32 MPV/90000
a=mid:2
a=sendonly
```

SDP Answer - from the splicer

```
v=0
o=bob 2808844564 2808844564 IN IP4 splicer.example.com
s=RTP Splicing Example
c=IN IP4 splicer.example.com
t=0 0
a=group:SPLICE foo 1
a=group:SPLICE bar 2
a=group:BUNDLE foo bar
m=audio 30000 RTP/AVP 0
a=mid:foo
b=AS:200
a=rtpmap:0 PCMU/8000
a=extmap:1 urn:ietf:params:rtp-hdrext:splicing-interval
a=recvonly
m=video 30000 RTP/AVP 32
a=mid:bar
b=AS:1000
a=rtpmap:32 MPV/90000
a=extmap:2 urn:ietf:params:rtp-hdrext:splicing-interval
a=recvonly
m=audio 30002 RTP/AVP 0
i=Substitutive audio RTP Stream
c=IN IP4 splicer.example.com
a=rtpmap:0 PCMU/8000
a=recvonly
a=mid:1
m=video 30004 RTP/AVP 32
i=Substitutive video RTP Stream
c=IN IP4 splicer.example.com
a=rtpmap:32 MPV/90000
a=mid:2
a=recvonly
```

6.4 Offer/Answer with BUNDLE: a Subset of Media are Spliced

In this example, the substitutive media only applies for video when splicing:

1. An Offer, in which the offerer assigns a unique address to each bundled "m="line within the BUNDLE group, and assigns a substitutive

media to the bundled video "m=" line for splicing.

2. An answer, in which the answerer selects its own BUNDLE address, and leave the substitutive media untouched.

SDP Offer - from the main RTP sender:

```
v=0
o=alice 1122334455 1122334466 IN IP4 splicing.example.com
s=RTP Splicing Example
c=IN IP4 splicing.example.com
t=0 0
a=group:SPLICE bar 2
a=group:BUNDLE foo bar
m=audio 10000 RTP/AVP 0 8 97
a=mid:foo
b=AS:200
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 iLBC/8000
a=sendonly
m=video 10002 RTP/AVP 31 32
a=mid:bar
b=AS:1000
a=rtpmap:31 H261/90000
a=rtpmap:32 MPV/90000
a=extmap:2 urn:ietf:params:rtp-hdrext:splicing-interval
a=sendonly
m=video 20000 RTP/AVP 31 32
i=Substitutive video RTP Stream
c=IN IP4 substitutive.example.com
a=rtpmap:31 H261/90000
a=rtpmap:32 MPV/90000
a=mid:2
a=sendonly
```

SDP Answer - from the splicer:

```
v=0
o=bob 2808844564 2808844564 IN IP4 splicer.example.com
s=RTP Splicing Example
c=IN IP4 splicer.example.com
t=0 0
a=group:SPLICE bar 2
a=group:BUNDLE foo bar
m=audio 30000 RTP/AVP 0
a=mid:foo
b=AS:200
```

```
a=rtpmap:0 PCMU/8000
a=recvonly
m=video 30000 RTP/AVP 32
a=mid:bar
b=AS:1000
a=rtpmap:32 MPV/90000
a=extmap:2 urn:ietf:params:rtp-hdrext:splicing-interval
a=recvonly
m=video 30004 RTP/AVP 32
i=Substitutive video RTP Stream
c=IN IP4 splicer.example.com
a=rtpmap:32 MPV/90000
a=mid:2
a=recvonly
```

7 Security Considerations

The security considerations of the RTP specification [RFC3550] and the general mechanism for RTP header extensions [RFC5285] apply. The splicer can either be a mixer or a translator, and all the security considerations of these two RTP intermediaries topologies described in [RFC7667] and [RFC7201] are applicable for the splicer.

The splicer replaces some content with other content in RTP packet, thus breaking any RTP-level end-to-end security, such as source authentication and integrity protection. End to end source authentication is not possible with any known existing splicing solution. A new solution can theoretically be developed that enables identification of the participating entities and what each provides, i.e., the different media sources, main and substituting, and the splicer which provides the RTP-level integration of the media payloads in a common timeline and synchronization context.

Since the splicer breaks RTP-level end-to-end security, it needs to be part of the signaling context and the necessary security associations (e.g., SRTP crypto contexts) established for the RTP session participants. When using the Secure Real-Time Transport Protocol (SRTP) [RFC3711], the splicer would have to be provisioned with the same security association as the main RTP sender.

If there is a concern about the confidentiality of the splicing time information, the header extension defined in this document MUST be also protected, for example, header extension encryption [RFC6904] can be used in this case. However, the malicious endpoint may get the splicing time information by other means, e.g., inferring from the communication between the main and substitutive content sources. To avoid the insertion of invalid substitutive content, the splicer MUST have some mechanisms to authenticate the substitutive stream source.

For cases that the splicing time information is changed by a malicious endpoint, the splicing, for example, may fail since it will not be available at the right time for the substitutive media to arrive. Another case is that an attacker may prevent the receivers receiving the content from the main sender by inserting extra splicing time information. To avoid the above cases happening, the authentication of the RTP header extension for splicing time information SHOULD be considered.

When a splicer implemented as a mixer sends the stream to the receivers, CSRC list, which can be used to detect RTP-level forwarding loops as defined in Section 8.2 of [RFC3550], may be removed for simplifying the receivers that can not handle multiple sources in the RTP stream. Hence, loops may occur to cause packets to loop back to upstream of the splicer and may form a serious denial-of-service threat. In such a case, non-RTP means, e.g., signaling among all the participants, MUST be used to detect and resolve loops.

8 IANA Considerations

8.1 RTCP Control Packet Types

Based on the guidelines suggested in [RFC5226], a new RTCP packet format has been registered with the RTCP Control Packet Type (PT) Registry:

Name: SNM

Long name: Splicing Notification Message

Value: TBA

Reference: This document

8.2 RTP Compact Header Extensions

The IANA has also registered a new RTP Compact Header Extension [RFC5285], according to the following:

Extension URI: urn:ietf:params:rtp-hdrext:splicing-interval

Description: Splicing Interval

Contact: Jinwei Xia <xiajinwei@huawei.com>

Reference: This document

8.3 SDP Grouping Semantic Extension

This document request IANA to register the new SDP grouping semantic extension called "SPLICE".

Semantics: Splice

Token:SPLICE

Reference: This document

9 Acknowledgement

The authors would like to thank the following individuals who help to review this document and provide very valuable comments: Colin Perkins, Bo Burman, Stephen Botzko, Ben Campbell.

10 References

10.1 Normative References

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

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

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

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

[RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, June 2010.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, November 2010.

[RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, April 2014.

[RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, November 2015.

10.2 Informative References

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

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

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

[RFC6285] Ver Steeg, B., Begen, A., Van Caenegem, T., and Z. Vax, "Unicast-Based Rapid Acquisition of Multicast RTP Sessions", RFC 6285, June 2011.

[RFC6904] Lennox, J.,"Encryption of Header Extensions in the Secure Real-Time Transport Protocol (SRTP)", April 2013.

[SCTE35] Society of Cable Telecommunications Engineers (SCTE), "Digital Program Insertion Cueing Message for Cable", 2011.

[RFC6828] Xia, J., "Content Splicing for RTP Sessions", RFC 6828, January 2013.

Authors' Addresses

Jinwei Xia
Huawei

Email: xiajinwei@huawei.com

Roni Even
Huawei

Email: ron.even.tlv@gmail.com

Rachel Huang
Huawei

Email: rachel.huang@huawei.com

Lingli Deng
China Mobile

Email: denglingli@chinamobile.com