Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 26, 2012

M. Westerlund
B. Burman
P. Sandgren
Ericsson

October 24, 2011

RTCP SDES Item SRCNAME to Label Individual Sources
draft-westerlund-avtext-rtcp-sdes-srcname-00

Abstract

This document defines a new SDES item called SRCNAME which uniquely identifies a single media source, like a camera or a microphone. That way anyone receiving the SDES information from a set of interlinked RTP sessions can determine which SSRCs are related to the same source. It can equally be used to label SSRC multiplexed related streams, such as FEC or Retransmission streams related to the original source stream in the same session. In addition the new SDES item is also defined for usage with the SDP source specific media attribute ("a=ssrc") enabling an end-point to declare and learn the source bindings ahead of receiving RTP/RTCP packets through signalling.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 26, 2012.

Table of Contents

1. Introduction

RTP has always been a protocol that supports multiple participants each sending their own media streams in RTP sessions. Previously many implementations have aimed only at point to point voice over IP with a single source in each end-point. Even client implementations aimed at video conferences have often been built with the assumption around central mixers that only deliver a single media stream per media type. However, more advanced client implementations may transmit multiple streams in the same RTP session and there may be tight relations between different streams and their SSRCs. For example, a client with several cameras that uses simulcast to send streams with different encodings of the video from each camera have the need of conveying the relation of the streams to the receiver. A similar example is a client with several cameras that uses SVC multi-session transmission [RFC6190] and also here the receiver needs to know which streams relate to which video source. Other examples of tight relations are a retransmission stream and its original stream as well as the case of forward error correction (FEC) where a client can send source streams and associated repair streams.

CNAME is not sufficient to express this relation although that is commonly inferred from end-points that have only one media stream per media type. The primary use of CNAME in multi-source usages is instead to indicate which end-point and what synchronization context a particular media stream relates to and that usually means that all streams sent from a client have the same CNAME. We are neither relying on using the same SSRC for all streams related to a particular media source as it is not robust against SSRC collision and forces potentially cascading SSRC changes between sessions. Also, using the same SSRC is not possible when SSRC-multiplexing is used.

A common solution to convey the relation between streams is to use SDP attributes. Session-multiplexed streams can be associated with an attribute that groups different RTP sessions and SSRC-multiplexed streams can be grouped at the media level for each RTP session. For example, in Forward Error Correction Grouping Semantics in the Session Description Protocol [RFC5956] an SDP media level attribute called "ssrc-group:FEC-FR" is used for grouping FEC associations when the different streams from a source are SSRC-multiplexed in the same RTP session. Using SDP attributes may work fine in the case when the receivers of the streams also get an SDP describing the bindings of all the streams, but that is not always the case. One such example is a conference session where clients are communicating with each other via an RTP Translator. The RTP Translator forwards all RTP and RTCP traffic from a client to all other clients and the clients can be prepared to receive any number of streams of certain specified

media. When a new client joins the session the other clients may not be notified with a SIP Update including a new SDP, instead the clients will detect the new client's streams via RTP and RTCP. In this case there is no way for a client to identify if certain streams are related to each other since that information only was included in the SDP.

RTP Retransmission Payload Format [RFC4588] describes a solution for finding the association between original streams and retransmission streams when SSRC-multiplexing is used. The association can be resolved when the receiver receives a retransmission packet matching a retransmission request sent earlier. However, the RFC continues with describing that this mechanism might fail if there are two outstanding requests for the same packet sequence number in two different original streams of a session. Therefore, to avoid ambiguity in unicast a receiver MUST NOT have two outstanding requests for the same packet sequence number in two different original streams before the association is resolved. For multicast, however, this ambiguity cannot be avoided and SSRC-multiplexing of original and retransmission streams is therefore prohibited in multicast. By defining a solution for one to one mapping between an original stream and any supporting streams this issue can be avoided in the future. Note: This document does not update RFC 4588 to use this solution, but it may be done in the future.

To enable an RTP session participant to determine the close relation of different streams without the above mentioned problems, a new method for identifying such sources is needed. RTP [RFC3550] defines the Source Description RTCP Packet (SDES), which contains one or more chunks, each of which is composed of SDES items describing the SSRC identified in that chunk. None of the present SDES items is, however, suitable for uniquely identifying a media source.

Therefore we propose that one defines a new SDES item called the SRCNAME which with a unique label identifies a single media source, like a camera or a microphone. The source may also be a particular media mix or conceptual stream, such as the "most active speaker" output by a RTP mixer performing stream switching. That way anyone receiving the SDES information from a set of interlinked RTP sessions or multiple SSRCs in the same session can determine which SSRCs are the same source. Connecting streams with SRCNAME can be done irrespective of which multiplexing type is used and it solves the problems with the current solutions described above.

It is, however, possible that a receiver will receive the RTP streams before receiving SDES packets with all SRCNAME items and that would mean that the receiver cannot make the connections between SSRCs and SRCNAMEs when starting to receive the media. "Source-Specific Media

Attributes in the Session Description Protocol (SDP)" [RFC5576] defines a way of declaring different attributes for SSRCs in each session in SDP and if a new source attribute is added to this framework it would be suitable for conveying the connections between SSRCs and SRCNAMEs before the media communication starts. Thus, in addition to the new SDES item we also define a new SDP source-specific media attribute called srcname, which enables an end-point to declare and learn the source bindings ahead of receiving RTP/RTCP packets. Of course, this new SDP source attribute will not be useful for the case described above when clients did not get updates with new client's stream bindings, but it will be useful in most other cases.

2. 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].

3. SDES Item SRCNAME

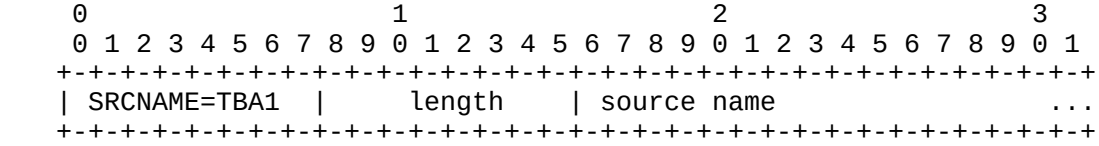
Source Descriptions are a method that should work with all RTP topologies (assuming that any intermediary node is supporting this item) and existing RTP extensions. Thus we propose that one defines a new SDES item called the SRCNAME which with a unique identifier identifies a single media source, like a camera, a microphone, a particular media mix, or conceptual stream. That way anyone receiving the SDES information from a set of interlinked RTP sessions or SSRCs in a single session can determine which SSRCs are related to the same source.

The SRCNAME is RECOMMENDED to be per communication session unique random identifiers generated according to "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)" [RFC6222] with the addition that a local counter enumerating the sources on the host also is concatenated to the key in step 4 prior to calculating the hash. The SRCNAME included in an RTCP packet MUST fulfill the requirements Section 6.5 in RTP [RFC3550] puts on SDES item values in general. These requirements is that it is a UTF-8 [RFC3629] string that have a maximum length of 255 bytes.

This SRCNAME's relation to CNAME is the following. CNAME represents an end-point and a synchronization context. If the different sources identified by SRCNAMEs should be played out synchronized when receiving them in a multi-stream context, then the sources need to be in the same synchronization context. Thus in all cases, all SSRCs

with the same SRCNAME will have the same CNAME. A given CNAME may contain multiple sets of sources using different SRCNAMEs.

The SDES SRCNAME item follows the same format as the other SDES items defined in RTP [RFC3550]:



When using the SRCNAME SDES item it is equally important to CNAME, thus it is RECOMMENDED to be included in all full compound RTCP packets being sent. It MAY also be included in non-compound packets in cases where the implementation believes that there might be new receivers needing the information.

4. SRCNAME in SDP

"Source-Specific Media Attributes in the Session Description Protocol (SDP)" [RFC5576] defines a way of declaring attributes for SSRC in each session in SDP. With a new SDES item, one can use this framework to define how also the SRCNAME can be provided in the SDP for each SSRC in each RTP session, thus enabling an end-point to declare and learn the source bindings ahead of receiving RTP/RTCP packets.

Hence, we propose a new SDP source attribute called srcname with the following structure:
a=ssrc:<ssrc-id> srcname:<srcname>

The srcname value MUST be identical to the SRCNAME value the media sender will send in the SDES SRCNAME item in the SDES RTCP packets.

FormalABNF syntax [RFC5234] for the "srcname" attribute:

```
srcname-attr = "srcname:" srcname

ssrcname = byte-string
   ; The definition of "byte-string" is in RFC 4566.

attribute =/ srcname-attr
   ; The definition of "attribute" is in RFC 4566.
```

5. Examples

This section shows SDP examples of declaring the SRCNAME in SDP. Only the relevant parts of the SDP are shown to improve readability. Please note that the below examples are all hypothetical as no decision has yet been to use the SRCNAME mechanism with the respective example.

5.1. Simulcast

In this use case the end-point is a client with a single audio source and two video sources and it uses simulcast for sending different encodings of the same video source. This example is based on Using Simulcast in RTP sessions [I-D.westerlund-avtcore-rtp-simulcast]. The following SDP describes this.

```
s=Simulcast enabled client
m=audio 49200 RTP/AVP 96
a=rtpmap:96 G719/48000/2
a=ssrc:521923924 cname:alice@foo.example.com
a=ssrc:521923924 srcname:2b:45:c7:12:83:e6
a=mid:1
m=video 49300 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42c01e
a=imageattr:* send [x=640,y=360] recv [x=640,y=360] [x=320,y=180]
a=ssrc:192392452 cname:alice@foo.example.com
a=ssrc:192392452 srcname:a3:d3:4b:f1:22:12
a=ssrc:834753488 cname:alice@foo.example.com
a=ssrc:834753488 srcname:7a:39:a9:3e:28:f7
a=mid:2
a=content:main
m=video 49400 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42c00d
a=imageattr:96 send [x=320,y=180]
a=ssrc:239245219 cname:alice@foo.example.com
a=ssrc:239245219 srcname:a3:d3:4b:f1:22:12
a=ssrc:734623563 cname:alice@foo.example.com
a=ssrc:734623563 srcname:7a:39:a9:3e:28:f7
a=mid:3
a=sendonly
```

The audio session is proposing to use one stereo stream of G.719 and the video sessions are proposing to send two different encodings of each video source, one with the resolution 640x360 and one with 320x180. The end-point also declares the SSRCs it intends to use with bindings to CNAME and SRCNAME, enabling the receiver of the SDP to bind together the video streams that originates from the same

video camera.

The use of the srcname attribute in the SDP is optional and the information can be retrieved from RTCP reporting, but it will then not be possible to correctly relate the video sources until the first RTCP report is received.

5.2. SVC with multi-session transmission

Here an example is shown of a client that uses SVC with multi-session transmission as described in RTP Payload Format for Scalable Video Coding [RFC6190]. RTP Payload Format for Scalable Video Coding [RFC6190] only describes examples for a client with one video source and the decoder dependencies of the different sessions are grouped using the Session grouping DDP attribute as defined in Signaling Media Decoding Dependency in the Session Description Protocol (SDP) [RFC5583] and implicitly CNAME.

However, if a client has two video sources and wishes to use multi-session transmission and send streams from both sources in each session an additional grouping mechanism is needed to group the different streams in the different sessions. SRCNAME is suitable for this and here we show an example where the DDP attribute groups the different sessions and the SRCNAME is used to relate the different SSRCs in each RTP session to one of the two video sources.

```
s=SVC MST client
a=group:DDP L1 L2 L3
m=video 20000 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=4de00a; packetization-mode=1;
 mst-mode=NI-TC; sprop-parameter-sets={sps0},{pps0};
a=ssrc:743947584 cname:bob@foo.example.com
a=ssrc:743947584 srcname:7e:83:c1:82:e8:a6
a=ssrc:283894947 cname:bob@foo.example.com
a=ssrc:283894947 srcname:b3:8d:f1:18:c5:84
a=mid:L1
m=video 20002 RTP/AVP 97
a=rtpmap:97 H264-SVC/90000
a=fmtp:97 profile-level-id=53000c; packetization-mode=1;
 mst-mode=NI-T; sprop-parameter-sets={sps1},{pps1};
a=ssrc:492784823 cname:bob@foo.example.com
a=ssrc:492784823 srcname:7e:83:c1:82:e8:a6
a=ssrc:892362397 cname:bob@foo.example.com
a=ssrc:892362397 srcname:b3:8d:f1:18:c5:84
a=mid:L2
a=depend:97 lay L1:96
m=video 20004 RTP/AVP 98
a=rtpmap:98 H264-SVC/90000
a=fmtp:98 profile-level-id=53001F; packetization-mode=1;
 mst-mode=NI-T; sprop-parameter-sets={sps2},{pps2};
a=ssrc:184562894 cname:bob@foo.example.com
a=ssrc:184562894 srcname:7e:83:c1:82:e8:a6
a=ssrc:305605682 cname:bob@foo.example.com
a=ssrc:305605682 srcname:b3:8d:f1:18:c5:84
a=mid:L3
a=depend:98 lay L1:96 L2:97
```

Thus, the client declares that it will send two video streams in each RTP session and the receiver is then able to relate the streams in the different sessions by using the SRCNAME binding. Without the SRCNAME binding it would not be possible for the receiver to know which streams belong to the same source.

5.3. Retransmission

This use case shows how SRCNAME can be used to connect retransmission streams to the original streams in the case of SSRC multiplexed RTP retransmission [RFC4588]. This is included to exemplify how RTP retransmission could be updated to provide explicit bindings between the source and the repair stream, but just an example and not a specification.

```
s=SSRC-multiplexed retransmission client
m=audio 49200 RTP/AVP 96
a=rtpmap:96 G719/48000/2
a=ssrc:521923924 cname:carol@foo.example.com
a=ssrc:521923924 srcname:88:3a:93:c1:3f:71
a=mid:1
m=video 49300 RTP/AVP 96 97
a=rtpmap:96 H264/90000
a=rtcp-fb:96 nack
a=fmtp:96 profile-level-id=42c01e
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96;rtx-time=200
a=ssrc:192392452 cname:carol@foo.example.com
a=ssrc:192392452 srcname:7b:6e:23:8b:31:a8
a=ssrc:834753488 cname:carol@foo.example.com
a=ssrc:834753488 srcname:7b:6e:23:8b:31:a8
a=ssrc:682394013 cname:carol@foo.example.com
a=ssrc:682394013 srcname:c4:98:d9:1a:fc:58
a=ssrc:284576129 cname:carol@foo.example.com
a=ssrc:284576129 srcname:c4:98:d9:1a:fc:58
a=mid:2
```

The client proposes to send two original video streams in the video session and a retransmission stream for each one of them. The retransmission streams are associated with the respective original stream by using the same SRCNAME and a receiver would then know which original stream a certain retransmission stream is associated with. This solves the ambiguity problem when SSRC-multiplexing is used for retransmission and it enables SSRC-multiplexing of original and retransmission streams to be used also in multicast sessions.

5.4. Forward Error Correction

Forward Error Correction Grouping Semantics in the Session Description Protocol [RFC5956] defines two SDP attributes for grouping the associated source and FEC-based repair streams. One can be used for grouping different RTP sessions and the other can be used for grouping SSRCs in the same RTP session, i.e. when session-respective SSRC-multiplexing is used. However, it may be advantageous to SSRC-multiplex the source streams in one RTP session and the repair streams in another since that gives a receiver the possibility to reject the repair session in case it does not support the proposed FEC. In this case the above mentioned grouping attributes cannot be used to associate the repair streams with the respective source stream since grouping of SSRCs cannot be made across RTP sessions. The following example shows how SRCNAME can be used for that.

```
s=FEC client
a=group:FEC-FR 1 2
m=video 49200 RTP/AVP 100
a=rtpmap:100 MP2T/90000
a=ssrc:847612849 cname:dave@foo.example.com
a=ssrc:847612849 srcname:45:a8:f4:19:b4:c3
a=ssrc:558237845 cname:dave@foo.example.com
a=ssrc:558237845 srcname:b8:58:29:c7:2f:9e
a=mid:1
m=application 49300 RTP/AVP 101
a=rtpmap:101 1d-interleaved-parityfec/90000
a=fmtp:101 L=5; D=10; repair-window=200000
a=ssrc:389572053 cname:dave@foo.example.com
a=ssrc:389572053 srcname:45:a8:f4:19:b4:c3
a=ssrc:185729479 cname:dave@foo.example.com
a=ssrc:185729479 srcname:b8:58:29:c7:2f:9e
a=mid:2
```

In this example the client proposes to send two video streams in one session and two repair streams in the other session. The repair streams are associated with the respective video stream by using the same SRCNAME. When receiving either this SDP or the SDES SRCNAME packets a receiver can make the connection between the source streams and the repair streams. Even a client not receiving the SDP will be able to do the association if it has established one RTP session for receiving source streams and another for receiving repair streams.

6. Usage with the Offer/Answer Model

The SDP offer/answer procedures for the a=ssrc is specified in Source-Specific Media Attributes in the Session Description Protocol (SDP) [RFC5576].

7. Backward Compatibility

Clients not supporting SRCNAME will not have the possibility to bind different streams to a specific media source, since they will not understand the SRCNAME SDES item. However, sending SRCNAME SDES items to a client not supporting it should not impose any problems since all clients should be prepared that new SDES items may be specified according to RTP [RFC3550].

According to the definition of SDP attributes in SDP: Session Description Protocol [RFC4566], if an attribute is received that is not understood, it MUST be ignored by the receiver. So a receiver

not supporting the ssrc attribute will simply ignore it.

Source-Specific Media Attributes in the Session Description Protocol (SDP) [RFC5576] defines rules of how new source attributes should be registered, which means that a receiver supporting RFC5576 should be prepared that new source attributes may be defined. This means that a user supporting some of the source attributes should not have any problems when the user receives an SDP with unknown source attributes.

8. IANA Considerations

Following the guidelines in SDP [RFC4566], in The Session Description Protocol (SDP) Grouping Framework [RFC5888], and in RTP [RFC3550], the IANA is requested to register:

1. A new SDES item named SRCNAME, as defined in Section 3. This item needs to be assigned an identifier TBA1.

2. A new SDP source attribute named srcname, as defined in Section 4.

9. Security Considerations

The SDES SRCNAMEs being opaque identifiers could potentially carry additional meanings or function as overt channel. If the SRCNAME would be permanent between sessions, they have the potential for compromising the users' privacy as they can be tracked between sessions. See Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs) [RFC6222] for more discussion.

A third party modification of the srcname labels either in the RTCP SDES items or in the SDP a=ssrc attribute can cause service disruption. By modifying labels the wrong streams could be associated, with potentially serious effects including media disruptions. If streams that are to be associated aren't associated, then another type of failures occur. To prevent modification, insertion or deletion of the srcname labels the carrying channel needs to be protected by integrity protection and source authentication. For RTCP various solutions exist, such as SRTP [RFC3711], DTLS [RFC4347], IPsec [RFC4301]. For protecting the SDP the signalling channel needs to provide protection. For SIP S/MIME [RFC3261] are the ideal, and hop by hopTLS [RFC5246] provides at least some protection, although not perfect. For SDP's retrieved using RTSP DESCRIBE [RFC2326] TLS would be the RECOMMENDED solution.

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.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

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

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

[RFC6222] Begen, A., Perkins, C., and D. Wing, "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)", RFC 6222, April 2011.

10.2. Informative References

[I-D.westerlund-avtcore-rtp-simulcast] Westerlund, M., Burman, B., Lindqvist, M., and F. Jansson, "Using Simulcast in RTP sessions", draft-westerlund-avtcore-rtp-simulcast (work in progress), October 2011.

[RFC2326] Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

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

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

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

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

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

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

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5583] Schierl, T. and S. Wenger, "Signaling Media Decoding Dependency in the Session Description Protocol (SDP)", RFC 5583, July 2009.

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

[RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in the Session Description Protocol", RFC 5956, September 2010.

[RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, May 2011.

Authors' Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

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

Bo Burman
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

Phone: +46 10 714 13 11
Email: bo.burman@ericsson.com

Patrik Sandgren
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

Phone: +46 10 717 97 41
Email: patrik.sandgren@ericsson.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 26, 2012

A. Akram
B. Burman
D. Grondal
M. Westerlund
Ericsson AB

October 24, 2011

RTP Media Stream Pause and Resume
draft-westerlund-avtext-rtp-stream-pause-00

Abstract

With the increased popularity of real-time multimedia applications, users demand more control over communication sessions. This document describes how a receiver in a multimedia conversation can pause and resume incoming data from a sender by sending real-time feedback messages when using Real-time Transport Protocol (RTP) for real time data transport. This document extends the Codec Control Messages (CCM) RTCP feedback package by adding a group of new real-time feedback messages used to pause and resume RTP data streams.

Table of Contents

1. Introduction

As real-time communication attracts more people, more applications are created; multimedia conversation applications being one example. Multimedia conversation further exists in many forms, for example, peer-to-peer chat application and multiparty video conferencing controlled by central media nodes, such as RTP Mixers.

Video conferencing MAY involve many participants; each has its own preferences and demands control over the communication session not only from the start but also during the session. This document describes several scenarios in multimedia communication where a participant chooses to temporarily pause incoming data from specific sources(s) and resuming it when needed. The receiver does not need to terminate the session from the source(s) and start all over again by negotiating the session parameters, for example using SIP [RFC3261] with SDP Offer/Answer [RFC3264].

Centralized nodes, like RTP Mixers, which either uses logic based on voice activity, other measurements, user input over properietary interfaces, or Media Stream Selection [I-D.westerlund-dispatch-stream-selection] could reduce the resources consumed in both the media sender and the network by temporarily pausing the media streams that aren’t required by the RTP Mixer. This becomes especially useful when the media sources are provided in multiple encoding versions (Simulcast) [I-D.westerlund-avtcore-rtp-simulcast] or with scalable encoding such as SVC [RFC6190]. There may be some of the defined encodings or combination of scalable layers that are not used all of the time.

As the the media streams required at any given point is highly dynamic, using the out-of-band signalling channel for pausing and even more importantly resuming a media stream is difficult due to the performance requirements. Instead, the pause and resume signalling should be in the media plane and go directly between the affected nodes. When using RTP [RFC3550] for media transport, using Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF) [RFC4585] appears approriate. No currently existing RTCP feedback message supports pausing and resuming an incoming data stream. As this is affects the generation of packets and may even allow the encoding process to be paused, the functionality appears to match Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF) [RFC5104] and should thus be defined as a Codec Control Message (CCM) extension.

2. Definition

2.1. Abbreviations

RTP Real-time Transport Protocol

RTCP Real-time Transport Control Protocol

SSRC Synchronization Source

CSRC Contributing Source

FB Feedback

AVPF Audio-Visual Profile with Feedback

FMT Feedback Message Type

PT Payload Type

CCM Codec Control Messages

MCU Multipoint Control Unit

2.2. Terminology

In addition to following, the definitions from RTP [RFC3550], AVPF [RFC4585] and CCM [RFC5104] also apply in this document.

Feedback Messages: CCM [RFC5104] categorised different RTCP feedback messages into four types, Request, Command, Indication and Notification. This document places the PAUSE and RESUME messages into Request category as they need acknowledgement.

Acknowledgement: The confirmation from receiver to sender that the message has been received.

Sender: The RTP entity that sends an RTP data stream.

Receiver: The RTP entity that receives an RTP data stream.

Mixer: The intermediate RTP node which receives a data stream from different nodes, combines them to make one stream and forwards to destinations, in the sense described in Topo-Mixer of RTP Topologies [RFC5117].

Participant: A member which is part of an RTP session, acting as receiver, sender or both.

Paused Sender: An RTP sender which receives a PAUSE request, defined in this memo, from all other members in a communication session and stops its transmission, i.e. no other participant receives its RTP transmission at any given time.

Pausing Receiver: An RTP receiver which sends a PAUSE request, defined in this memo, to other participant(s).

2.3. Requirements Language

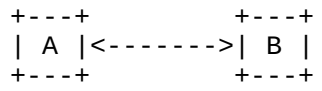
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Use Cases

This section discusses the main use cases for media stream pause and resume.

3.1. Point to Point

This is the most basic use case with an RTP session containing two end-points. Each end-point has one or more SSRCs.



Point to Point

The usage of media stream pause in this use case is to temporarily halt media delivery of media streams that the sender provides but the receiver doesn't currently use. This can for example de due to minimized applications where the video stream isn't actually shown on any display, and neither is it used in any other way, such as being recorded.

3.2. RTP Mixer to Media Sender

One of the most commonly used topologies in centralized conferencing is based on the RTP Mixer. The main reason for this is that it provides a very consistent view of the RTP session towards each participant. That is accomplished through the Mixer having its' own SSRCs and any media sent to the participants will be sent using those

SSRCs. If the Mixer wants to identify the underlying media sources for its' conceptual streams, it can identify them using CSRC. The media stream the Mixer provides can be an actual media mixing of multiple media sources, but it might also be as simple as selecting one of the underlying sources based on some Mixer policy or control signalling.

Figure 1: RTP Mixer

The media streams being delivered to a given receiver, A, can depend on several things. It can either be the RTP Mixer's own logic and measurements such as voice activity on the incoming audio streams. It can also be a human controlling the conference that determines how the media should be mixed; this would be more common in lecture or similar applications where regular listners may be prevented from breaking into the session unless approved by the moderator. The media selection could also be under the user's control using a protocol like Media Stream Selection [I-D.westerlund-dispatch-stream-selection].The media streams may also be simulcasted or scalably encoded, thus providing multiple versions that can be delivered by the media sender. These examples indicate that there are numerous reasons why a particular media stream would not currently be in use, but must be available for use at very short notice if any dynamic event occurs that causes a different media stream selection to be done in the Mixer.

Because of this, it would be highly beneficial if the Mixer could request to pause a particular media stream from being delivered to it. It also needs to be able to resume delivery with minimal delay.

3.3. Media Receiver to RTP mixer

An end-point like A in Figure 1 could potentially request to pause the delivery of a given media stream, like one of B's, over any of the SSRCs used by the Mixer by sending a pause request for the CSRC identifying the media stream. However, the authors are of the opinion that this is not a suitable solution.

First of all, the Mixer might not include CSRC in it's stream indications. Secondly, an end-point cannot rely on the CSRC to

correctly identify the media stream be paused when the delivered media is some type of mix. A media stream identification solution is needed to support this.

In addition, pause is only part of the semantics when it comes to selecting media streams. As can be seen in MESS [I-D.westerlund-dispatch-stream-selection], it can be beneficial to have both include and exclude semantics. In addition, substitution and possibility to control in what local media stream the selected media stream is to be provided gives richer functionality.

Due to the above reasons, we exclude this use case from consideration.

4. Design Considerations

This section describes the requirements that this memo needs to meet.

4.1. Real-time Nature

The first section (Section 1) of this memo describes some possible reasons why a receiver may pause an RTP sender. Pausing and resuming is time-dependent, i.e. a receiver may choose to pause an RTP stream for a certain duration after which the receiver may want the sender to resume. This time dependency means that the messages related to pause and resume must be transmitted to the sender in real-time in order for them to be purposeful.

4.2. Message Direction

It is the responsibility of a receiver, who wants to pause or resume a stream from the sender(s), to transmit PAUSE and RESUME messages. A sender who likes to pause itself, can simply do it.

4.3. Apply to Individual Sources

The PAUSE and RESUME messages apply to single media streams identified by their SSRC, which means the receiver targets the sender’s SSRC in the PAUSE and RESUME requests. If a paused sender starts sending with a new SSRC, the receivers will need to send a new PAUSE request in order to pause it.

4.4. Consensus

A sender must not pause an SSRC until all receivers that the sender knows of have requested it to be paused. The reason is that in RTP topologies where the media stream is shared between multiple

receivers, a single receiver on that shared network, independent of it being multicast or a transport Translator based, must not cause the media stream to be paused without the consent of all other receivers. A consequence of this is that a newly joining receiver needs to cause the sender to resume a paused stream. Any receiver wanting to resume a stream must also cause it to be resumed.

4.5. Acknowledgements

RTP does not guarantee reliable data transmission. It uses whatever assurance the lower layer transport protocol can provide. However, this is commonly UDP that provides no reliability guarantees. Thus it is possible that a PAUSE and/or RESUME message transmitted from an RTP end-point does not reach its destination, i.e. the targeted media sender. In some cases when a PAUSE or RESUME message reaches the media sender, it will not be able to pause the stream, instead the sender awaits requests from other receivers as well to fulfill the consensus requirement. In that case an RTP receiver MAY assume that previous PAUSE or RESUME message was lost and falsely retransmit it. In order to avoid this condition, the media sender target of a PAUSE or RESUME request needs to send an acknowledgement in response to each PAUSE and RESUME message.

4.6. Retransmitting Requests

As PAUSE or RESUME requests as well as Acknowledgments can be lost, the sender of a request will need to retransmit it in case no acknowlegement is received. The retransmission should take the round trip time into account, and will also need to take the normal RTCP bandwidth and timing rules applicable to the RTP session into account, when scheduling retransmission of feedback.

When it comes to resume requests that are more time critical, the best resume performance may be achieved by repeating the request as often as possible until a sufficient number have been sent to reach a high probability of request delivery, an acknowledgement has been received, or the media stream gets delivered.

4.7. Sequence Numbering

Every PAUSE and RESUME request message will need to have a sequence number to separate retransmissions from new requests. The sequence number is incremented by one every time a new request is transmitted. The PAUSE and RESUME message should share the same sequence number space. The advantage of using same sequence number space is to avoid the ambiguity which message to the request receiver should follow in case of retransmissions. For example, if an RTP sender receives both PAUSE and RESUME messages before deciding which message to respond to

(may be due to late packet arrival or any other reason), it can follow the message with higher sequence number.

Each acknowledgement will have the same sequence number as in the message (PAUSE or RESUME) it is responding to.

5. Solution Overview

The PAUSE and RESUME functionality is based on sending RTCP feedback messages from any RTP session participant that wants to pause or resume a media stream targeted at the media stream sender, as identified by the sender SSRC. A single Feedback message specification is used. The message consists of a number of Feedback Control Information (FCI) blocks, where each block can be a PAUSE request, a RESUME request or one of four different kinds of acknwoledgements. This structure allows a single feedback message to request pause or resume on a number of media streams.

To ensure reliability of the established state at the targeted media senders, acknowlegments are used. However, due to the requirement to not pause until all RTP session receivers, i.e. the ones that send RTCP Receiver Reports on the media sender's stream, are ok with it, most acknowlegements will NACK. This NACK says the session participant has established state for the media receiver that it desires a paused state, but it couldn't comply due to other session participants not having requested to pause the stream.

The transmission of any RTCP feedback messages follows the regular AVPF defined timing rules and depends on the session's mode of operation.

6. Participants States

This document introduces a new state the media stream in an RTP sender can have, a paused state.

6.1. Paused State

A media stream is in paused state when the sender pauses its transmission after receiving PAUSE requests from all other receiving participants in the session, which means no participant is willing to receive it's transmission. This requires the media stream sender to track all RTP session participants to determine that all have requested a pause state with the sender.

Following sub-sections discusses some potential issues when an RTP

sender goes into paused state.

6.1.1. RTCP BYE Message

When a participant leaves the communication session, it sends an RTCP BYE message. In addition to the semantics described in section 6.3.4 and 6.3.7 of RTP [RFC3550], following two conditions MUST also be considered when an RTP participant sends an RTCP BYE message,

- If a paused sender sends an RTCP BYE message, receivers observing this SHALL NOT send further PAUSE or RESUME requests to it.

- Since a sender pauses its transmission on receiving the PAUSE requests from all receivers in a session, the sender keeps record of all the receivers which do and which do not want to receive its transmission. If a pausing receiver sends an RTCP BYE message observed by the sender, the sender SHALL NOT consider that receiver when it decides to pause its transmission.

These conditions are also valid if an RTP Translator is used in the communication. When an RTP Mixer implementing this memo is involved between the participants (which forwards the stream by marking the RTP data with its own SSRC), it SHALL be a responsibility of the Mixer to control sending PAUSE and RESUME requests to the sender. The above conditions also apply to the sender and receiver parts of the RTP Mixer, respectively.

6.1.2. SSRC Time-out

Section 6.3.5 in RTP [RFC3550] describes the SSRC time-out of an RTP participant. Every RTP participant maintains a sender and receiver list in a session. If a participant does not get any RTP or RTCP packets from other participant(s) for last five RTCP reporting intervals it removes that participant from the receiver list.

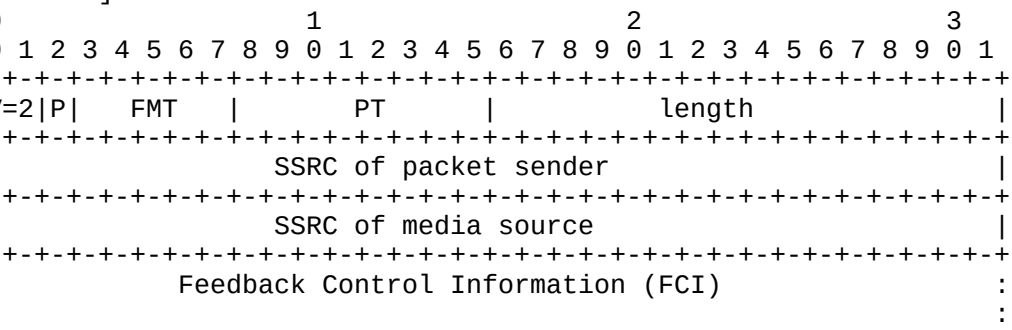
7. Message Format

Section 6 of AVPF [RFC4585] defines three types of low-delay RTCP feedback messages, i.e. Transport layer, Payload-specific, and Application layer feedback messages. This document defines a new Transport layer feedback message, this message is either a PAUSE request, a RESUME request, or one of four different types of acknowledgements in response to either PAUSE or RESUME requests.

The Transport layer feedback messages are identified by having the RTCP payload type be RTPFB (205) as defined by AVPF [RFC4585]. The PAUSE and RESUME messages are identified by Feedback Message Type

(FMT) value in common packet header for feedback message defined in section 6.1 of AVPF [RFC4585]. The PAUSE and RESUME transport feedback message is identified by the FMT value = TBA1.

The Common Packet Format for Feedback Messages is defined by AVPF [RFC4585] is:



For the PAUSE and RESUME messages, the following interpretation of the packet fields will be:

FMT: The FMT value identifying the PAUSE and RESUME message: TBA1

PT: Payload Type = 205 (RTPFB)

Length: As defined by AVPF, i.e. he length of this packet in 32-bit words minus one, including the header and any padding.

SSRC of packet sender: The SSRC of the RTP session participant sending the request(s) or acknowledgments in the FCI.

SSRC of media source: Not used, SHALL be set to 0. The FCI identifies the SSRC the request is for or whose request the acknowlegement are on.

The Feedback Control Information (FCI) field consist of one or more PAUSE, RESUME, or their acknowledgement messages, or any future extension. These messages have the following FCI format:

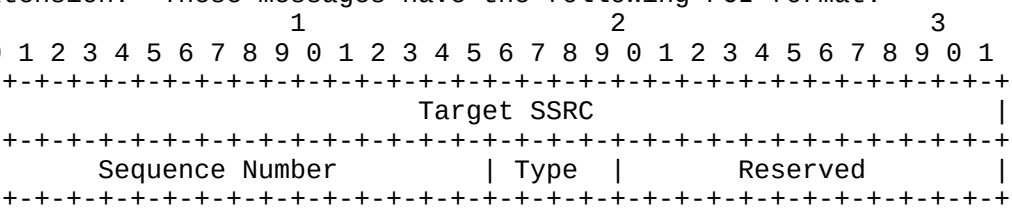


Figure 2: Syntax of FCI Entry in the PAUSE and RESUME message

The FCI fields have the following definitions:

Target SSRC (32 bits): For a Request message, the "Target SSRC" value is the SSRC that this request is intended for. For any type of Acknowlegement type defined in this document, the SSRC is the SSRC who sent the request being acknowledged. A CSRC MUST NOT be used as a target as the interpretation of such a request is unclear.

Sequence Number (16 bits): Sequence number of the request that SHALL be incremented by one for each new request. Both PAUSE and RESUME messages SHALL share the same sequence number space. Each requesting SSRC has its own sequence number space with each target SSRC. In other words, A requesting B to PAUSE or RESUME has a different sequence number space than A and C. Also, B requesting of A to PAUSE or RESUME will have a different sequence number space.

Type (4 bits): The pause feedback type, i.e. either PAUSE or RESUME or their acknowledgements. The values are as follows,

0: PAUSE message

1: RESUME message

2: Pause-Acknowledgement (PACK)

3: Resume-Acknowledgement (RACK)

4: Negative-Acknowledgement (NACK)

5: REFUSE

6-15: Reserved for future use

Reserved: (12 bits): SHALL be ignored by receivers implementing this memo and MUST be set to 0 by senders implementing this memo.

7.1. Message Acknowledgements

To let the sender of PAUSE and RESUME requests verify the reception and the target’s reaction to the request, the target of a PAUSE or RESUME request SHALL send an acknowledgment for each request received. All transmissions of request and acknowlegement are governed by the transmission rules as defined by Section 7.2. A request sender that hasn’t received any acknowledgement after one Round-Trip Time (RTT) MAY retransmit the request again.

After having received an acknowledgement on a request, a receiver SHOULD avoid sending further requests of the same type to the same sender to avoid unnecessary bandwidth consumption. However, a receiver MAY repeat a request of the same type, e.g. if it is for some reason necessary to re-confirm the sender's opinion of the receiver's request status. Consequently, a sender SHALL respond with corresponding acknowledgement to all requests, even if the request seems unnecessary and does not cause the sender to change state.

Every acknowledgement SHALL have the same sequence number as the request message (PAUSE or RESUME) it acknowledges. The sender can respond to PAUSE or RESUME requests in four different ways.

7.1.1. Negative-Acknowledgement (NACK)

In order for the sender to pause its transmission, it MUST receive PAUSE request from all the receivers in a session. Consider there are N receiving participants in a session. When a sender receives a PAUSE request, it MUST check if it has received requests from N-1 participants. If the number of requesting participants are less than N-1 it replies with NACK, which is the indication to the requester that though the request has been received, the transmission can not be paused at this stage because there are still some receiver(s) in the session that want to receive it. If a pausing receiver is no longer interested in pausing the SSRC, it MAY send an RESUME request to the sender from which it has previously received a NACK. The sender shall then reply with RACK to that receiver Section 7.1.3.

The NACK MUST only be sent in response to a PAUSE request. The NACK MUST have the same sequence number as in the PAUSE request.

7.1.2. Pause-Acknowledgement (PACK)

When an RTP sender receives a PAUSE request from all the receivers in a session, it sends a Pause-Acknowledgement (PACK) to the receivers and enters into Paused state as discussed in Section 6.1. It means that if there are N participants in a session and the sender receives PAUSE request(s) from N-1th participant, it pauses its transmission and sends a PACK to all the PAUSE requesters. The other participants can detect that the media sender is paused based on it sending a PACK.

The PACK MUST only be sent in response to a PAUSE request. The PACK MUST contain the same sequence number as in the PAUSE request.

7.1.3. Resume-Acknowledgement (RACK)

When an RTP sender receives a RESUME request from any of the receivers in a session, it replies with Resume-Acknowledgement (RACK) and resumes its transmission, if it is in Paused state (discussed in Section 6.1).

The RACK MUST only be sent in response to a RESUME request. The RACK MUST match the sequence number in RESUME request.

7.1.4. REFUSE

If any PAUSE and/or RESUME request can not be fulfilled by the sender due to some reason, it replies with REFUSE acknowledgement.

The REFUSE MAY be sent in response to PAUSE or RESUME requests. The REFUSE MUST contain the same sequence number as in the PAUSE/RESUME request.

7.2. Transmission Rules

To be Written

8. Examples

Following are the use cases when there MAY be a need to use PAUSE and RESUME messages,

1. Point-to-Point session
2. Point-to-multipoint using Mixer
3. Point-to-multipoint using Translator

8.1. Point-to-Point Session

This is the most basic scenario, which involves two participants, each acting as a sender and/or receiver. Any RTP data receiver sends PAUSE or RESUME message to the sender, which pauses or resumes transmission accordingly.

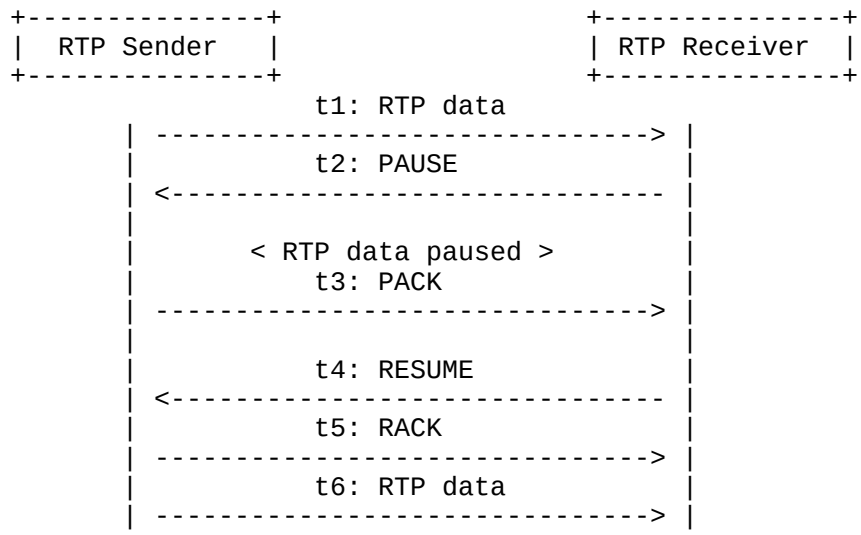


Figure 3: The pause and resume operation in Point-to-Point scenario

Figure 3 shows the basic pause and resume operation in Point-to-Point scenario. At time t1, an RTP sender sends data to a receiver. At time t2, the RTP receiver requests the sender to pause the stream. The sender pauses the data and replies with a Pause-Acknowledgement (PACK). Some time later (at time t4) the receiver requests the sender to resume, which resumes its transmission and replies with Resume-Acknowledgement (RACK).

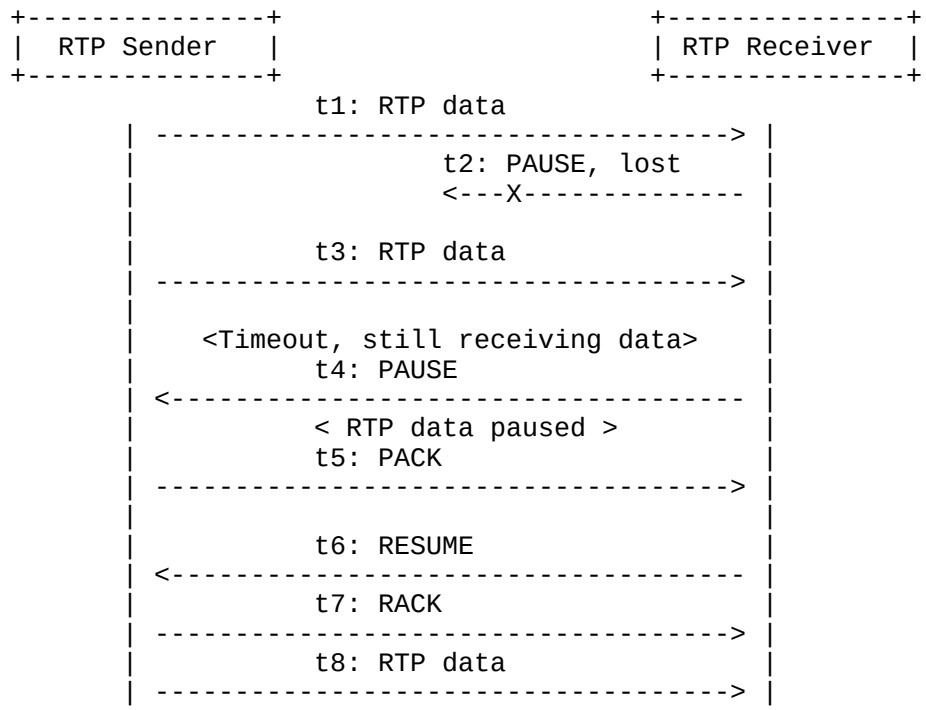


Figure 4: The pause and resume operation with PAUSE lost

Figure 4 describes what happens if a PAUSE message from an RTP receiver does not reach the RTP sender. After sending a PAUSE message, the receiver waits for a time-out to detect if the sender has paused the data transmission or has sent any acknowledgement according to the rules discussed in Section 7.1. As the PAUSE message is lost on the way (at time t2), RTP data continues to reach to the receiver. When the timer expires, receiver schedules retransmit of the PAUSE message. If PAUSE message reaches to the RTP sender, it stops streaming and replies with PACK. The same rules apply to the RESUME message, i.e., the RTP receiver waits for a time-out value after sending the RESUME message until it gets the transmission or receives any acknowledgement.

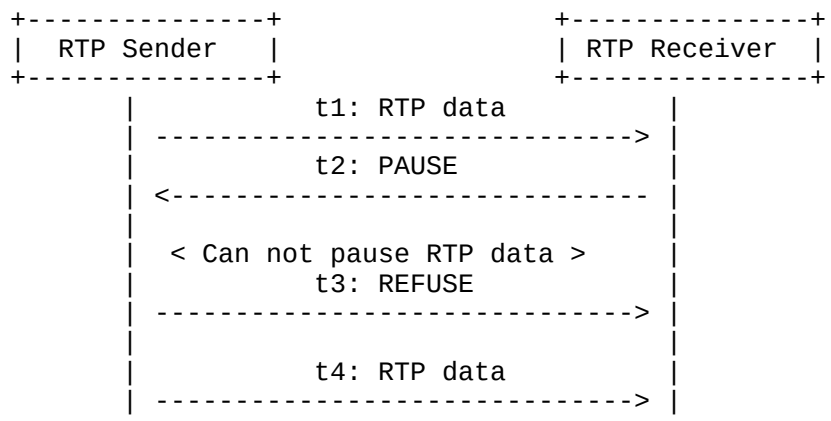


Figure 5: The pause request is refused in Point-to-Point scenario

In Figure 5, the receiver requests to pause the sender, which refuses to pause due to session policy and responds with REFUSE message.

8.2. Point-to-multipoint using Mixer

An RTP Mixer is an intermediate node connecting different transport-level clouds. The Mixer receives the streams from different RTP sources, selects or combines them based on the application's need and forwards the generated stream(s) to the destination. The Mixer puts its' own SSRC(s) in RTP data packets instead of the original source(s).

The Mixer keeps track of all the media streams delivered to the Mixer and how they currently are used. It selects the video stream to deliver to the receiver R based on the voice activity of the media senders. The video stream will be delivered to R using M's SSRC and with an CSRC indicating the orignal source.

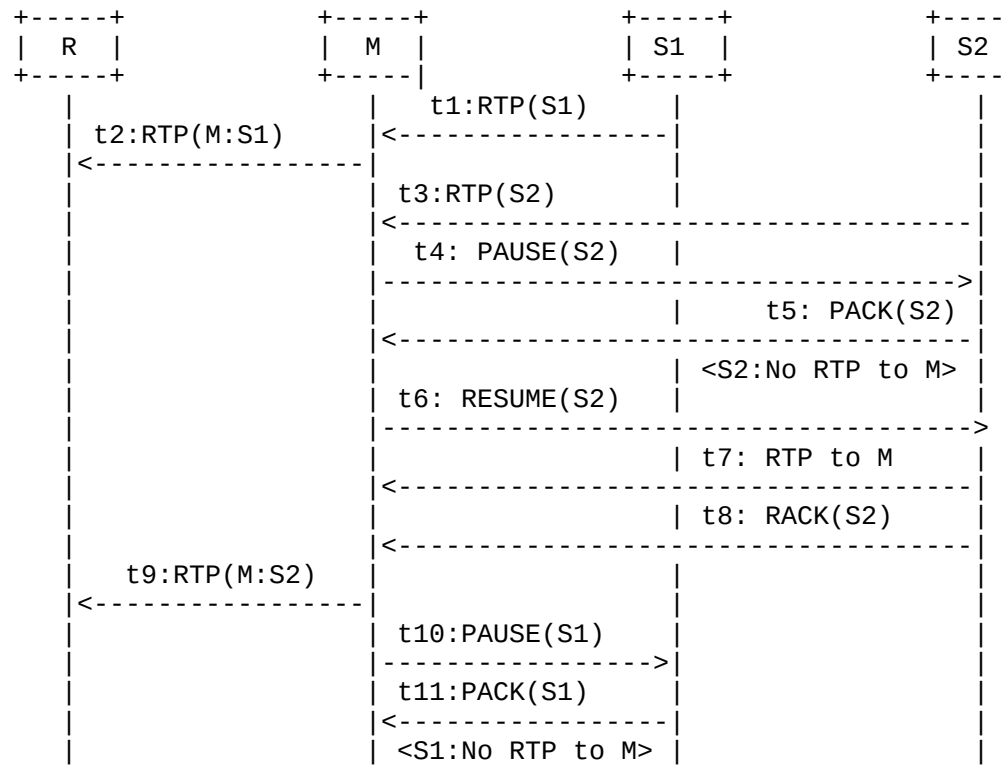


Figure 6: The pause and resume operations for an Voice Activated Mixer

The session starts at t1 with S1 being the most active speaker and thus being selected as the single video stream to be delivered to R (t2) using the Mixer SSRC but with the CSRC indicated after the colon in the figure. Then S2 joins the session at t3 and starts delivering media to the Mixer. As S2 has less voice activity then S1, the Mixer decides to pause S2 at t4 by sending S2 a PAUSE request. At t5, S2 acknowledges with a PACK and at the same instant stops delivering RTP to the Mixer. At t6, the user at S2 starts speaking and becomes the most active speaker and the Mixer decides to switch the video stream to S2, and therefore sends a RESUME request to S2. At t7, S2 has received the RESUME request and acts on it by resuming RTP media delivery to M. It also schedules the transmission of a RACK, which is sent at t8. When the media from t7 arrives at the Mixer, it switches this media into its SSRC (M) at t9 and changes the CSRC to S2. As S1 now becomes unused, the Mixer issues a PAUSE request to S1 at t10, which is acknowledged at t11 with a PACK and the RTP media stream from S1 stops being delivered.

8.3. Point-to-multipoint using Translator

A transport Translator in an RTP session forwards the message from one peer to all the others. Unlike Mixer, the Translator does not mix the streams and change the SSRC of the message. These examples are to show that the message can be safely used also in a transport Translator case.

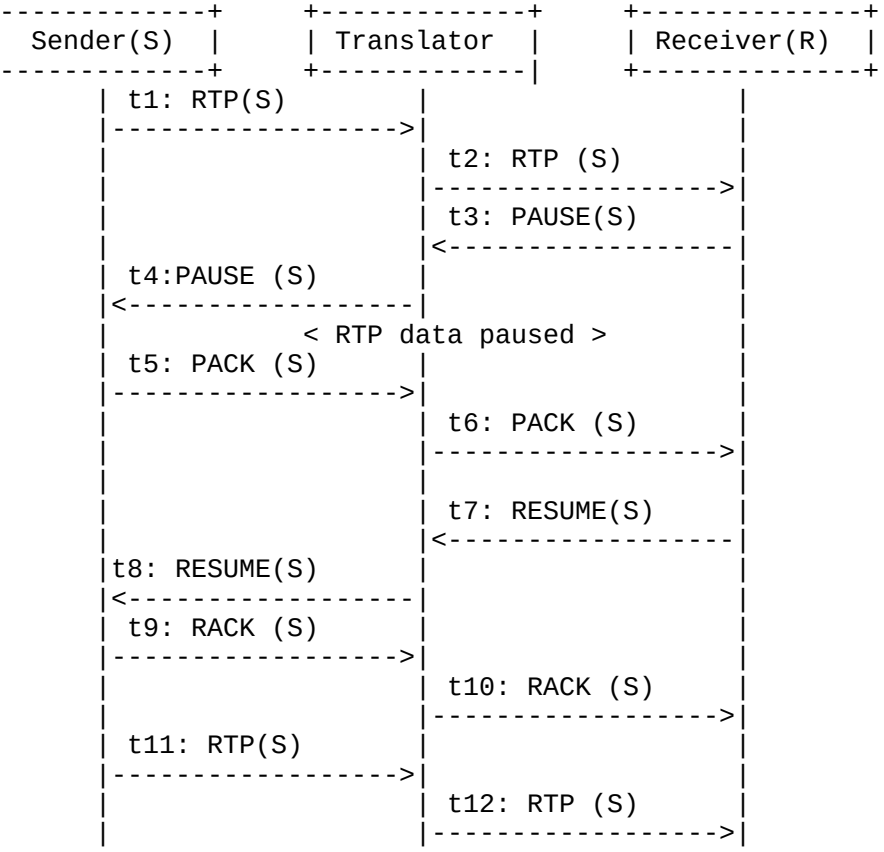


Figure 7: The pause and resume operation between two participants using the Translator

Figure 7 describes how a Translator can help the receiver in pausing and resuming the sender. The sender S sends RTP data to the receiver R through Translator, which just forwards the data without modifying the SSRCs. The receiver sends PAUSE requests to the sender, which checks that there is no other receiver which wants to receive the data, hence pauses itself and replies with PACK. Similarly the receiver resumes the sender by sending RESUME request through

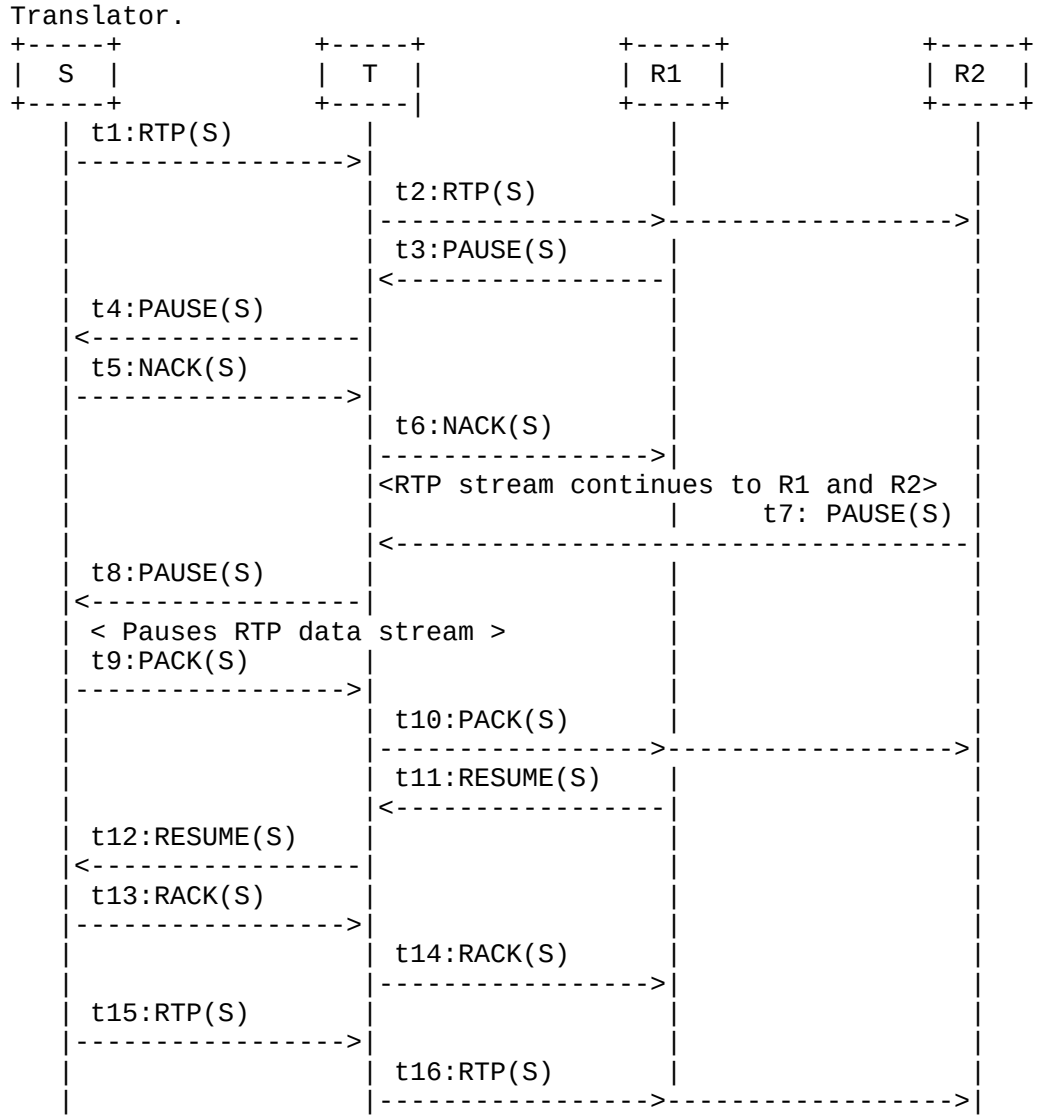


Figure 8: The pause and resume operation between one sender and two receivers through translator

Figure 8 explains the pause and resume operations when a transport Translator is involved between a sender and two receivers in an RTP session. Each message exchange is represented by the time it happens. At time t1, Sender (S) starts sending media to the Translator, which is forwarded to R1 and R2 through the Translator,

T. R1 and R2 receives RTP data from Translator at t2. At this point both R1 and R2 will send RTCP Receiver Reports to S informing that they receive S's media stream.

After some time (at t3), R1 chooses to pause the stream. On receiving the PAUSE request from R1, S checks if there are any other receiver which still wants to receive the data. At this time, S knows that R2 exists and has not indicated that it wants to pause the stream. The sender S replies with NACK to R1 and continues to send data to T which forwards to both R1 and R2. At t7, the receiver R2 also selects to pause the data by sending a PAUSE request. Now the sender S knows that no receiver (neither R1 nor R2) want the stream, it concludes that the stream must be paused. S now stops sending the stream and replies with PACK to R1 and R2. When any of the receivers (R1 or R2) choses to resume the stream from S, it sends a RESUME request to the sender. In reply, the RTP sender sends a RACK to the requesting RTP receiver and resumes streaming.

Consider an RTP session which includes one or more receivers, paused sender(s), and a Translator. A new participant joins the session, which is not aware of the paused sender(s). On receiving knowledge about the newly joined participant, e.g. any RTP traffic or RTCP report (i.e. either SR or RR) from the newly joined participant, the paused sender(s) resumes the transmission since there is now a receiver in the session that did not pause the sender. It SHALL depend on the new receiver to pause or continue that stream(s).

9. Signalling

The capability of handling PAUSE and RESUME messages MAY be exchanged at a higher layer such as SDP. This document extends the rtcp-fb attribute defined in section 4 of AVPF [RFC4585] to include the request for pause and resume. Like AVPF [RFC4585] and CCM [RFC5104], this document recommends to use the rtcp-fb attribute at media level and it must not be used at session level. This memo follows all the rules defined in AVPF for rtcp-fb attribute relating to payload type in a session description.

Section 7.1 of CCM [RFC5104] defines a new feedback value "ccm", which indicates the support of codec control using RTCP feedback. The CCM [RFC5104] defines four different parameters which SHOULD be used with the feedback value "ccm" to indicate the specific codec control command.

This memo defines a new parameter, "pause", which aggregatively represent the PAUSE, RESUME messages and their acknowledgements (i.e., PACK, NACK, RACK and REFUSE). An endpoint implementing this

memo and using SDP to signal capability MUST use the new "pause" extension to ccm signaling. Similarly, a sender or receiver SHOULD NOT use the messages from this memo towards receivers that did not declare capability for it.

The below figure is an example how to show support for pausing and resuming the stream according to this memo:

```
v=0
o=alice 3203093520 3203093520 IN IP4 host.example.com
s=Pausing Media
t=0 0
c=IN IP4 host.example.com
m=audio 49170 RTP/AVPF 98
a=rtpmap:98 H263-1998/90000
a=rtcp-fb:98 ccm pause
```

Figure 9: An SDP example with pause and resume capability

10. IANA Considerations

As outlined in Section 7, this memo requests IANA to allocate

1. The ’pause’ tag to be used with ccm under rtcp-fb AVPF attribute in SDP.

2. The FMT number TBA1 to be allocated to the PAUSE and RESUME functionality from this memo.

3. A registry listing registered values for ’pause’ Types.

4. PAUSE, RESUME, PACK, RACK, NACK, and REFUSE with the listed numbers in the pause Type registry.

11. Security Considerations

This document extends the CCM [RFC5104] and defines new messages, i.e., PAUSE and RESUME. The exchange of these new messages MAY have some security implications, which need to be addressed by the user. Following are some important implications,

1. Identity spoofing - An attacker can spoof him/herself as an authenticated user and can falsely pause or resume any source transmission. In order to prevent this type of attack, a strong authentication and integrity protection mechanism is needed.

2. Denial of Service (DoS) - An attacker can falsely paused all the source stream which MAY result in Denial of Service (DoS). An Authentication protocol MAY save from this attack.

3. Man-in-Middle Attack (MiMT) - The pausing and resuming of the RTP source is prone to a Man-in-Middle attack. The public key authentication May be used to prevent MiMT.

12. Acknowledgements

13. References

13.1. 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.

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

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

13.2. Informative References

[I-D.westerlund-avtcore-rtp-simulcast] Westerlund, M., Burman, B., Lindqvist, M., and F. Jansson, "Using Simulcast in RTP sessions", draft-westerlund-avtcore-rtp-simulcast (work in progress), October 2011.

[I-D.westerlund-dispatch-stream-selection] Grondal, D., Westerlund, M., and B. Burman, "Media Stream Selection (MESS)", draft-westerlund-dispatch-stream-selection-00 (work in progress), October 2011.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E.

Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

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

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

[RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, May 2011.

Authors' Addresses

Azam Akram
Ericsson AB
Farogatan 6
SE - 164 80 Kista,
Sweden

Phone: +46107142658
Fax: +46107175550
Email: muhammad.azam.akram@ericsson.com
URI: www.ericsson.com

Bo Burman
Ericsson AB
Farogatan 6
SE - 164 80 Kista,
Sweden

Phone: +46107141311
Fax: +46107175550
Email: bo.burman@ericsson.com
URI: www.ericsson.com

Daniel Grondal
Ericsson AB
Farogatan 6
SE - 164 80 Kista,
Sweden

Phone: +46107147505
Fax: +46107175550
Email: daniel.grondal@ericsson.com
URI: www.ericsson.com

Magnus Westerlund
Ericsson AB
Farogatan 6
SE- Kista 164 80,
Sweden

Phone: +46107148287
Fax:
Email: magnus.westerlund@ericsson.com
URI: www.ericsson.com

Audio/Video Transport Extensions
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2012

A. Williams
Audinate

October 31, 2011

IEEE 1588/802.1AS Synchronisation for RTP Streams
draft-williams-avtext-avbsync-02

Abstract

This memo specificies an RTP header extension for carrying in-band synchronization metadata provided by the IEEE1588/802.1AS Precision Time Protocols.

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 [1].

Status of this Memo

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

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

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

This Internet-Draft will expire on May 3, 2012.

Copyright Notice

Table of Contents

1. Outline

Alternative approach

Use existing header extension for timestamps. (RFC 6051)

Create additional small header extension for metadata (traceability, uncertainty, media clock discontinuity).

IDMS wants accurate timing too and has a method for describing clocks. Create a single specification for signalling clock domains which covers both use cases. IDMS can use that too.

2. Introduction

Synchronisation between RTP flows and between devices rendering RTP flows is currently facilitated by means of NTP format timestamps taken with respect to a shared reference clock. In many applications (e.g. professional, commercial and automotive AV), the NTP clock synchronisation protocol does not meet the necessary time alignment and synchronisation speed requirements.

Like NTP, the IEEE1588 Precision Time Protocol (PTP) family of clock synchronisation protocols provide a shared reference clock in an network - typically a LAN. IEEE1588 provides sub-microsecond synchronisation between devices on a LAN and typically locks within seconds at startup rather than minutes. With support from Ethernet switches, IEEE1588 protocols can achieve nanosecond timing accuracy in LANs. Network interface chips and cards supporting hardware time-stamping of timing critical protocol messages are also available.

When using IEEE1588 clock synchronisation, networked AV systems can achieve sub 1 microsecond time alignment accuracy when rendering AV signals and can support latencies less than 1ms through a gigabit LAN.

Three flavours of IEEE1588 are in use today:

- IEEE 1588-2002 [4]: the original "Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". This is often called IEEE1588v1 or PTPv1.

- IEEE 1588-2008 [5]: the second version of the "Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". This is a revised version of the original IEEE1588-2002 standard and is often called IEEE1588v2 or PTPv2.

- IEEE 802.1AS [6]: "Timing and Synchronization for Time Sensitive Applications in Bridged Local Area Networks". This is a Layer-2 only profile of IEEE 1588-2008 for use in Audio/Video Bridged LANs.

By using an IEEE 1588 derived reference clock, synchronisation of RTP streams and devices in LANs can be considerably improved.

3. Reference Clocks and Clock Domains

RTP uses NTP format timestamps to exchange information about media clock timing. NTP timestamps may come from a clock which is synchronised to a global time reference, but this is not assumed nor is there a standardised mechanism available to indicate that timestamps are derived from a common reference clock. Therefore, RTP implementations typically assume that NTP timestamps are taken using unsynchronised clocks and must compensate for absolute time differences and rate differences. Without a shared reference clock, RTP can time align flows from the same source at a given receiver using relative timing, however tight synchronisation between two or more different receivers (possibly with different network paths) or between two or more senders is not possible.

Each IEEE1588 clock is identified by a globally unique EUI-64 called a "ClockIdentity". A slave clock using one of the IEEE1588 family of network time protocols acquires the ClockIdentity/EUI-64 of the grandmaster clock that is the ultimate source of timing information. A master clock which is itself slaved to another master clock passes the grand master clock identity through to its slaves. The grandmaster clock identity can be used to identify a set of clocks in a single clock domain.

The IEEE1588 protocol family also carries an indication that the grand master clock provides traceable time. Traceablility of a time source implies that the clock is ultimately slaved to a global time source (e.g. GPS). A slave may consider one or more grandmaster clocks providing traceable time as belonging to the same (global) clock domain. Informally, traceabiliity trumps ClockIdentity when comparing clock domains. Traceable time is particularly applicable for wide area applications such as broadcast, transport and reflection seismology.

Several instances of the IEEE1588v1/v2 protocol may operate independently on a single network, forming distinct PTP network protocol domains each of which may have a different master clock. As the IEEE1588 standards have developed, the definition of PTP domains has changed. IEEE1588v1 identifies protocol subdomains by a textual

name and IEEE1588v2 identifies protocol domains using a numeric domain number. 802.1AS is a Layer2 profile of IEEE1588v2 supporting a single numeric clock domain (0). This specification assumes that an IEEE1588 clock master for multiple domains will provide the same timing information to all domains or that each clock domain has a different master. In other words, this specification assumes that a timing domain can be uniquely identified using the ClockIdentity of the grandmaster clock alone.

Accurate signal playout time alignment and accurate capture of input signal phase relationships are important requirements for A/V systems. In distributed AV systems containing many senders/receivers and differing network path lengths, a shared reference clock providing absolute time is needed. Relative timing using unsychronised clocks cannot provide the required level of time alignment (+/- 1us).

4. Timestamp formats

IEEE 1588/802.1AS timestamps use International Atomic Time (TAI) rather than UTC. Timestamps are 80 bits in total, divided into two parts:

PTP_sec: 48 bits seconds since epoch

PTP_nsec: 32 bits nanoseconds

A shorter 32 bit timestamp has also been defined for use in streaming media protocols in the following way:

as_timestamp = (PTP_sec * 10^9 + PTP_nsec) modulo 2^32

The shorter as_timestamp field covers a time interval slightly larger than 4 seconds.

5. IEEE 1733 / AVB RTCP Packet Type

IEEE 1733 [7] defines the "AVB RTCP packet" type reproduced in Figure 1. RTCP AVB packets contain a mapping between RTP timestamp and an 802.1AS timestamp as well as additional clock and QoS information.

The RTCP packet type shown below provides the corresponding IEEE1588 timestamp for an RTP timestamp in a similar fashion to an RTCP Sender Report (SR). In addition, the source of the timing information (i.e. the grandmaster ClockIdentity) is included. Clock domain information

allows an RTP implementation to determine whether sender and receiver timestamps have been taken using a shared reference clock. If the sender and receiver share a common clock domain, timestamps and be used and compared in an absolute sense.

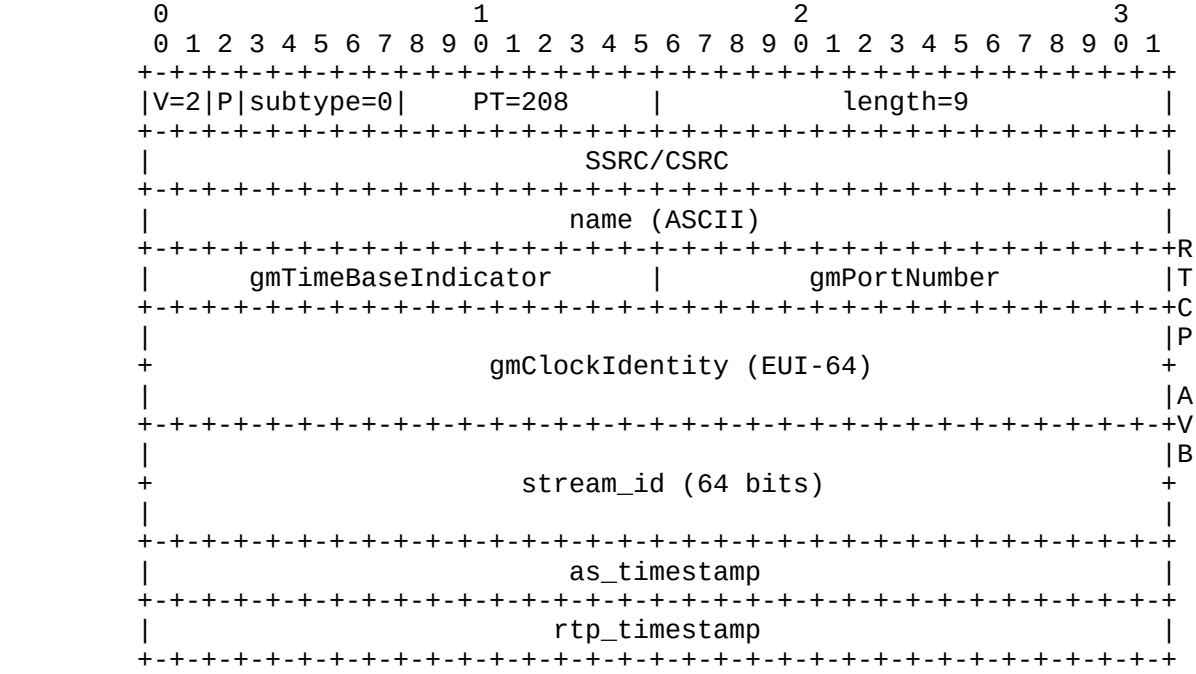


Figure 1: IEEE 1733 / AVB RTCP packet format

A brief description of the major fields follows:

name Reserved, must be set to zero and ignored on reception.

gmTimeBaseIndicator This field identifies the clock master time base. The gmTimeBaseIndicator increments each time a step change in time or frequency occurs. If the value of the gmTimeBaseIndicator and gmIdentity in the RTCP packet (i.e., of the sender) match the gmTimeBaseIndicator and gmIdentity at the receiver, the receiver is assured that timestamps have been taken using a shared reference clock. If they do not match, actions performed by the receiver are application dependent but may include entering a holdover mode until a positive match is again achieved.

gmPortNumber An integer identifying the port used on a grand master.

gmClockIdentity An EUI-64 identifying the grand master clock used by the source to generate as_timestamps for this flow.

stream_id A 64 bit number identifying the 802.1Qat [8] QoS reservation associated with this RTP flow.

as_timestamp The 32 bit 802.1AS timestamp (Section 4) associated with the RTP timestamp carried in this packet.

rtp_timestamp The RTP timestamp of a media packet.

Please consult the IEEE 1733 specification [7] for more details.

5.1. Observations

The RTCP packet type above provides the basic information for linking IEEE1588 time to RTP timestamps, however there are some additions which would improve the functionality provided.

The IEEE1733 specifcation does not define any SDP signalling. This means that the QoS parameters (ie the stream_id) cannot be signalled at call/flow setup time using SIP/RTSP. Whilst IEEE1733 provides clock domain information, it doesn't really define how clock domains work. Further, IEEE1733 does not include metadata to indicate clock changes. This specification addresses each of these issues.

5.2. RTCP Packet Subtypes

Systems not using the full AVB protocol suite can still benefit from timing improvements offered by IEEE 1588v1 and IEEE 1588v2. In general, the IEEE 1733 / AVB RTCP packet format (Figure 1) can be used to relate RTP timestamp instants in media packets to a reference clock provided by any of the IEEE 1588/PTP family of clock synchronisation protocols.

The subtype field indicates which IEEE 1588 protocol was used by the timestamping clock:

0x00 IEEE 802.1AS (defined by IEEE 1733)

0x01 IEEE 1588v1

0x02 IEEE 1588v2

6. Timing Header Extension

The header extension shown below provides a combination of media timestamps and timing metadata.

The provision of IEEE1588 timestamps in the RTP header is analogous to the existing NTP timestamp header extension supporting rapid synchonisation of RTP flows. High performance systems often handle RTP media packets in hardware and carriage of timing metadata in the media packets provides increased performance (e.g. faster lock times) and simplifies the system. In contrast to using RTCP, a header extension guarantees that timing metadata arrives at the receiver with the first media packet allowing them to be processed immediately. This approach facilitates fast switching between sources as is commonly used in zoned paging applications.

If the sender and receiver share a clock domain, source timestamps in media packets facilitate the collection of high quality information about the actual delays experienced by media packets through the network. Accurate delay information is important for diagnosing operational problems and for system optimisation.

The metadata included in the header allows senders to signal changes in clock disposition to receivers. Since IEEE1588 uses an election mechanism to determine the clock master it is possible for the master and/or grand master to change during the transmission of an RTP flow. Changing from one master clock to another may involve changes in clock frequency and/or phase. Additionally, the election protocol is not guaranteed to complete at the same time at all nodes in the clock domain, so there will be a transition period during which timestamps at the sender and receiver are not directly comparable. A sender indicates changes in IEEE1588 clocking by setting the timestamp uncertain (U) bit in the header. For example, the bit may be set when the best master clock election process is triggered due to loss of the current master clock. Once set, this bit should remain set during the entire period where PTP timing is in flux and for at least 5 seconds after PTP timing has stabilised (5 seconds allows as_timestamps which cover about 4 seconds to be disambiguated). When PTP timestamps are uncertain the receiver may refrain from updating the rtp_timestamp to wallclock mapping, effectively processing packets on the basis of rtp_timestamp alone.

In AV systems, media clocks are often provided via an external connection. When a media clock is disconnected or switched from one source to another (e.g. between two different SP/DIF ports), synchronisation may be lost and noise may be generated. A sender can indicate the loss of a media clock for a transmitted signal by toggling the media clock restart (M) bit. A receiver may use a

change in the media clock restart bit to mute audio or to hold a frame or to trigger resynchronisation to the new media clock.

In some systems, senders and receivers are on different networks with different grand master clocks however they may be part of a single global clock domain that uses traceable time. A sender can indicate that media packet timestamps have been taken with a traceable clock by setting T=1. Since it is possible for a PTP clock master to change during an RTP flow, the T bit may change. For example, if the traceable grand master clock used by the sender fails, PTP will elect a new grand master which may not be slaved to a global time source. In systems where traceable time is important, all candidate PTP masters should support traceable time and media signals timestamps should be clearly marked as traceable or not traceable. If timestamps are not traceable, T MUST be set to zero.

Figure 2 shows the fields of the AVB sync header extension. It uses the standard RTP header extension mechanism defined in RFC 5285 [2].

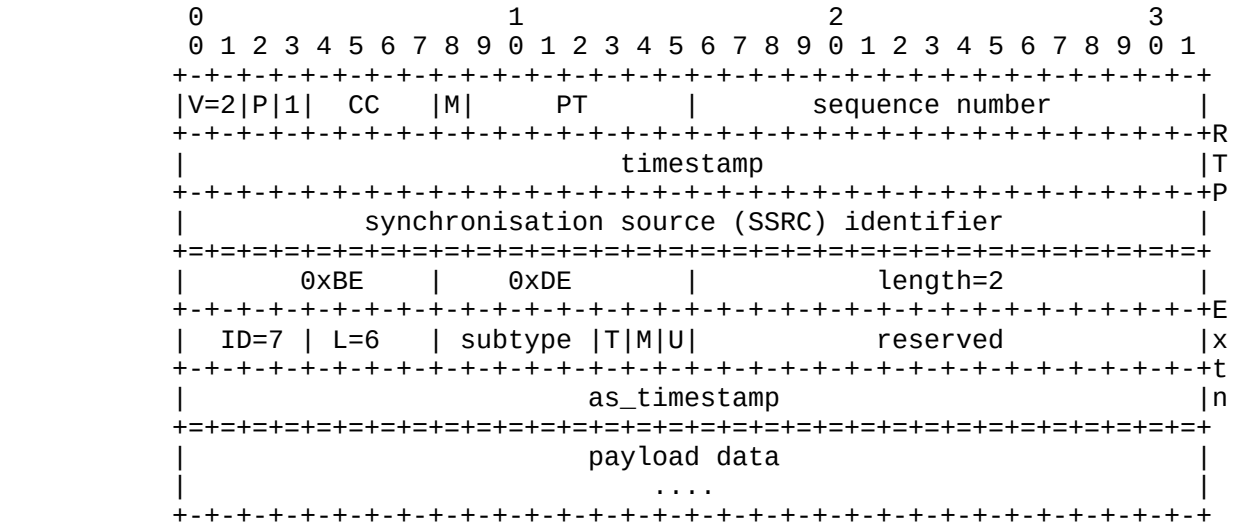


Figure 2: Timestamp Header Extension

The fields are defined as follows:

T Bit field indicating traceable timestamps. If T=1, the timestamp in this packet was taken by a clock slaved to a global time source otherwise T MUST be set to zero.

M Toggling the value of the media clock restart field (M) indicates a change in the source of the media clock. This kind of change need not be seamless but allows the receiver to take any appropriate action to minimize the disruption to the stream. The value of the M bit is toggled once by the sender each time the media clock source is changed. Toggling the bit rather than setting/resetting ensures media clock restarts will not be missed if it were to appear in a single RTP packet which happens to be dropped.

U The sender sets U=1 to indicate that timestamps may not be globally synchronized with network time. An example is the invocation of the best master clock algorithm because of a PTP SYNC timeout . Receivers may use the U bit, possibly in conjunction with their knowledge of the status of the PTP clock, to prevent unacceptable disturbances in the recovered media streams.

subtype See Section 5.2.

as_timestamp A 32 bit IEEE 1588/802.1AS timestamp as defined in Section 4.

reserved As this specification evolves, additional fields may be included in this header.

The as_timestamp MUST correspond to the same instant as the RTP timestamp in the packet's header, and MUST be derived from the same clock used to generate the as_timestamps in the RTCP AVB packets. Provided that it has knowledge of the SSRC to CNAME mapping, either from prior receipt of an RTCP CNAME packet or via out of band signalling such as RFC 5576 [3], the receiver can use the information provided as input to the synchronization algorithm, in exactly the same way as if an additional RTCP AVB packet had been received for the flow.

7. SDP signalling

The Session Description Protocol (SDP) allows clock domain and QoS information to be signalled via SIP or RTSP at call setup time. In the case of SIP, changes in information can also be signalled during the RTP session.

7.1. Clock domain

General format for signalling clock domains (todo: convert to EBNF):

a=clockdomain:ptp-version=PTP-PROTOCOL gmid=EUI-64 traceable=YES-OR-NO

Examples:
a=clockdomain:ptp-version=IEEE1588v1 gmid=39-A7-94-FF-FE-07-CB-D0 traceable=yes

a=clockdomain:ptp-version=IEEE1588v2 gmid=39-A7-94-FF-FE-07-CB-D0 traceable=yes

a=clockdomain:ptp-version=802.1AS gmid=39-A7-94-FF-FE-07-CB-D0 traceable=no

7.2. Quality of Service

General format for signalling AVB QoS to a receiver:
a=8021qat-qos:stream-id=EUI-64

Example:
a=8021qat-qos:stream-id=00-1D-C1-97-BB-3A-01-01

8. An Alternative Approach

Some RTP specifications already exists which cover aspects of the functionality described in this specification. Most notably, there is a specification for a header extension allowing NTP format timestamps to be included in RTP media packet headers (RFC 6051).

In addition, some new work (http://tools.ietf.org/html/draft-ietf-avtcore-idms-02) specifies a mechanism for describing clock sources including those based on IEEE 1588.

Rather developing new specifications based on the 32 bit as_timestamp format, an alternative approach which could fulfil the goals of this specification could be to:

1. Use the existing RFC 6051 to carry timestamps in RTP media packets (nothing to be done).

2. Create a new specification describing clock sources/domains and their signalling that is independent of both this specification and the IDMS specification. This would allow clock domains to be signalled generally in RTP. This might be done in avtcore.

3. Create a new specification augmenting RFC 6051 which carries the timing metadata (ptp uncertainty, media clock change,

traceability).

4. This specification: the header extension and clock domain signally go into other documents. There may be some remaining text describing how AVB systems in particular would use the above mechanisms.

9. IANA Considerations

TBD: A URN will be required to signal the presence of this header extension, such as:

urn:ietf:params:rtp-hdrext:avb-sync

10. Acknowledgements

The timing metadata (U and M) bits were inspired by the IEEE 1722 specification.

11. References

11.1. Normative References

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

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

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

11.2. Informative References

[4] Institute of Electrical and Electronics Engineers, "1588-2002 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2002, 2002, <http://standards.ieee.org/findstds/standard/1588-2002.html>.

[5] Institute of Electrical and Electronics Engineers, "1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008,

<http://standards.ieee.org/findstds/standard/1588-2008.html>.

[6] Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and Metropolitan Area Networks - IEEE Draft Standard for Local and Metropolitan Area Networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks", IEEE Std 802.1AS-2011, 2011, <http://standards.ieee.org/findstds/standard/802.1AS-2011.html>.

[7] Institute of Electrical and Electronics Engineers, "1733-2011 - IEEE Standard for Layer 3 Transport Protocol for Time-Sensitive Applications in Local Area Networks", IEEE Draft Std 1733/D7.0, February 2011, <http://standards.ieee.org/findstds/standard/1733-2011.html>.

[8] Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and Metropolitan Area Networks---Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)", IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005), 2010, <http://standards.ieee.org/about/get/>.

Appendix A. An Appendix

Author’s Address

Aidan Williams
Audinate
Level 1, 458 Wattle St
Ultimo, NSW 2007
Australia

Phone: +61 2 8090 1000
Fax: +61 2 8090 1001
Email: aidan.williams@audinate.com