AVTCORE WG
Internet-Draft
Updates: 3550, 3551 (if approved)
Intended status: Standards Track
Expires: January 08, 2016

M. Westerlund
Ericsson
C. Perkins
University of Glasgow
J. Lennox
Vidyo

July 07, 2015

Sending Multiple Types of Media in a Single RTP Session
draft-ietf-avtcore-multi-media-rtp-session-08

Abstract

This document specifies how an RTP session can contain RTP Streams with media from multiple media types such as audio, video, and text. This has been restricted by the RTP Specification, and thus this document updates RFC 3550 and RFC 3551 to enable this behaviour for applications that satisfy the applicability for using multiple media types in a single RTP session.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 3
3. Background and Motivation 3
4. Applicability 4
5. Using Multiple Media Types in a Single RTP Session 7
5.1. Allowing Multiple Media Types in an RTP Session 7
5.2. Demultiplexing Media Streams 8
5.3. Per-SSRC Media Type Restrictions 8
5.4. RTCP Considerations 9
6. Extension Considerations 9
6.1. RTP Retransmission Payload Format 9
6.2. RTP Payload Format for Generic FEC 11
6.3. RTP Payload Format for Redundant Audio 11
7. Signalling 12
8. Security Considerations 12
9. IANA Considerations 13
10. Acknowledgements 13
11. References 13
11.1. Normative References 13
11.2. Informative References 14
Authors' Addresses 15

1. Introduction

The Real-time Transport Protocol [RFC3550] was designed to use separate RTP sessions to transport different types of media. This implies that different transport layer flows are used for different media streams. For example, a video conferencing application might send audio and video traffic RTP flows on separate UDP ports. With increased use of network address/port translation, firewalls, and other middleboxes it is, however, becoming difficult to establish multiple transport layer flows between endpoints. Hence, there is pressure to reduce the number of concurrent transport flows used by RTP applications.

This memo updates [RFC3550] and [RFC3551] to allow multiple media types to be sent in a single RTP session in certain cases, thereby reducing the number of transport layer flows that are needed. It makes no changes to RTP behaviour when using multiple RTP streams containing media of the same type (e.g., multiple audio streams or multiple video streams) in a single RTP session, however

[I-D.ietf-avtcore-rtp-multi-stream] provides important clarifications to RTP behaviour in that case.

This memo is structured as follows. Section 2 defines terminology. Section 3 further describes the background to, and motivation for, this memo and Section 4 describes the scenarios where this memo is applicable. (tbd: fixme)

2. Terminology

The terms Encoded Stream, Endpoint, Media Source, RTP Session, and RTP Stream are used as defined in [I-D.ietf-avtext-rtp-grouping-taxonomy]. We also define the following terms:

Media Type: The general type of media data used by a real-time application. The media type corresponds to the value used in the <media> field of an SDP m= line. The media types defined at the time of this writing are "audio", "video", "text", "application", and "message".

Quality of Service (QoS): Network mechanisms that are intended to ensure that the packets within a flow or with a specific marking are transported with certain properties.

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

3. Background and Motivation

RTP was designed to support multimedia sessions, containing multiple types of media sent simultaneously, by using multiple transport layer flows. The existence of network address translators, firewalls, and other middleboxes complicates this, however, since a mechanism is needed to ensure that all the transport layer flows needed by the application can be established. This has three consequences:

1. increased delay to establish a complete session, since each of the transport layer flows needs to be negotiated and established;

2. increased state and resource consumption in the middleboxes, that can lead to unexpected behaviour when middlebox resource limits are reached; and

3. increased risk that a subset of the transport layer flows will fail to be established, thus preventing the application from communicating.

Using fewer transport layer flows can hence be seen to reduce the risk of communication failure, and can lead to improved reliability and performance.

One of the benefits of using multiple transport layer flows is that it makes it easy to use network layer quality of service (QoS) mechanisms to give differentiated performance for different flows. However, we note that many RTP-using application don't use network QoS features, and don't expect or desire any separation in network treatment of their media packets, independent of whether they are audio, video or text. When an application has no such desire, it doesn't need to provide a transport flow structure that simplifies flow based QoS.

Given this, it might seem desirable for RTP-based applications to send all their media streams bundled into one RTP session, that runs on a single transport layer flow. Unfortunately, this is prohibited by the RTP specification, since RTP makes certain assumptions that can be incompatible with sending multiple media types in a single RTP session. Specifically, the RTP control protocol (RTCP) timing rules assume that all RTP media flows in a single RTP session have broadly similar RTCP reporting and feedback requirements, which can be problematic when different types of media are multiplexed together. Certain RTP extensions also make assumptions that are incompatible with sending different media types in a single RTP session.

This memo updates [RFC3550] and [RFC3551] to allow RTP sessions to contain more than just one media type, and gives guidance on when it is safe to perform such multiplexing.

4. Applicability

This specification has limited applicability, and anyone intending to use it MUST ensure that their application and use meets the following criteria:

Equal treatment of media: The use of a single RTP session enforces similar treatment on all types of media used within the session. Applications that require significantly different network QoS or RTCP configuration for different media streams are better suited by sending those media streams on separate RTP session, using separate transport layer flows for each, since that gives greater flexibility. Further guidance is given in [I-D.ietf-avtcore-multiplex-guidelines] and [I-D.ietf-dart-dscp-rtp].

Compatible Media Requirements: The RTCP timing rules enforce a single RTCP reporting interval for all participants in an RTP

session. Flows with very different media requirements, for example a low-rate audio flow with no feedback needs and a high-quality video flow with different repair mechanisms, cannot be multiplexed together since this results in either excessive or insufficient RTCP for some flows, depending how the RTCP session bandwidth, and hence reporting interval, is configured.

Signalled Support: The extensions defined in this memo are not compatible with unmodified [RFC3550]-compatible endpoints. Their use requires signalling and mutual agreement by all participants within an RTP session. This requirement can be a problem for signalling solutions that can’t negotiate with all participants. For declarative signalling solutions, mandating that the session is using multiple media types in one RTP session can be a way of attempting to ensure that all participants in the RTP session follow the requirement. However, for signalling solutions that lack methods for enforcing that a receiver supports a specific feature, this can still cause issues.

Consistent support for multiple media types in a single RTP session: In multiparty communication scenarios it is important to separate two different cases. One case is where the RTP session contains multiple participants in a common RTP session. This occurs for example in Any Source Multicast (ASM) and Relay (Transport Translator) topologies as defined in RTP Topologies [I-D.ietf-avtcore-rtp-topologies-update]. It can also occur in some implementations of RTP mixers that share the same SSRC/CSRC space across all participants. The second case is when the RTP session is terminated in a middlebox and the other participants sources are projected or switched into each RTP session and rewritten on RTP header level including SSRC mappings.

For the first case, with a common RTP session or at least shared SSRC/CSRC values, all participants in multiparty communication are REQUIRED to support multiple media types in an RTP session. An participant using two or more RTP sessions towards a multiparty session can't be collapsed into a single session with multiple media types. The reason is that in case of multiple RTP sessions, the same SSRC value can be use in both RTP sessions without any issues, but when collapsed to a single session there is an SSRC collision. In addition some collisions can't be represented in the multiple separate RTP sessions. For example, in a session with audio and video, an SSRC value used for video will not show up in the Audio RTP session at the participant using multiple RTP sessions, and thus not trigger any collision handling. Thus any application using this type of RTP session structure MUST have a homogeneous support for multiple media types in one RTP session, or be forced to insert a translator node between that participant and the rest of the RTP session.

For the second case of separate RTP sessions for each multiparty participant and a central node it is possible to have a mix of single RTP session users and multiple RTP session users as long as one is willing to remap the SSRCs used by a participant with multiple RTP sessions into non-used values in the single RTP session SSRC space for each of the participants using a single RTP session with multiple media types. It can be noted that this type of implementation has to understand all types of RTP/RTCP extension being used in the RTP sessions to correctly be able to translate them between the RTP sessions. It might also suffer issues due to differencies in configured RTCP bandwidth and other parameters between the RTP sessions. It can also negatively impact the possibility for loop detection, as SSRC/CSRC can't be used to detect the loops, instead some other RTP stream or media source identity name space that is common across all interconnect parts are needed.

Ability to operate with limited payload type space: An RTP session has only a single 7-bit payload type space for all its payload type numbers. Some applications might find this space limiting when media different media types and RTP payload formats are using within a single RTP session.

Avoids incompatible Extensions: Some RTP and RTCP extensions rely on the existence of multiple RTP sessions and relate media streams between sessions. Others report on particular media types, and cannot be used with other media types. Applications that send multiple types of media into a single RTP session need to avoid such extensions.

5. Using Multiple Media Types in a Single RTP Session

This section defines what needs to be done or avoided to make an RTP session with multiple media types function without issues.

5.1. Allowing Multiple Media Types in an RTP Session

Section 5.2 of "RTP: A Transport Protocol for Real-Time Applications" [RFC3550] states:

> For example, in a teleconference composed of audio and video media encoded separately, each medium SHOULD be carried in a separate RTP session with its own destination transport address.
>
> Separate audio and video streams SHOULD NOT be carried in a single RTP session and demultiplexed based on the payload type or SSRC fields.

This specification changes both of these sentences. The first sentence is changed to:

> For example, in a teleconference composed of audio and video media encoded separately, each medium SHOULD be carried in a separate RTP session with its own destination transport address, unless specification [RFCXXXX] is followed and the application meets the applicability constraints.

The second sentence is changed to:

> Separate audio and video media sources SHOULD NOT be carried in a single RTP session, unless the guidelines specified in [RFCXXXX] are followed.

Second paragraph of Section 6 in RTP Profile for Audio and Video Conferences with Minimal Control [RFC3551] says:

> The payload types currently defined in this profile are assigned to exactly one of three categories or media types: audio only, video only and those combining audio and video. The media types are marked in Tables 4 and 5 as "A", "V" and "AV", respectively. Payload types of different media types SHALL NOT be interleaved or multiplexed within a single RTP session, but multiple RTP sessions MAY be used in parallel to send multiple media types. An RTP source MAY change payload types within the same media type during a session. See the section "Multiplexing RTP Sessions" of RFC 3550 for additional explanation.

This specifications purpose is to violate that existing SHALL NOT under certain conditions. Thus this sentence also has to be changed to allow for multiple media type's payload types in the same session. The above sentence is changed to:

> Payload types of different media types SHALL NOT be interleaved or multiplexed within a single RTP session unless as specified and under the restriction in Multiple Media Types in an RTP Session [RFCXXXX]. Multiple RTP sessions MAY be used in parallel to send multiple media types.

RFC-Editor Note: Please replace RFCXXXX with the RFC number of this specification when assigned.

5.2. Demultiplexing Media Streams

When receiving packets from a transport layer flow, an endpoint will first separate the RTP and RTCP packets from the non-RTP packets, and pass them to the RTP/RTCP protocol handler. The RTP and RTCP packets are then demultiplexed based on their SSRC into the different media streams. For each media stream, incoming RTCP packets are processed, and the RTP payload type is used to select the appropriate media decoder.

This process remains the same irrespective of whether multiple media types are sent in a single RTP session or not. It is important to note that the RTP payload type is never used to demultiplex media streams. Media streams are distinguished by SSRC, and the payload type is then used to route data for a particular SSRC to the right media decoder.

5.3. Per-SSRC Media Type Restrictions

An SSRC in an RTP session MUST NOT change media type during its lifetime. For example, an SSRC cannot start sending audio, then change to sending video. The lifetime of an SSRC ends when an RTCP BYE packet for that SSRC is sent, or when it ceases transmission for long enough that it times out for the other participants in the session.

The main motivation is that a given SSRC has its own RTP timestamp and sequence number spaces. The same way that you can't send two encoded streams of audio on the same SSRC, you can't send one encoded audio and one encoded video stream on the same SSRC. Each encoded stream when made into an RTP stream needs to have the sole control over the sequence number and timestamp space. If not, one would not be able to detect packet loss for that particular encoded stream. Nor can one easily determine which clock rate a particular SSRCs

timestamp will increase with. For additional arguments why RTP payload type based multiplexing of multiple media sources doesn't work see [I-D.ietf-avtcore-multiplex-guidelines].

Within an RTP session where multiple media types have been configured for use, an SSRC can only send one type of media during its lifetime (i.e., it can switch between different audio codecs, since those are both the same type of media, but cannot switch between audio and video). Different SSRCs MUST be used for the different media sources, the same way multiple media sources of the same media type already have to do. The payload type will inform a receiver which media type the SSRC is being used for. Thus the payload type MUST be unique across all of the payload configurations independent of media type that is used in the RTP session.

5.4. RTCP Considerations

When sending multiple types of media that have different rates in a single RTP session, endpoints MUST follow the guidelines for handling RTCP described in Section 7 of [I-D.ietf-avtcore-rtp-multi-stream].

6. Extension Considerations

This section outlines known issues and incompatibilities with RTP and RTCP extensions when multiple media types are used in a single RTP sessions. Future extensions to RTP and RTCP need to consider, and document, any potential incompatibility.

6.1. RTP Retransmission Payload Format

SSRC-multiplexed RTP retransmission [RFC4588] is actually very straightforward. Each retransmission RTP payload type is explicitly connected to an associated payload type. If retransmission is only to be used with a subset of all payload types, this is not a problem, as it will be evident from the retransmission payload types which payload types have retransmission enabled for them.

Session-multiplexed RTP retransmission is also possible to use where an retransmission session contains the retransmissions of the associated payload types in the source RTP session. The only difference to the previous case is if the source RTP session is one which contains multiple media types. This results in the retransmission streams in the RTP session for the retransmission having multiple associated media types.

When using SDP signalling for a multiple media type RTP session, i.e. BUNDLE [I-D.ietf-mmusic-sdp-bundle-negotiation], the session multiplexed case do require some recommendations on how to signal

this. To avoid breaking the semantics of the FID grouping as defined by [RFC5888] each media line can only be included in one FID group. FID is used by RTP retransmission to indicate the SDP media lines that is a source and retransmission pair. Thus, for SDP using BUNDLE, each original media source (m= line) that is retransmitted needs a corresponding media line in the retransmission RTP session. In case there are multiple media lines for retransmission, these media lines will form a independent BUNDLE group from the BUNDLE group with the source streams.

Below is an SDP example (Figure 1) which shows the grouping structures. This example is not legal SDP and only the most important attributes has been left in place. Note that this SDP is not an initial BUNDLE offer. As can be seen there are two bundle groups, one for the source RTP session and one for the retransmissions. Then each of the media sources are grouped with its retransmission flow using FID, resulting in three more groupings.

```
a=group:BUNDLE foo bar fiz
a=group:BUNDLE zoo kelp glo
a=group:FID foo zoo
a=group:FID bar kelp
a=group:FID fiz glo
m=audio 10000 RTP/AVP 0
a=mid:foo
a=rtpmap:0 PCMU/8000
m=video 10000 RTP/AVP 31
a=mid:bar
a=rtpmap:31 H261/90000
m=video 10000 RTP/AVP 31
a=mid:fiz
a=rtpmap:31 H261/90000
m=audio 40000 RTP/AVPF 99
a=rtpmap:99 rtx/90000
a=fmtp:99 apt=0;rtx-time=3000
a=mid:zoo
m=video 40000 RTP/AVPF 100
a=rtpmap:100 rtx/90000
a=fmtp:199 apt=31;rtx-time=3000
a=mid:kelp
m=video 40000 RTP/AVPF 100
a=rtpmap:100 rtx/90000
a=fmtp:199 apt=31;rtx-time=3000
a=mid:glo
```

Figure 1: SDP example of Session Multiplexed RTP Retransmission

6.2. RTP Payload Format for Generic FEC

The RTP Payload Format for Generic Forward Error Correction [RFC5109], and also its predecessor [RFC2733], requires some considerations, and they are different depending on what type of configuration of usage one has.

Independent RTP Sessions, i.e. where source and repair data are sent in different RTP sessions. As this mode of configuration requires different RTP session, there has to be at least one RTP session for source data, this session can be one using multiple media types. The repair session only needs one RTP Payload type indicating repair data, i.e. x/ulpfec or x/parityfec depending if RFC 5109 or RFC 2733 is used. The media type in this session is not relevant and can in theory be any of the defined ones. It is RECOMMENDED that one uses "Application".

If one uses SDP signalling with BUNDLE [I-D.ietf-mmusic-sdp-bundle-negotiation], then the RTP session carrying the FEC streams will be its own BUNDLE group. The media line with the source stream for the FEC and the FEC stream's media line will be grouped using media line grouping using the FEC or FEC-FR [RFC5956] grouping. This is very similar to the situation that arise for RTP retransmission with session multiplexing discussed above inSection 6.1.

The RTP Payload Format for Generic Forward Error Correction [RFC5109] and its predecessor [RFC2733] requires a separate RTP session unless the FEC data is carried in RTP Payload for Redundant Audio Data [RFC2198].

Note that the Source-Specific Media Attributes [RFC5576] specification defines an SDP syntax (the "FEC" semantic of the "ssrc-group" attribute) to signal FEC relationships between multiple RTP streams within a single RTP session. However, this can't be used as the FEC repair packets need to have the same SSRC value as the source packets being protected. [RFC5576] does not normatively update and resolve that restriction. There is ongoing work on an ULP extension to allow it be use FEC RTP streams within the same RTP Session as the source stream [I-D.lennox-payload-ulp-ssrc-mux].

6.3. RTP Payload Format for Redundant Audio

In stream, using RTP Payload for Redundant Audio Data [RFC2198] combining repair and source data in the same packets. This is possible to use within a single RTP session. However, the usage and configuration of the payload types can create an issue. First of all it might be necessary to have one payload type per media type for the

FEC repair data payload format, i.e. one for audio/ulpfec and one for text/ulpfec if audio and text are combined in an RTP session. Secondly each combination of source payload and its FEC repair data has to be an explicit configured payload type. This has potential for making the limitation of RTP payload types available into a real issue.

7. Signalling

Establishing an RTP session with multiple media types requires signalling. This signalling needs to fulfil the following requirements:

1. Ensure that any participant in the RTP session is aware that this is an RTP session with multiple media types.

2. Ensure that the payload types in use in the RTP session are using unique values, with no overlap between the media types.

3. Configure the RTP session level parameters, such as RTCP RR and RS bandwidth, AVPF trr-int, underlying transport, the RTCP extensions in use, and security parameters, commonly for the RTP session.

4. RTP and RTCP functions that can be bound to a particular media type SHOULD be reused when possible also for other media types, instead of having to be configured for multiple code-points. Note: In some cases one will not have a choice but to use multiple configurations.

The signalling of multiple media types in one RTP session in SDP is specified in "Multiplexing Negotiation Using Session Description Protocol (SDP) Port Numbers" [I-D.ietf-mmusic-sdp-bundle-negotiation].

8. Security Considerations

Having an RTP session with multiple media types doesn’t change the methods for securing a particular RTP session. One possible difference is that the different media have often had different security requirements. When combining multiple media types in one session, their security requirements also have to be combined by selecting the most demanding for each property. Thus having multiple media types can result in increased overhead for security for some media types to ensure that all requirements are meet.

Otherwise, the recommendations for how to configure and RTP session do not add any additional requirements compared to normal RTP, except

for the need to be able to ensure that the participants are aware that it is a multiple media type session. If not that is ensured it can cause issues in the RTP session for both the unaware and the aware one. Similar issues can also be produced in an normal RTP session by creating configurations for different end-points that doesn't match each other.

9. IANA Considerations

This memo makes no request of IANA.

10. Acknowledgements

The authors would like to thank Christer Holmberg, Gunnar Hellstroem, and Charles Eckel for the feedback on the document.

11. References

11.1. Normative References

[I-D.ietf-avtcore-rtp-multi-stream]
Lennox, J., Westerlund, M., Wu, W., and C. Perkins, "Sending Multiple Media Streams in a Single RTP Session", draft-ietf-avtcore-rtp-multi-stream-07 (work in progress), March 2015.

[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-22 (work in progress), June 2015.

[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.

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

11.2. Informative References

[I-D.ietf-avtcore-multiplex-guidelines]
Westerlund, M., Perkins, C., and H. Alvestrand, "Guidelines for using the Multiplexing Features of RTP to Support Multiple Media Streams", draft-ietf-avtcore-multiplex-guidelines-03 (work in progress), October 2014.

[I-D.ietf-avtcore-rtp-topologies-update]
Westerlund, M. and S. Wenger, "RTP Topologies", draft-ietf-avtcore-rtp-topologies-update-10 (work in progress), July 2015.

[I-D.ietf-avtext-rtp-grouping-taxonomy]
Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", draft-ietf-avtext-rtp-grouping-taxonomy-07 (work in progress), June 2015.

[I-D.ietf-dart-dscp-rtp]
Black, D. and P. Jones, "Differentiated Services (DiffServ) and Real-time Communication", draft-ietf-dart-dscp-rtp-10 (work in progress), November 2014.

[I-D.lennox-payload-ulp-ssrc-mux]
Lennox, J., "Supporting Source-Multiplexing of the Real-Time Transport Protocol (RTP) Payload for Generic Forward Error Correction", draft-lennox-payload-ulp-ssrc-mux-00 (work in progress), February 2013.

[I-D.westerlund-avtcore-transport-multiplexing]
Westerlund, M. and C. Perkins, "Multiplexing Multiple RTP Sessions onto a Single Lower-Layer Transport", draft-westerlund-avtcore-transport-multiplexing-07 (work in progress), October 2013.

[RFC2198] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V., Handley, M., Bolot, J., Vega-Garcia, A., and S. Fosse-Parisis, "RTP Payload for Redundant Audio Data", RFC 2198, September 1997.

[RFC2733] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", RFC 2733, December 1999.

[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.

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.

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

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, April 2010.

[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.

Authors' Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

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

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

Email: csp@csperkins.org

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

Email: jonathan@vidyo.com