Stream:	Internet Engineering Task Force (IETF)		
RFC:	8860		
Updates:	3550, 3551		
Category:	Standards Track		
Published:	September 2020		
ISSN:	2070-1721		
Authors:	M. Westerlund	C. Perkins	J. Lennox
	Ericsson	*University of Glasgow*	*8x8 / Jitsi*

RFC 8860
Sending Multiple Types of Media in a Single RTP Session

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 specifications (RFCs 3550 and 3551), and thus this document updates RFCs 3550 and 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

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8860.

Copyright Notice

Table of Contents

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 RTP 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, [RFC8108] 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; Section 4 describes the scenarios where this memo is applicable. Section 5 discusses issues arising from the base RTP and RTP Control Protocol (RTCP) specifications [RFC3550] [RFC3551] when using multiple types of media in a single RTP session, while Section 6 considers the impact of RTP extensions. We discuss signalling in Section 7. Finally, security considerations are discussed in Section 8.

2. Terminology

The terms "encoded stream", "endpoint", "media source", "RTP session", and "RTP stream" are used as defined in [RFC7656]. 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 a Session Description Protocol (SDP) "m=" line. The media types defined at the time of this writing are "audio", "video", "text", "image", "application", and "message" [RFC4566] [RFC6466].

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**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

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 QoS mechanisms to give differentiated performance for different flows. However, we note that many applications that use RTP 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 the above issues, it might seem appropriate for RTP-based applications to send all their RTP streams bundled into one RTP session, running over a single transport-layer flow. However, this is prohibited by the RTP specifications [RFC3550] [RFC3551], because the design of RTP makes certain assumptions that can be incompatible with sending multiple media types in a single RTP session. Specifically, the 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. Various RTP extensions also make assumptions about Synchronisation Source (SSRC) use and RTCP reporting 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 one media type in certain circumstances and gives guidance on when it is safe to send multiple media types in a single RTP session.

4. Applicability

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

Equal treatment of media: The use of a single RTP session normally results in similar network treatment for all types of media used within the session. Applications that require significantly different network QoS or RTCP configuration for different RTP streams are better suited to sending those RTP streams in separate RTP sessions, using separate transport-layer flows for each, since that method provides greater flexibility. Further guidance on how to provide differential treatment for some media streams is given in [RFC8872] and [RFC7657].

Compatible RTCP behaviour: The RTCP timing rules enforce a single RTCP reporting interval for all participants in an RTP session. Flows with very different media sending rates or RTCP feedback requirements cannot be multiplexed together, since this leads to either excessive or insufficient RTCP for some flows, depending on how the RTCP session bandwidth, and hence the reporting interval, are configured. For example, it is likely infeasible to find a single RTCP configuration that simultaneously suits both a low-rate audio flow with no feedback and a high-quality video flow with sophisticated RTCP-based feedback. Thus, combining these into a single RTP session is difficult and/or inadvisable.

Signalled support: The extensions defined in this memo are not compatible with unmodified endpoints that are compatible with [RFC3550]. 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 use 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 a requirement that a receiver support a specific feature, this can still cause issues.

Consistent support for multiparty RTP sessions: If it is desired to send multiple types of media in a multiparty RTP session, then all participants in that session need to support sending multiple types of media in a single RTP session. It is not possible, in the general case, to implement a gateway that can interconnect an endpoint that uses multiple types of media sent using separate RTP sessions with one or more endpoints that send multiple types of media in a single RTP session.

One reason for this is that the same SSRC value can safely be used for different streams in multiple RTP sessions, but when collapsed to a single RTP session there is an SSRC collision. This would not be an issue, since SSRC collision detection will resolve the conflict, except that some RTP payload formats and extensions use matching SSRCs to identify related flows and will break when a single RTP session is used.

A middlebox that remaps SSRC values when combining multiple RTP sessions into one also needs to be aware of all possible RTCP packet types that might be used, so that it can remap the SSRC values in those packets. This is impossible to do without restricting the set of RTCP packet types that can be used to those that are known by the middlebox. Such a middlebox might also have difficulty due to differences in configured RTCP bandwidth and other parameters between the RTP sessions.

Finally, the use of a middlebox that translates SSRC values can negatively impact the possibility of loop detection, as SSRC/CSRC (Contributing Source) can't be used to detect the loops; instead, some other RTP stream or media source identity namespace that is common across all interconnected parts is 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 to be limiting (i.e., overly restrictive) when using different media types and RTP payload formats within a single RTP session.

Avoidance of incompatible extensions: Some RTP and RTCP extensions rely on the existence of multiple RTP sessions and relate RTP 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 the guidelines specified in [RFC8860] are 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 [RFC8860] are followed.

The second paragraph of Section 6 of "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 specification's purpose is to override the above-listed "**SHALL NOT**" under certain conditions. Thus, this sentence also has to be changed to allow for multiple media types' payload types in the same session. The sentence containing "**SHALL NOT**" in the above paragraph is changed to:

> Payload types of different media types **SHALL NOT** be interleaved or multiplexed within a single RTP session unless [RFC8860] is used and the application conforms to the applicability constraints. Multiple RTP sessions **MAY** be used in parallel to send multiple media types.

5.2. Demultiplexing Media Types within an RTP Session

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 into the different RTP streams based on their SSRC. For each RTP 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.

As explained below, it is important to note that the RTP payload type is never used to distinguish RTP streams. The RTP packets are demultiplexed into RTP streams based on their SSRC; the RTP payload type is then used to select the correct media-decoding pathway for each RTP stream.

5.3. Per-SSRC Media Type Restrictions

An SSRC in an RTP session can change between media formats of the same type, subject to certain restrictions [RFC7160], but **MUST NOT** change its media type during its lifetime. For example, an SSRC can change between different audio formats, but it cannot start sending audio

and 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 with the same SSRC, you can't send one encoded audio and one encoded video stream with the same SSRC. Each encoded stream, when made into an RTP stream, needs to have 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 could one easily determine which clock rate a particular SSRC's timestamp will increase with. For additional arguments regarding why multiplexing of multiple media sources that is based on RTP payload type doesn't work, see [RFC8872].

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 it 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 the 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 as provided in Section 7 of [RFC8108].

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 session. Future extensions to RTP and RTCP need to consider, and document, any potential incompatibilities.

6.1. RTP Retransmission Payload Format

The RTP retransmission payload format [RFC4588] can operate in either SSRC-multiplexed mode or session-multiplexed mode.

In SSRC-multiplexed mode, retransmitted RTP packets are sent in the same RTP session as the original packets but use a different SSRC with the same RTCP Source Description (SDES) CNAME. If each endpoint sends only a single original RTP stream and a single retransmission RTP stream in the session, this is sufficient. If an endpoint sends multiple original and retransmission RTP streams, as would occur when sending multiple media types in a single RTP session, then each original RTP stream and the retransmission RTP stream have to be associated using heuristics. By having retransmission requests outstanding for only one SSRC not yet mapped, a receiver can determine the binding between the original and retransmission RTP streams. Another

alternative is the use of different RTP payload types, allowing the signalled "apt" (associated payload type) parameter [RFC4588] of the RTP retransmission payload format to be used to associate retransmitted and original packets.

Session-multiplexed mode sends the retransmission RTP stream in a separate RTP session to the original RTP stream, but using the same SSRC for each, with the association being done by matching SSRCs between the two sessions. This is unaffected by the use of multiple media types in a single RTP session, since each media type will be sent using a different SSRC in the original RTP session, and the same SSRCs can be used in the retransmission session, allowing the streams to be associated. This can be signalled using SDP with the BUNDLE grouping extension [RFC8843] and the Flow Identification (FID) grouping extension [RFC5888]. These SDP extensions require each "m=" line to only be included in a single FID group, but the RTP retransmission payload format uses FID groups to indicate the "m=" lines that form an original and retransmission pair. Accordingly, when using the BUNDLE extension to allow multiple media types to be sent in a single RTP session, each original media source ("m=" line) that is retransmitted needs a corresponding "m=" line in the retransmission RTP session. If there are multiple media lines for retransmission, these media lines will form an independent BUNDLE group from the BUNDLE group with the source streams.

An example SDP fragment showing the grouping structures is provided in Figure 1. This example is not legal SDP, and only the most important attributes have been left in place. Note that this SDP is not an initial BUNDLE offer. As can be seen in this example, there are two bundle groups -- one for the source RTP session and one for the retransmissions. Then, each of the media sources is 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 (FEC), as defined in [RFC5109] (and its predecessor, [RFC2733]), can either send the FEC stream as a separate RTP stream or send the FEC combined with the original RTP stream as a redundant encoding [RFC2198].

When sending FEC as a separate stream, the RTP payload format for generic FEC requires that FEC stream to be sent in a separate RTP session to the original stream, using the same SSRC, with the FEC stream being associated by matching the SSRC between sessions. The RTP session used for the original streams can include multiple RTP streams, and those RTP streams can use multiple media types. The repair session only needs one RTP payload type to indicate FEC data, irrespective of the number of FEC streams sent, since the SSRC is used to associate the FEC streams with the original streams. Hence, it is **RECOMMENDED** that the FEC stream use the "application/ulpfec" media type in the case of support for [RFC5109] and the "application/parityfec" media type in the case of support for [RFC2733]. It is legal, but **NOT RECOMMENDED**, to send FEC streams using media-specific payload format names (e.g., using both the "audio/ulpfec" and "video/ulpfec" payload formats for a single RTP session containing both audio and video flows), since this (1) unnecessarily uses up RTP payload type values and (2) adds no value for demultiplexing because there might be multiple streams of the same media type).

The combination of an original RTP session using multiple media types with an associated generic FEC session can be signalled using SDP with the BUNDLE extension [RFC8843]. In this case, the RTP session carrying the FEC streams will be its own BUNDLE group. The "m=" line for each original stream and the "m=" line for the corresponding FEC stream are grouped using the SDP Grouping Framework, using either the FEC-FR grouping [RFC5956] or, for backwards compatibility, the FEC grouping [RFC4756]. This is similar to the situation that arises for RTP retransmission with session-based multiplexing as discussed in Section 6.1.

The source-specific media attributes specification [RFC5576] defines an SDP extension (the "FEC" semantic of the "ssrc-group" attribute) to signal FEC relationships between multiple RTP streams within a single RTP session. This cannot be used with generic FEC, since the FEC repair packets need to have the same SSRC value as the source packets being protected. There existed a proposal (now abandoned) for an Uneven Level Protection (ULP) extension to enable transmission of the FEC RTP streams within the same RTP session as the source stream [FEC-Src-Multiplexing].

When the FEC is sent as a redundant encoding, the considerations in Section 6.3 apply.

6.3. RTP Payload Format for Redundant Audio

The RTP payload format for redundant audio [RFC2198] can be used to protect audio streams. It can also be used along with the generic FEC payload format to send original and repair data in the same RTP packets. Both are compatible with RTP sessions containing multiple media types.

This payload format requires each different redundant encoding to use a different RTP payload type number. When used with generic FEC in sessions that contain multiple media types, this requires each media type to use a different payload type for the FEC stream. For example, if audio and text are sent in a single RTP session with generic ULP FEC sent as a redundant encoding for each, then payload types need to be assigned for FEC using the audio/ulpfec and text/ulpfec payload formats. If multiple original payload types are used in the session, different redundant payload types need to be allocated for each one. This has potential to rapidly exhaust the available RTP payload type numbers.

7. Signalling

Establishing a single RTP session using multiple media types requires signalling. This signalling has to:

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. ensure that RTP session-level parameters -- for example, the RTCP RR and RS bandwidth modifiers [RFC3556], the RTP/AVPF trr-int parameter [RFC4585], transport protocol, RTCP extensions in use, and any security parameters -- are consistent across the session; and
4. ensure that RTP and RTCP functions that can be bound to a particular media type are reused where possible, rather than configuring multiple code points for the same thing.

When using SDP signalling, the BUNDLE extension [RFC8843] is used to signal RTP sessions containing multiple media types.

8. Security Considerations

RTP provides a range of strong security mechanisms that can be used to secure sessions [RFC7201] [RFC7202]. The majority of these are independent of the type of media sent in the RTP session; however, it is important to check that the security mechanism chosen is compatible with all types of media sent within the session.

Sending multiple media types in a single RTP session will generally require that all use the same security mechanism, whereas media sent using different RTP sessions can be secured in different ways. When different media types have different security requirements, it might be necessary to send them using separate RTP sessions to meet those different requirements. This can have significant costs in terms of resource usage, session setup time, etc.

9. IANA Considerations

This document has no IANA actions.

10. References

10.1. Normative References

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

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

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

[RFC8108] Lennox, J., Westerlund, M., Wu, Q., and C. Perkins, "Sending Multiple RTP Streams in a Single RTP Session", RFC 8108, DOI 10.17487/RFC8108, March 2017, <https://www.rfc-editor.org/info/rfc8108>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8843] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", RFC 8843, DOI 10.17487/RFC8843, September 2020, <https://www.rfc-editor.org/info/rfc8843>.

10.2. Informative References

[FEC-Src-Multiplexing] Lennox, J., "Supporting Source-Multiplexing of the Real-Time Transport Protocol (RTP) Payload for Generic Forward Error Correction", Work in Progress, Internet-Draft, draft-lennox-payload-ulp-ssrc-mux-00, 18 February 2013, <https://tools.ietf.org/html/draft-lennox-payload-ulp-ssrc-mux-00>.

[RFC2198] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V., Handley, M., Bolot, J.C., Vega-Garcia, A., and S. Fosse-Parisis, "RTP Payload for Redundant Audio Data", RFC 2198, DOI 10.17487/RFC2198, September 1997, <https://www.rfc-editor.org/info/rfc2198>.

[RFC2733] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", RFC 2733, DOI 10.17487/RFC2733, December 1999, <https://www.rfc-editor.org/info/rfc2733>.

[RFC3556] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, DOI 10.17487/RFC3556, July 2003, <https://www.rfc-editor.org/info/rfc3556>.

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

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <https://www.rfc-editor.org/info/rfc4585>.

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

[RFC4756] Li, A., "Forward Error Correction Grouping Semantics in Session Description Protocol", RFC 4756, DOI 10.17487/RFC4756, November 2006, <https://www.rfc-editor.org/info/rfc4756>.

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

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

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

[RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in the Session Description Protocol", RFC 5956, DOI 10.17487/RFC5956, September 2010, <https://www.rfc-editor.org/info/rfc5956>.

[RFC6466] Salgueiro, G., "IANA Registration of the 'image' Media Type for the Session Description Protocol (SDP)", RFC 6466, DOI 10.17487/RFC6466, December 2011, <https://www.rfc-editor.org/info/rfc6466>.

[RFC7160] Petit-Huguenin, M. and G. Zorn, Ed., "Support for Multiple Clock Rates in an RTP Session", RFC 7160, DOI 10.17487/RFC7160, April 2014, <https://www.rfc-editor.org/info/rfc7160>.

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

[RFC7202] Perkins, C. and M. Westerlund, "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution", RFC 7202, DOI 10.17487/RFC7202, April 2014, <https://www.rfc-editor.org/info/rfc7202>.

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

[RFC7657] Black, D., Ed. and P. Jones, "Differentiated Services (Diffserv) and Real-Time Communication", RFC 7657, DOI 10.17487/RFC7657, November 2015, <https://www.rfc-editor.org/info/rfc7657>.

[RFC8872] Westerlund, M., Burman, B., Perkins, C., Alvestrand, H., and R. Even, "Guidelines for Using the Multiplexing Features of RTP to Support Multiple Media Streams", RFC 8872, DOI 10.17487/RFC8872, September 2020, <https://www.rfc-editor.org/info/rfc8872>.

Acknowledgements

The authors would like to thank Christer Holmberg, Gunnar Hellström, Charles Eckel, Tolga Asveren, Warren Kumari, and Meral Shirazipour for their feedback on this document.

Authors' Addresses

Magnus Westerlund
Ericsson
Torshamnsgatan 23
SE-164 80 Stockholm
Sweden
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
8x8, Inc. / Jitsi
Jersey City, NJ 07302
United States of America
Email: jonathan.lennox@8x8.com