

AVTCORE Working Group  
INTERNET-DRAFT  
Updates: 7983, 5764  
Category: Standards Track  
Expires: September 5, 2020

B. Aboba  
Microsoft Corporation  
G. Salgueiro  
Cisco Systems  
C. Perkins  
University of Glasgow  
5 March 2020

Multiplexing Scheme Updates for QUIC  
draft-aboba-avtcore-rfc7983bis-00.txt

#### Abstract

This document defines how QUIC, Datagram Transport Layer Security (DTLS), Real-time Transport Protocol (RTP), RTP Control Protocol (RTCP), Session Traversal Utilities for NAT (STUN), Traversal Using Relays around NAT (TURN), and ZRTP packets are multiplexed on a single receiving socket.

This document updates RFC 7983 and RFC 5764.

#### Status of This Memo

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

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

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

This Internet-Draft will expire on September 5, 2020.

## Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	3
1.1. Terminology . . . . .	3
2. Multiplexing of TURN Channels . . . . .	3
3. Updates to RFC 7983 . . . . .	4
4. Security Considerations . . . . .	5
5. IANA Considerations . . . . .	6
6. References . . . . .	6
6.1. Normative References . . . . .	6
6.2. Informative References . . . . .	7
Acknowledgements . . . . .	7
Authors' Addresses . . . . .	8

## 1. Introduction

"Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)" [RFC7983] defines a scheme for a Real-time Transport Protocol (RTP) [RFC3550] receiver to demultiplex DTLS [RFC6347], Session Traversal Utilities for NAT (STUN) [RFC5389], Secure Real-time Transport Protocol (SRTP) / Secure Real-time Transport Control Protocol (SRTCP) [RFC3711], ZRTP [RFC6189] and TURN Channel packets arriving on a single port.

This document updates [RFC7983] and [RFC5764] to also allow QUIC [I-D.ietf-quic-transport] to be multiplexed on the same port. For peer-to-peer operation in WebRTC scenarios as described in [WEBRTC-QUIC][WEBRTC-QUIC-TRIAL], RTP is used to transport audio and video and QUIC is used for data exchange, SRTP [RFC3711] is keyed using DTLS-SRTP [RFC5764] and therefore SRTP/SRTCP [RFC3550], STUN, TURN, DTLS [RFC6347] and QUIC need to be multiplexed on the same port.

Since new versions of QUIC are allowed to change aspects of the wire image, there is no guarantee that future versions of QUIC beyond version 1 will adhere to the multiplexing scheme described in this document.

### 1.1. Terminology

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

## 2. Multiplexing of TURN Channels

TURN channels are an optimization where data packets are exchanged with a 4-byte prefix instead of the standard 36-byte STUN overhead (see Section 2.5 of [RFC5766]). [RFC7983] allocated the values from 64 to 79 in order to allow TURN channels to be demultiplexed when the TURN Client does the channel binding request in combination with the demultiplexing scheme described in [RFC7983].

As noted in [I-D.aboba-avtcore-quic-multiplexing], the first octet of a QUIC short header packet falls in the range 64 to 127, thereby overlapping with the allocated range for TURN channels of 64 to 79.

The first octet of QUIC long header packets fall in the range 192 to 255. Since QUIC long header packets precede QUIC short header packets, if no packets with a first octet in the range of 192 to 255 have been received, a packet whose first octet is in the range of 64 to 79 can be demultiplexed unambiguously as TURN Channel traffic.

Since WebRTC implementations supporting QUIC data exchange do not utilize TURN Channels, once packets with a first octet in the range of 192 to 255 have been received, a packet whose first octet is in the range of 64 to 127 can be demultiplexed as QUIC traffic.

### 3. Updates to RFC 7983

This document updates the text in Section 7 of [RFC7983] (which in turn updates [RFC5764]) as follows:

OLD TEXT

The process for demultiplexing a packet is as follows. The receiver looks at the first byte of the packet. If the value of this byte is in between 0 and 3 (inclusive), then the packet is STUN. If the value is between 16 and 19 (inclusive), then the packet is ZRTP. If the value is between 20 and 63 (inclusive), then the packet is DTLS. If the value is between 64 and 79 (inclusive), then the packet is TURN Channel. If the value is in between 128 and 191 (inclusive), then the packet is RTP (or RTCP, if both RTCP and RTP are being multiplexed over the same destination port). If the value does not match any known range, then the packet MUST be dropped and an alert MAY be logged. This process is summarized in Figure 3.

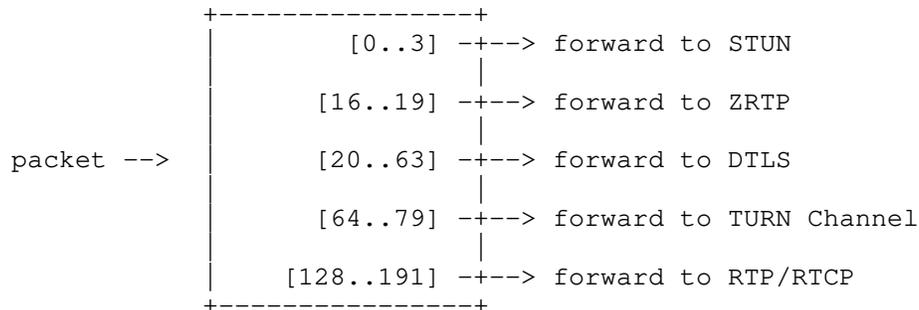


Figure 3: The DTLS-SRTP receiver's packet demultiplexing algorithm.

END OLD TEXT

NEW TEXT

The process for demultiplexing a packet is as follows. The receiver looks at the first byte of the packet. If the value of this byte is in between 0 and 3 (inclusive), then the packet is STUN. If the value is between 16 and 19 (inclusive), then the packet is ZRTP. If the value is between 20 and 63 (inclusive), then the packet is DTLS. If the value is in between 128 and 191 (inclusive) then the packet is

RTP (or RTCP, if both RTCP and RTP are being multiplexed over the same destination port). If the value is between 80 and 127 or between 192 and 255 (inclusive) then the packet is QUIC. If the value is between 64 and 79 inclusive, then if a packet has been previously forwarded that is in the range of 192 and 255, then the packet is QUIC, otherwise it is TURN Channel.

If the value does not match any known range, then the packet MUST be dropped and an alert MAY be logged. This process is summarized in Figure 3.

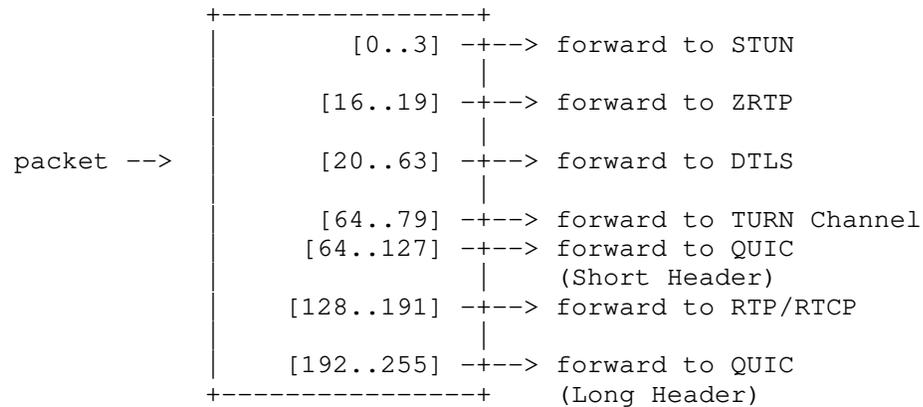


Figure 3: The receiver's packet demultiplexing algorithm.

END NEW TEXT

#### 4. Security Considerations

The solution discussed in this document could potentially introduce some additional security considerations beyond those detailed in [RFC7983].

Due to the additional logic required, if mis-implemented, heuristics have the potential to mis-classify packets.

When QUIC is used for only for data exchange, the TLS-within-QUIC exchange [I-D.ietf-quic-tls] derives keys used solely to protect the QUIC data packets. If properly implemented, this should not affect the transport of SRTP nor the derivation of SRTP keys via DTLS-SRTP, but if badly implemented, both transport and key derivation could be adversely impacted.

## 5. IANA Considerations

This document does not require actions by IANA.

## 6. References

### 6.1. Normative References

[I-D.ietf-quic-tls]

Thomson, M. and S. Turner, "Using Transport Layer Security (TLS) to Secure QUIC", draft-ietf-quic-tls-27 (work in progress), February 21, 2020.

[I-D.ietf-quic-transport]

Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-27 (work in progress), February 21, 2020.

[RFC2119]

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

[RFC3550]

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

[RFC3711]

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

[RFC5389]

Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <<http://www.rfc-editor.org/info/rfc5389>>.

[RFC5764]

McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <<http://www.rfc-editor.org/info/rfc5764>>.

[RFC5766]

Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, DOI 10.17487/RFC5766, April 2010, <<http://www.rfc->

[editor.org/info/rfc5766](http://editor.org/info/rfc5766)>.

- [RFC7983] Petit-Huguenin, M. and G. Salgueiro, "Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)", RFC 7983, DOI 10.17487/RFC7983, September 2016, <<http://www.rfc-editor.org/info/rfc7983>>.

## 6.2. Informative References

- [I-D.aboba-avtcore-quic-multiplexing] Aboba, B., Thatcher, P. and C. Perkins, "QUIC Multiplexing", draft-aboba-avtcore-quic-multiplexing-04 (work in progress), January 28, 2020.
- [RFC6189] Zimmermann, P., Johnston, A., Ed., and J. Callas, "ZRTP: Media Path Key Agreement for Unicast Secure RTP", RFC 6189, DOI 10.17487/RFC6189, April 2011, <<http://www.rfc-editor.org/info/rfc6189>>.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.
- [WEBRTC-QUIC] Thatcher, P. and B. Aboba, "QUIC API For Peer-to-Peer Connections", W3C Community Group Draft (work in progress), January 2020, <<https://w3c.github.io/webrtc-quic>>
- [WEBRTC-QUIC-TRIAL] Hampson, S., "RTCQuicTransport Coming to an Origin Trial Near You (Chrome 73)", January 2019, <<https://developers.google.com/web/updates/2019/01/rtcquictransport-api>>

## Acknowledgments

We would like to thank Martin Thomson, Roni Even and other participants in the IETF QUIC and AVTCORE working groups for their discussion of the QUIC multiplexing issue, and their input relating to potential solutions.

Authors' Addresses

Bernard Aboba  
Microsoft Corporation  
One Microsoft Way  
Redmond, WA 98052  
USA

Email: [bernard.aboba@gmail.com](mailto:bernard.aboba@gmail.com)

Gonzalo Salgueiro  
Cisco Systems  
7200-12 Kit Creek Road  
Research Triangle Park, NC 27709  
United States of America

Email: [gsalguei@cisco.com](mailto:gsalguei@cisco.com)

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

Email: [csp@csp Perkins.org](mailto:csp@csp Perkins.org)

AVTCORE Working Group  
INTERNET-DRAFT  
Updates: 7983, 5764  
Category: Standards Track  
Expires: May 19, 2021

B. Aboba  
Microsoft Corporation  
G. Salgueiro  
Cisco Systems  
C. Perkins  
University of Glasgow  
19 November 2020

Multiplexing Scheme Updates for QUIC  
draft-aboba-avtcore-rfc7983bis-01.txt

#### Abstract

This document defines how QUIC, Datagram Transport Layer Security (DTLS), Real-time Transport Protocol (RTP), RTP Control Protocol (RTCP), Session Traversal Utilities for NAT (STUN), Traversal Using Relays around NAT (TURN), and ZRTP packets are multiplexed on a single receiving socket.

This document updates RFC 7983 and RFC 5764.

#### 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 19, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction . . . . .	3
1.1. Terminology . . . . .	3
2. Multiplexing of TURN Channels . . . . .	3
3. Updates to RFC 7983 . . . . .	4
4. Security Considerations . . . . .	5
5. IANA Considerations . . . . .	6
6. References . . . . .	6
6.1. Normative References . . . . .	6
6.2. Informative References . . . . .	7
Acknowledgements . . . . .	7
Authors' Addresses . . . . .	8

## 1. Introduction

"Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)" [RFC7983] defines a scheme for a Real-time Transport Protocol (RTP) [RFC3550] receiver to demultiplex DTLS [RFC6347], Session Traversal Utilities for NAT (STUN) [RFC5389], Secure Real-time Transport Protocol (SRTP) / Secure Real-time Transport Control Protocol (SRTCP) [RFC3711], ZRTP [RFC6189] and TURN Channel packets arriving on a single port.

This document updates [RFC7983] and [RFC5764] to also allow QUIC [I-D.ietf-quic-transport] to be multiplexed on the same port. For peer-to-peer operation in WebRTC scenarios as described in [WEBRTC-QUIC][WEBRTC-QUIC-TRIAL], RTP is used to transport audio and video and QUIC is used for data exchange, SRTP [RFC3711] is keyed using DTLS-SRTP [RFC5764] and therefore SRTP/SRTCP [RFC3550], STUN, TURN, DTLS [RFC6347] and QUIC need to be multiplexed on the same port.

Since new versions of QUIC are allowed to change aspects of the wire image, there is no guarantee that future versions of QUIC beyond version 1 will adhere to the multiplexing scheme described in this document.

### 1.1. Terminology

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

## 2. Multiplexing of TURN Channels

TURN channels are an optimization where data packets are exchanged with a 4-byte prefix instead of the standard 36-byte STUN overhead (see Section 2.5 of [RFC5766]). [RFC7983] allocated the values from 64 to 79 in order to allow TURN channels to be demultiplexed when the TURN Client does the channel binding request in combination with the demultiplexing scheme described in [RFC7983].

As noted in [I-D.aboba-avtcore-quic-multiplexing], the first octet of a QUIC short header packet falls in the range 64 to 127, thereby overlapping with the allocated range for TURN channels of 64 to 79.

The first octet of QUIC long header packets fall in the range 192 to 255. Since QUIC long header packets precede QUIC short header packets, if no packets with a first octet in the range of 192 to 255 have been received, a packet whose first octet is in the range of 64 to 79 can be demultiplexed unambiguously as TURN Channel traffic.

Since WebRTC implementations supporting QUIC data exchange do not utilize TURN Channels, once packets with a first octet in the range of 192 to 255 have been received, a packet whose first octet is in the range of 64 to 127 can be demultiplexed as QUIC traffic.

3. Updates to RFC 7983

This document updates the text in Section 7 of [RFC7983] (which in turn updates [RFC5764]) as follows:

OLD TEXT

The process for demultiplexing a packet is as follows. The receiver looks at the first byte of the packet. If the value of this byte is in between 0 and 3 (inclusive), then the packet is STUN. If the value is between 16 and 19 (inclusive), then the packet is ZRTP. If the value is between 20 and 63 (inclusive), then the packet is DTLS. If the value is between 64 and 79 (inclusive), then the packet is TURN Channel. If the value is in between 128 and 191 (inclusive), then the packet is RTP (or RTCP, if both RTCP and RTP are being multiplexed over the same destination port). If the value does not match any known range, then the packet MUST be dropped and an alert MAY be logged. This process is summarized in Figure 3.

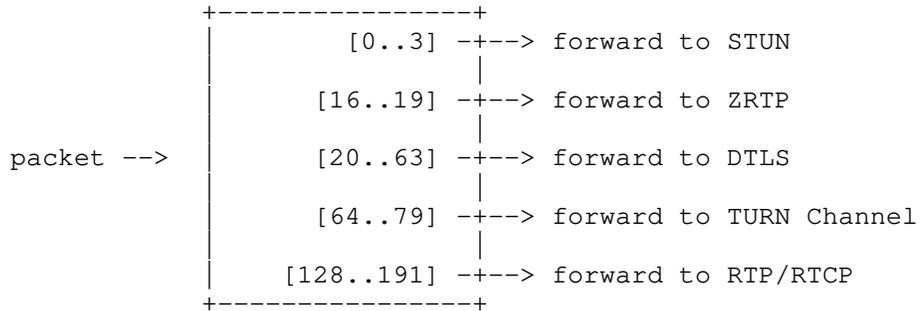


Figure 3: The DTLS-SRTP receiver's packet demultiplexing algorithm.

END OLD TEXT

NEW TEXT

The process for demultiplexing a packet is as follows. The receiver looks at the first byte of the packet. If the value of this byte is in between 0 and 3 (inclusive), then the packet is STUN. If the value is between 16 and 19 (inclusive), then the packet is ZRTP. If the value is between 20 and 63 (inclusive), then the packet is DTLS. If the value is in between 128 and 191 (inclusive) then the packet is

RTP (or RTCP, if both RTCP and RTP are being multiplexed over the same destination port). If the value is between 80 and 127 or between 192 and 255 (inclusive) then the packet is QUIC. If the value is between 64 and 79 inclusive, then if a packet has been previously forwarded that is in the range of 192 and 255, then the packet is QUIC, otherwise it is TURN Channel.

If the value does not match any known range, then the packet MUST be dropped and an alert MAY be logged. This process is summarized in Figure 3.

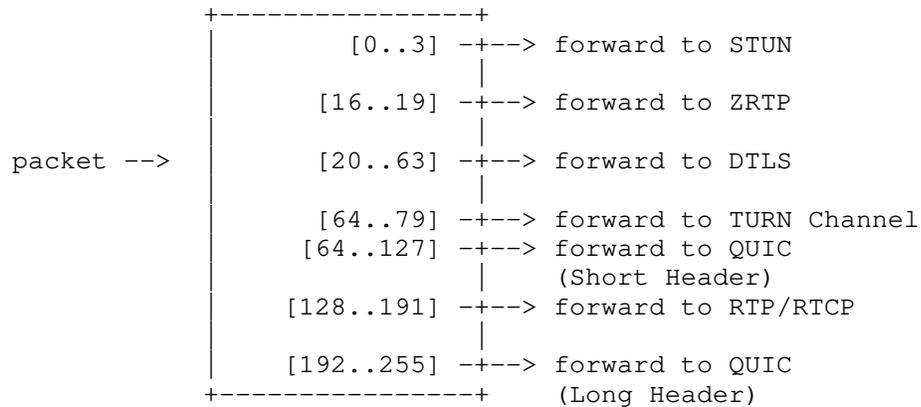


Figure 3: The receiver's packet demultiplexing algorithm.

END NEW TEXT

#### 4. Security Considerations

The solution discussed in this document could potentially introduce some additional security considerations beyond those detailed in [RFC7983].

Due to the additional logic required, if mis-implemented, heuristics have the potential to mis-classify packets.

When QUIC is used for only for data exchange, the TLS-within-QUIC exchange [I-D.ietf-quic-tls] derives keys used solely to protect the QUIC data packets. If properly implemented, this should not affect the transport of SRTP nor the derivation of SRTP keys via DTLS-SRTP, but if badly implemented, both transport and key derivation could be adversely impacted.

## 5. IANA Considerations

This document does not require actions by IANA.

## 6. References

### 6.1. Normative References

[I-D.ietf-quic-tls]

Thomson, M. and S. Turner, "Using Transport Layer Security (TLS) to Secure QUIC", draft-ietf-quic-tls-32 (work in progress), October 20, 2020.

[I-D.ietf-quic-transport]

Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-32 (work in progress), October 20, 2020.

[RFC2119]

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

[RFC3550]

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

[RFC3711]

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

[RFC5389]

Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <<http://www.rfc-editor.org/info/rfc5389>>.

[RFC5764]

McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <<http://www.rfc-editor.org/info/rfc5764>>.

[RFC5766]

Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, DOI 10.17487/RFC5766, April 2010, <<http://www.rfc->

[editor.org/info/rfc5766](http://www.rfc-editor.org/info/rfc5766)>.

- [RFC7983] Petit-Huguenin, M. and G. Salgueiro, "Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)", RFC 7983, DOI 10.17487/RFC7983, September 2016, <<http://www.rfc-editor.org/info/rfc7983>>.

## 6.2. Informative References

- [I-D.aboba-avtcore-quick-multiplexing] Aboba, B., Thatcher, P. and C. Perkins, "QUIC Multiplexing", draft-aboba-avtcore-quick-multiplexing-04 (work in progress), January 28, 2020.
- [RFC6189] Zimmermann, P., Johnston, A., Ed., and J. Callas, "ZRTP: Media Path Key Agreement for Unicast Secure RTP", RFC 6189, DOI 10.17487/RFC6189, April 2011, <<http://www.rfc-editor.org/info/rfc6189>>.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.
- [WEBRTC-QUIC] Thatcher, P. and B. Aboba, "QUIC API For Peer-to-Peer Connections", W3C Community Group Draft (work in progress), January 2020, <<https://w3c.github.io/webrtc-quick>>
- [WEBRTC-QUIC-TRIAL] Hampson, S., "RTCQuicTransport Coming to an Origin Trial Near You (Chrome 73)", January 2019, <<https://developers.google.com/web/updates/2019/01/rtcquictransport-api>>

## Acknowledgments

We would like to thank Martin Thomson, Roni Even and other participants in the IETF QUIC and AVTCORE working groups for their discussion of the QUIC multiplexing issue, and their input relating to potential solutions.

Authors' Addresses

Bernard Aboba  
Microsoft Corporation  
One Microsoft Way  
Redmond, WA 98052  
USA

Email: [bernard.aboba@gmail.com](mailto:bernard.aboba@gmail.com)

Gonzalo Salgueiro  
Cisco Systems  
7200-12 Kit Creek Road  
Research Triangle Park, NC 27709  
United States of America

Email: [gsalguei@cisco.com](mailto:gsalguei@cisco.com)

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

Email: [csp@csp Perkins.org](mailto:csp@csp Perkins.org)

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: 21 December 2020

G. Hellstrom  
Gunnar Hellstrom Accessible Communication  
19 June 2020

Real-time text solutions for multi-party sessions  
draft-hellstrom-avtcore-multi-party-rtt-solutions-02

Abstract

This document specifies methods for Real-Time Text (RTT) media handling in multi-party calls. The main transport is to carry Real-Time text by the RTP protocol in a time-sampled mode according to RFC 4103 [RFC4103]. The mechanisms enable the receiving application to present the received real-time text media separated per source, in different ways according to user preferences. Some presentation related features are also described explaining suitable variations of transmission and presentation of text.

Call control features are described for the SIP environment. A number of alternative methods for providing the multi-party negotiation, transmission and presentation are discussed and a recommendation for the main ones is provided. The main solution for SIP based centralized multi-party handling of real-time text is achieved through a media control unit coordinating multiple RTP text streams into one RTP stream.

Alternative methods using a single RTP stream and source identification inline in the text stream are also described, one of them being provided as a lower functionality fallback method for endpoints with no multi-party awareness for RTT.

Bridging methods where the text stream is carried without the contents being dealt with in detail by the bridge are also discussed.

Brief information is also provided for multi-party RTT in the WebRTC environment.

The intention is to provide background for decisions, specification and implementation of selected methods.

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 <https://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 21 December 2020.

#### Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

#### Table of Contents

1. Introduction . . . . .	4
1.1. Requirements Language . . . . .	5
2. Centralized conference model . . . . .	5
3. Requirements on multi-party RTT . . . . .	6
4. RTP based solutions . . . . .	7
4.1. Coordination of text RTP streams . . . . .	7
4.1.1. RTP-based solutions with a central mixer . . . . .	7
4.1.1.1. RTP Mixer using default RFC 4103 methods . . . . .	7
4.1.1.2. RTP Mixer using the default method but decreased transmission interval . . . . .	8
4.1.1.3. RTP Mixer with frequent transmission and indicating sources in CSRC-list . . . . .	9
4.1.1.4. RTP Mixer using timestamp to identify redundancy . . . . .	10
4.1.1.5. RTP Mixer with multiple primary data in each packet and individual sequence numbers . . . . .	11
4.1.1.6. RTP Mixer with multiple primary data in each packet . . . . .	12
4.1.1.7. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy in the packets . . . . .	13

4.1.1.8.	RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy and separate sequence number in the packets . . . .	15
4.1.1.9.	RTP Mixer indicating participants by a control code in the stream . . . . .	17
4.1.1.10.	Mixing for multi-party unaware user agents . . .	18
4.1.2.	RTP-based bridging with minor RTT media contents reformatting by the bridge . . . . .	20
4.1.2.1.	RTP Translator sending one RTT stream per participant . . . . .	20
4.1.2.2.	Distributing packets in an end-to-end encryption structure . . . . .	23
4.1.2.3.	Mesh of RTP endpoints . . . . .	23
4.1.2.4.	Multiple RTP sessions, one for each participant . . . . .	24
5.	Preferred RTP-based multi-party RTT transport method . . . .	25
6.	Session control of RTP-based multi-party RTT sessions . . . .	25
6.1.	Implicit RTT multi-party capability indication . . . . .	26
6.2.	RTT multi-party capability declared by SIP media-tags . .	27
6.3.	SDP media attribute for RTT multi-party capability indication . . . . .	28
6.4.	Simplified SDP media attribute for RTT multi-party capability indication . . . . .	29
6.5.	SDP format parameter for RTT multi-party capability indication . . . . .	30
6.6.	A text media subtype for support of multi-party rtt . . .	31
6.7.	Preferred capability declaration method for RTP-based transport. . . . .	31
6.8.	Identification of the source of text for RTP-based solutions . . . . .	32
7.	RTT bridging in WebRTC . . . . .	32
7.1.	RTT bridging in WebRTC with one data channel per source . . . . .	32
7.2.	RTT bridging in WebRTC with one common data channel . . .	33
7.3.	Preferred rtt multi-party method for WebRTC . . . . .	34
8.	Presentation of multi-party text . . . . .	34
8.1.	Associating identities with text streams . . . . .	34
8.2.	Presentation details for multi-party aware endpoints. . .	35
8.2.1.	Bubble style presentation . . . . .	35
8.2.2.	Other presentation styles . . . . .	37
9.	Presentation details for multi-party unaware endpoints. . . .	37
10.	Security Considerations . . . . .	37
11.	IANA Considerations . . . . .	38
12.	Congestion considerations . . . . .	38
13.	Acknowledgements . . . . .	38
14.	Change history . . . . .	38
14.1.	Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-02 . .	38

14.2.	Changes to	
	draft-hellstrom-avtcore-multi-party-rtt-solutions-01 . . .	38
14.3.	Changes from draft-hellstrom-mmusic-multi-party-rtt-02 to	
	draft-hellstrom-avtcore-multi-party-rtt-solutions-00 . . .	39
14.4.	Changes from version	
	draft-hellstrom-mmusic-multi-party-rtt-01 to -02 . . . . .	39
15.	References . . . . .	39
15.1.	Normative References . . . . .	39
15.2.	Informative References . . . . .	39
	Author's Address . . . . .	43

## 1. Introduction

Real-time text (RTT) is a medium in real-time conversational sessions. Text entered by participants in a session is transmitted in a time-sampled fashion, so that no specific user action is needed to cause transmission. This gives a direct flow of text in the rate it is created, that is suitable in a real-time conversational setting. The real-time text medium can be combined with other media in multimedia sessions.

Media from a number of multimedia session participants can be combined in a multi-party session. The present document specifies how the real-time text streams can be handled in multi-party sessions. Recommendations are provided for preferred methods.

The description is mainly focused on the transport level, but also describes a few session and presentation level aspects.

Transport of real-time text is specified in RFC 4103 [RFC4103] RTP Payload for text conversation. It makes use of RFC 3550 [RFC3550] Real Time Protocol, for transport. Robustness against network transmission problems is normally achieved through redundant transmission based on the principle from RFC 2198 [RFC2198], with one primary and two redundant transmission of each text element. Primary and redundant transmissions are combined in packets and described by a redundancy header. This transport is usually used in the SIP Session Initiation Protocol RFC 3261 [RFC3261] environment.

A very brief overview of functions for real-time text handling in multi-party sessions is described in RFC 4597 [RFC4597] Conferencing Scenarios, sections 4.8 and 4.10. The present specification builds on that description and indicates which protocol mechanisms should be used to implement multi-party handling of real-time text.

Real-time text can also be transported in the WebRTC environment, by using WebRTC data channels according to [I-D.ietf-mmusic-t140-usage-data-channel]. Multi-party aspects for WebRTC solutions are briefly covered.

### 1.1. Requirements Language

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

## 2. Centralized conference model

In the centralized conference model for SIP, introduced in RFC 4353 [RFC4353] "A Framework for Conferencing with the Session Initiation Protocol (SIP)", one function co-ordinates the communication with participants in the multi-party session. This function also controls media mixer functions for the media appearing in the session. The central function is common for control of all media, while the media mixers may work differently for each media.

The central function is called the Focus UA. Many variants exist for setting up sessions including the multipoint control centre. It is not within scope of this description to describe these, but rather the media specific handling in the mixer required to handle multi-party calls with RTT.

The main principle for handling real-time text media in a centralized conference is that one RTP session for real-time text is established including the multipoint media control centre and the participating endpoints which are going to have real-time text exchange with the others.

The different possible mechanisms for mixing and transporting RTT differs in the way they multiplex the text streams and how they identify the sources of the streams. RFC 7667 [RFC7667] describes a number of possible use cases for RTP. This specification refers to different sections of RFC 7667 for further reading of the situations caused by the different possible design choices.

The recommended method for using RTT in a centralized conference model is specified in [I-D.ietf-avtcore-multi-party-rtt-mix] based on the recommendations in the present document.

Real-time text can also be transported in the WebRTC environment, by using WebRTC data channels according to [I-D.ietf-mmusic-t140-usage-data-channel]. Ways to handle multi-party calls in that environment are also specified.

### 3. Requirements on multi-party RTT

The following requirements are placed on multi-party RTT:

A solution shall be applicable to IMS (3GPP TS 22.173) [TS22173], SIP based VoIP and Next Generation Emergency Services (NENA i3 [NENAI3], ETSI TS 103 479 [TS103479], RFC 6443 [RFC6443]).

The transmission interval for text must not be longer than 500 milliseconds when there is anything available to send. Ref ITU-T T.140 [T140].

If text loss is detected or suspected, a missing text marker shall be inserted in the text stream. Ref ITU-T T.140 Amendment 1 [T140ad1]. ETSI EN 301 549 [EN301549]

The display of text from the members of the conversation shall be arranged so that the text from each participant is clearly readable, and its source and the relative timing of entered text is visualized in the display. Mechanisms for looking back in the contents from the current session should be provided. The text should be displayed as soon as it is received. Ref ITU-T T.140 [T140]

Bridges must be multimedia capable (voice, video, text). Ref NENA i3 STA-010.2. [NENAI3]

R7: It MUST be possible to use real-time text in conferences both as a medium of discussion between individual participants (for example, for sidebar discussions in real-time text while listening to the main conference audio) and for central support of the conference with real-time text interpretation of speech. Ref RFC 5194. [RFC5194]

It should be possible to protect RTT contents with usual means for privacy and integrity. Ref RFC 6881 section 16. [RFC6881]

Conferencing procedures are documented in RFC 4579 [RFC4579]. Ref NENA i3 STA-010.2. [NENAI3]

Conferencing applies to any kind of media stream by which users may want to communicate. Ref 3GPP TS 24.147 [TS24147]

The framework for SIP conferences is specified in RFC 4353 [RFC4353]. Ref 3GPP TS 24.147 [TS24147]

The mixer performance requirements can be expressed in two numbers.

1) The number of participants who can transmit simultaneously with the text not being delayed in the mixer more than 500 milliseconds. This requirement is depending on the application. Five simultaneous transmitting participants is a sufficiently high number for most situations.

2) The switching time from when the mixer is transmitting text from one participant and text arrives from another participant, until the mixer sends the text from the second participant. This time should not be more than 500 milliseconds when there are up to five participants sending text simultaneously.

#### 4. RTP based solutions

##### 4.1. Coordination of text RTP streams

Coordinating and sending text RTP streams in the multi-party session can be done in a number of ways. The most suitable methods are specified here with pros and cons.

A receiving and presenting endpoint MUST separate text from the different sources and identify and display them accordingly.

##### 4.1.1. RTP-based solutions with a central mixer

A set of solutions can be based on the central RTP mixer. They are described here and a preferred method selected.

##### 4.1.1.1. RTP Mixer using default RFC 4103 methods

Without any extra specifications, a mixer would transmit with 300 milliseconds intervals, and use RFC 4103 [RFC4103] with the default redundancy of one original and two redundant transmissions. The source of the text would be indicated by a single member in the CSRC list. Text from different sources cannot be transmitted in the same packet. Therefore, from the time when the mixer sent one piece of new text from one source, it will need to transmit that text again twice as redundant data, before it can send text from another source. The switching time will thus be 900 milliseconds. The mixer can not even send text from two simultaneous sources without introducing more than 500 milliseconds delay. This is clearly insufficient.

Pros:

Only a capability negotiation method is needed. No other update of standards are needed, just a general remark that traditional RTP-mixing is used.

## Cons:

Clearly insufficient mixer switching performance.

A bit complex handling of transmission when there is new text available from more than one source. The mixer needs to send two packets more with redundant text from the current source before starting to send anything from the other source.

#### 4.1.1.2. RTP Mixer using the default method but decreased transmission interval

This method makes use of the default RTP-mixing method briefly described in Section 4.1.1.1. The only difference is that the transmission interval is decreased to 100 milliseconds when there is text from more than one source available for transmission. This increases the switching performance to three source switches per second. The delay of new text from a participant can be one second if five users send new text simultaneously. Text from two simultaneous users would not get more delayed than 400 ms.

## Pros:

Minor influence on standards

Can be sdp-declared as "text/red" with a multi-party attribute for capability negotiation.

## Cons:

Too long delay of new text from more than two simultaneous sources.

Slightly higher risk for loss of text at bursty packet loss than for the recommended transmission interval (300 ms) for RFC 4103.

When complete loss of packets occur (beyond recovery), it is not possible to deduct from which source text was lost.

A bit complex handling of transmission when there is new text available from more than one source. The mixer needs to send two packets more with redundant text from the current source before starting to send anything from the other source.

#### 4.1.1.3. RTP Mixer with frequent transmission and indicating sources in CSRC-list

An RTP media mixer combines text from participants into one RTP stream, thus all using the same destination address/port combination, the same RTP SSRC, and one sequence number series as described in Section 7.1 and 7.3 of RTP RFC 3550 [RFC3550] about the Mixer function. This method is also briefly described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

The sources of the text in each RTP packet are identified by the CSRC list in the RTP packets, containing the SSRC of the initial sources of text. The order of the CSRC parameters is with the SSRC of the source of the primary text first, followed by the SSRC of the first level redundancy, and then the second level redundancy.

The transmission interval should be 100 milliseconds when there is text to transmit from more than one source, and otherwise 300 ms.

The identification of the sources is made through the CSRC fields and can be made more readable at the receiver through the RTCP SDES CNAME and NAME packets as described in RTP [RFC3550].

Information provided through the notification according to RFC 4575 [RFC4575] when the participant joined the conference provides also suitable information and a reference to the SSRC.

A receiving endpoint is supposed to separate text items from the different sources and identify and display them accordingly.

The ordered CSRC lists in the RFC 4103 [RFC4103] packets make it possible to recover from loss of one and two packets in sequence and assign the recovered text to the right source. For more loss, a marker for possible loss should be inserted or presented.

The conference server needs to have authority to decrypt the payload in the received RTP packets in order to be able to recover text from redundant data or insert the missing text marker in the stream, and repack the text in new packets.

Even if the format is very similar to "text/red" of RFC 4103, it has been indicated that it needs to be declared as a new media subtype, e.g. "text/rex".

Pros:

This method has low overhead and less complexity than the methods in Section 4.1.1.1, Section 4.1.1.2, Section 4.1.1.4 and Section 4.1.1.6.

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream (normally primary and two redundant levels).

This method can be implemented with most RTP implementations.

The source switching performance is sufficient for well-behaving conference participants. There can be switching between five source per second with an introduced delay of maximum 500 ms. With just two parties typing simultaneously, the delay will be a maximum of 100 ms.

Cons:

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the sources of the totally lost data.

Slightly higher risk for loss of text at bursty packet loss than for the recommended transmission interval for RFC 4103.

Requires a different sub media format, e.g. "text/rex".

The conference server needs to be allowed to decrypt/encrypt the packet payload. This is however normal for media mixers for other media.

#### 4.1.1.4. RTP Mixer using timestamp to identify redundancy

This method has text only from one source per packet, as the original RFC 4103 [RFC4103] specifies. Packets with text from different sources are instead allowed to be merged. The recovery procedure in the receiver will use the RTP timestamp and timestamp offsets in the redundancy headers to evaluate if a piece of redundant data should be recovered or not in case of packet loss.

In this method, the transmission interval is 100 milliseconds when text from more than one source is available for transmission.

Pros:

The format of each packet is equal to what is specified in RFC 4103 [RFC4103].



Transmission is done as soon as there is new text available, but not with shorter interval than 150 ms and not longer than 300 ms while there is anything to send.

A new media subtype is needed, e.g. "text/rex".

This is an SDP offer example for both traditional "text/red" and multi-party "text/rex" format:

```
m=text 11000 RTP/AVP 101 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=rtpmap:101 rex/1000
a=fmtp:100 98/98/98
a=fmtp:101 98/98/98
```

Pros:

The source switching performance is good. Text from 16 participants can be transmitted simultaneously.

New text from 16 simultaneous sources can be transmitted within 300 milliseconds. This is good performance.

When more consecutive packet loss than the number of generations of redundant data appears, it is still possible to deduct the sources of the totally lost data, when next text from these sources arrive.

Cons:

The format of each packet is different from what is specified in RFC 4103 [RFC4103].

A new media subtype is needed.

The recovery procedure is a bit complex.

#### 4.1.1.6. RTP Mixer with multiple primary data in each packet

This method allows primary as well as redundant text from more than one source per packet. The packet payload contains an ordered set of redundant and primary data with the same number of generations of redundancy as once agreed in the SDP negotiation. The data header reflects these parts of the payload. The CSRC list contains one CSRC member per source in the payload and in the same order. The

The maximum number of members in the CSRC-list is 16, and that is therefore the maximum number of sources that can be represented in each packet provided that all data can be fitted into the size allowable in one packet.

Transmission is done as soon as there is new text available, but not with shorter interval than 150 ms and not longer than 300 ms while there is anything to send.

A new media subtype is needed, e.g. "text/rex".

SDP would be the same as in Section 4.1.1.6.

Pros:

The source switching performance is good. Text from 16 participants can be transmitted simultaneously.

New text from 16 simultaneous sources can be transmitted within 150 milliseconds. This is good performance.

Cons:

The format of each packet is different from what is specified in RFC 4103 [RFC4103].

A new media subtype is needed.

The recovery procedure is a bit complex [RFC4103].

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the sources of the totally lost data.

#### 4.1.1.7. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy in the packets

This method allows primary data from one source and redundant text from other sources in each packet. The packet payload contains primary data in "text/t140" format, and redundant data in RFC 5109 FEC [RFC5109] format called "text/ulpfec". That means that the redundant data contains the sequence number and the CSRC and other characteristics from the RTP header when the data was sent as primary. The redundancy can be sent at a selected number of packets after when it was sent as primary, in order to improve the protection against bursty packet loss. The redundancy level is recommended to be the same as in original RFC 4103.

RFC 4103 says that the protection against loss can be made by other methods than plain redundancy, so this method is in line with that statement.

Transmission is done as soon as there is new text available, but not with shorter interval than 100 ms and not longer than 300 ms while there is anything to send (new or redundant text).

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the sources of the totally lost data.

The sdp can indicate the format as "text/red" with "text/ulpfec" redundant data in this way. with traditional RFC 4103 with "text/red" with "text/t140" as redundant data as a fallback.

```
m=text 49170 RTP/AVP 98 101 100 102
a=rtpmap:98 red/1000
a=fmtp:98 100/102/102
a=rtpmap:102 ulpfec/1000
a=rtpmap:100 t140/1000
a=rtpmap:101 red/1000
a=fmtp:101 100/100/100
a=fmtp:100 cps=200
```

The "text/ulpfec" format includes an indication of how far back the redundancy belongs, making it possible to cover bursty packet loss better than the other formats with short transmission intervals. For real-time text, it is recommended to send three packets between the primary and the redundant transmissions of text. That makes the transmission cover between 500 and 1500 ms of bursty packet loss. The variation is because of the varying packet interval between many and one simultaneously transmitting source.

The "text/ulpfec" format has a number of parameters. One is the length of the data to be protected which in this case must be the whole t140block.

Pros:

The source switching performance is good. Text from 5 participants can be transmitted within 500 ms.

Good recovery from bursty packet loss.

The method is based on existing standards. No new registrations are needed.

#### Cons:

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the sources of the totally lost data.

Even if the switching performance is good, it is not as good as for the method called "RTP Mixer with multiple primary data in each packet" Section 4.1.1.6. With more than 5 simultaneously sending sources, there will be a noticeable delay of text of over 500 ms, with 100 ms added per simultaneous source. This is however beyond the requirements and would be a concern only in congestion situations.

The recovery procedure is a bit complex [RFC5109].

There is more overhead in terms of extra data and extra packets sent than in the other methods. With the recommended two redundant generations of data, each packet will be 36 bytes longer than with traditional RFC 4103, and at each pause in transmission five extra packets with only redundant data will be sent compared to two extra packets for the traditional RFC 4103 case.

#### 4.1.1.8. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy and separate sequence number in the packets

This method allows primary data from one source and redundant text from other sources in each packet. The packet payload contains primary data in a new "text/t140e" format, and redundant data in RFC 5109 FEC [RFC5109] format called "text/ulpfec". That means that the redundant data contains the sequence number and the CSRC and other characteristics from the RTP header when the data was sent as primary. The redundancy can be sent at a selected number of packets after when it was sent as primary, in order to improve the protection against bursty packet loss. The redundancy level is recommended to be the same as in original RFC 4103. The "text/t140e" format contains a source-specific sequence number and the t140block.

RFC 4103 says that the protection against loss can be made by other methods than plain redundancy, so this method is in line with that statement.

Transmission is done as soon as there is new text available, but not with shorter interval than 100 ms and not longer than 300 ms while there is anything to send (new or redundant text).

When more consecutive packet loss than the number of generations of redundant data appears, it is possible to deduct which sources lost data when new data arrives from the sources. This is done by monitoring the received source specific sequence numbers preceding the text.

This is an example of how can indicate the format as "text/red" with "text/t140e" as primary and "text/ulpfec" redundant data, with traditional RFC 4103 with "text/red" with "text/t140" as redundant data as a fallback.

```
m=text 49170 RTP/AVP 98 101 100 102 103
a=rtpmap:98 red/1000
a=fmtp:98 100/102/102
a=rtpmap:102 ulpfec/1000
a=rtpmap:103 t140/1000
a=rtpmap:100 t140e/1000
a=rtpmap:101 red/1000
a=fmtp:101 103/103/103
a=fmtp:100 cps=200
```

The "text/ulpfec" format includes an indication of how far back the redundancy belongs, making it possible to cover bursty packet loss better than the other formats with short transmission intervals. For real-time text, it is recommended to send three packets between the primary and the redundant transmissions of text. That makes the transmission cover between 500 and 1500 ms of bursty packet loss. The variation is because of the varying packet interval between many and one simultaneously transmitting source.

The "text/ulpfec" format has a number of parameters. One is the length of the data to be protected which in this case must be the whole t140block.

Pros:

The source switching performance is good. Text from 5 participants can be transmitted within 500 ms.

Good recovery from bursty packet loss.

The method is based on an existing standard for FEC.

When more consecutive packet loss than the number of generations of redundant data appears, it is possible to deduct the source of the lost data when new text arrives from the source.

Cons:

Even if the switching performance is good, it is not as good as for the method called "RTP Mixer with multiple primary data in each packet" Section 4.1.1.6. With more than 5 simultaneously sending sources, there will be a noticeable delay of text of over 500 ms, with 100 ms added per simultaneous source. This is however beyond the requirements and would be a concern only in congestion situations.

The recovery procedure is a bit complex [RFC5109].

There is more overhead in terms of extra data and extra packets sent than in the other methods. With the recommended two redundant generations of data, each packet will be 40 bytes longer than with traditional RFC 4103, and at each pause in transmission five extra packets with only redundant data will be sent compared to two extra packets for the traditional RFC 4103 case.

A new text media subtype "text/t140e" needs to be registered.

#### 4.1.1.9. RTP Mixer indicating participants by a control code in the stream

Text from all participants except the receiving one is transmitted from the media mixer in the same RTP session and stream, thus all using the same destination address/port combination, the same RTP SSRC and , one sequence number series as described in Section 7.1 and 7.3 of RTP RFC 3550 [RFC3550] about the Mixer function. The sources of the text in each RTP packet are identified by a new defined T.140 control code "c" followed by a unique identification of the source in UTF-8 string format.

The receiver can use the string for presenting the source of text. This method is on the RTP level described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

The inline coding of the source of text is applied in the data stream itself, and an RTP mixer function is used for coordinating the sources of text into one RTP stream.

Information uniquely identifying each user in the multi-party session is placed as the parameter value "n" in the T.140 application protocol function with the function code "c". The identifier shall thus be formatted like this: SOS c n ST, where SOS and ST are coded as specified in ITU-T T.140 [T140]. The "c" is the letter "c". The n parameter value is a string uniquely identifying the source. This parameter shall be kept short so that it can be repeated in the transmission without concerns for network load.

A receiving endpoint is supposed to separate text items from the different sources and identify and display them accordingly.

The conference server need to be allowed to decrypt/encrypt the packet payload in order to check the source and repack the text.

Pros:

If loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103]stream. (normally primary and two redundant levels.

This method can be implemented with most RTP implementations.

The method can also be used with other transports than RTP

Cons:

The method implies a moderate load by the need to insert the source often in the stream.

If more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the source of the totally lost data.

The mixer needs to be able to generate suitable and unique source identifications which are suitable as labels for the sources.

Requires an extension on the ITU-T T.140 standard, best made by the ITU.

There is a risk that the control code indicating the change of source is lost and the result is false source indication of text.

The conference server need to be allowed to decrypt/encrypt the packet payload.

#### 4.1.1.10. Mixing for multi-party unaware user agents

Multi-party real-time text contents can be transmitted to multi-party unaware user agents if source labelling and formatting of the text is performed by a mixer. This method has the limitations that the layout of the presentation and the format of source identification is purely controlled by the mixer, and that only one source at a time is allowed to present in real-time. Other sources need to be stored temporarily waiting for an appropriate moment to switch the source of transmitted text. The mixer controls the switching of sources and

inserts a source identifier in text format at the beginning of text after switch of source. The logic of the mixer to detect when a switch is appropriate should detect a number of places in text where a switch can be allowed, including new line, end of sentence, end of phrase, a period of inactivity, and a word separator after a long time of active transmission.

This method MAY be used when no support for multi-party awareness is detected in the receiving endpoint. The base for this method is described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

See [I-D.ietf-avtcore-multi-party-rtt-mix] for a procedure for mixing RTT for a conference-unaware endpoint.

Pros:

Can be transmitted to conference-unaware endpoints.

Can be used with other transports than RTP

Cons:

Does not allow full real-time presentation of more than one source at a time. Text from other sources will be delayed.

The only realistic presentation format is a style with the text from the different sources presented with a text label indicating source, and the text collected in a chat style presentation but with more frequent turn-taking.

Endpoints often have their own system for adding labels to the RTT presentation. In that case there will be two levels of labels in the presentation, one for the mixer and one for the sources.

If loss of more packets than can be recovered by the redundancy appears, it is not possible to detect which source was struck by the loss. It is also possible that a source switch occurred during the loss, and therefore a false indication of the source of text can be provided to the user after such loss.

Because of all these cons, this method is not recommended and MUST NOT be used as the main method, but only as the last resort for backwards interoperability with multi-party unaware endpoints.

The conference server need to be allowed to decrypt/encrypt the packet payload.

#### 4.1.2. RTP-based bridging with minor RTT media contents reformatting by the bridge

It may be desirable to send text in a multi-party setting in a way that allows the text stream contents to be distributed without being dealt with in detail in any central server. A number of such methods are described. However, when writing this specification, no one of these methods have a specified way of establishing the session by sdp.

##### 4.1.2.1. RTP Translator sending one RTT stream per participant

Within the RTP session, text from each participant is transmitted from the RTP media translator (bridge) in a separate RTP stream, thus using the same destination address/port combination, the same payload type number (PT) but separate RTP SSRC parameters and sequence number series as described in Section 7.1 and 7.2 of RTP RFC 3550 [RFC3550] about the Translator function. The source of the text in each RTP packet is identified by the SSRC parameter in the RTP packets, containing the SSRC of the initial source of text.

A receiving and presenting endpoint is supposed to separate text items from the different sources and identify and display them in a suitable way.

This method is described in RFC 7667, section 3.5.1 Relay-transport translator or 3.5.2 Media translator [RFC7667].

The identification of the source is made through the SSRC. The translation to a readable label can be done by mapping to information from the RTCP SDES CNAME and NAME packets as described in RTP[RFC3550], and also through information in the text media member in the conference notification described in RFC 4575 [RFC4575].

The sdp exchange for establishing this mixing type can be equal to what is used for basic two-party use of RFC 4103 with just an added attribute for indicating multi-party capability.

```
m=text 49170 RTP/AVP 98 103
a=rtpmap:98 red/1000
a=fmtp:98 103/103/103
a=rtpmap:103 t140/1000
a=fmtp:103 cps=150
a=RTT-mix:RTP-translator
```

A similar answer including the same RTT-mix attribute would indicate that multi-party coding can begin. An answer without the same RTT-mix attribute could result in diversion to use of the mixing method for multi-party unaware endpoints Section 4.1.1.10 if more than two parties are involved in the session.

The bridge can add new sources in the communication to a participant by first sending a conference notification according to RFC 4575 [RFC4575] with the SSRC of the new source included in the corresponding "text" media member, or by sending an RTCP message with the new SSRC in an SDES packet.

A receiver should be prepared to receive such indications of new streams being added to the multi-party session, so that the new SSRC is not taken for a change in SSRC value for an already established RTP stream.

Transmission, reception, packet loss recovery and text loss indication is performed per source in the separate RTP streams in the same way as in two-party sessions with RFC 4103 [RFC4575].

Text is recommended to be sent by the bridge as soon as it is available for transmission, but not less than 250 ms after a previous transmission. This will in many cases result in close to 0 added delay by the bridge, because most RTT senders use a 300 ms transmission interval.

It is sometimes said that this configuration is not supported by current media declarations in sdp. RFC 3264 [RFC3264] specifies in some places that one media description is supposed to describe just one RTP media stream. However this is not directly referencing an RTP stream, and use of multiple RTP streams in the same RTP session is recommended in many other RFCs.

This confusion is clarified in RFC 5576 [RFC5576] section 3 by the following statements:

"The term "media stream" does not appear in the SDP specification itself, but is used by a number of SDP extensions, for instance, Interactive Connectivity Establishment (ICE) [ICE], to denote the object described by an SDP media description. This term is unfortunately rather confusing, as the RTP specification [RFC3550] uses the term "media stream" to refer to an individual media source or RTP packet stream, identified by an SSRC, whereas an SDP media stream describes an entire RTP session, which can contain any number of RTP sources."

In most cases, it will be sufficient that new sources are introduced with a conference notification or RTCP message. However, RFC 5576 [RFC5576] specifies attributes which may be used to more explicitly announce new sources or restart of earlier established RTP streams.

This method is encouraged by draft-ietf-avtcore-multiplex-guidelines [I-D.ietf-avtcore-multiplex-guidelines] section 5.2.

Normal operation will be that the bridge receives text packets from the source and handles any text recovery and indication of loss needed before queueing the resulting clean text for transmission from the bridge to the receivers.

It may however also be possible for the bridge to just convey the packet contents as received from the sources, with minor adjustments, and let the receiving endpoint handle all aspects of recovery and indication of loss, even for the source to bridge path. In that case also the sequence number must be maintained as it was at reception in the bridge. This mode needs further study before application.

#### Pros:

This method is the natural way to do multi-party bridging with RFC 4103 based RTT. Only a small addition is included in the session establishment to verify capability by the parties because many implementations are done without multi-party capability.

This method has moderate overhead in terms of work for the mixer, but high in terms of packet transmission rate. Five sources sending simultaneously cause the bridge to send 15 packets per second to each receiver.

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream (normally primary and two redundant levels).

More loss than what can be recovered, can be detected and the marker for text loss can be inserted in the correct stream.

It may be possible in some scenarios to keep the text encrypted through the Translator.

Minimal delay. The delay can often be kept close to 0 with at least 5 simultaneous sending participants.

#### Cons:

There may be RTP implementations not supporting the Translator model. They will need to use the fall-back to multi-party-unaware mixing. An investigation about how common this is is needed before the method is used.

With many simultaneous sending sources, the total rate of packets will be high, and can cause congestion. The requirement to handle 5 simultaneous sources in this specification will cause 15 packets per second that is on the high side but still manageable in most cases, e.g. considering that audio usually use 50 packets per second.

#### 4.1.2.2. Distributing packets in an end-to-end encryption structure

In order to achieve end-to-end encryption, it is possible to let the packets from the sources just pass through a central distributor, and handle the security agreements between the participants. Specifications exist for a framework with this functionality for application on RTP based conferences in [I-D.ietf-perc-private-media-framework]. The RTP flow and mixing characteristics has similarities with the method described under "RTP Translator sending one RTT stream per participant" above. RFC 4103 RTP streams [RFC4103] would fit into the structure and it would provide a base for end-to-end encrypted rtt multi-party conferencing.

Pros:

Good security

Straightforward multi-party handling.

Cons:

Does not operate under the usual SIP central conferencing architecture.

Requires the participants to perform a lot of key handling.

Is work in progress when this is written.

#### 4.1.2.3. Mesh of RTP endpoints

Text from all participants are transmitted directly to all others in one RTP session, without a central bridge. The sources of the text in each RTP packet are identified by the source network address and the SSRC.

This method is described in RFC 7667, section 3.4 Point to multi-point using mesh [RFC7667].

**Pros:**

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream. (normally primary and two redundant levels).

This method can be implemented with most RTP implementations.

Transmitted text can also be used with other transports than RTP

**Cons:**

This model is not described in IMS, NENA and EENA specifications, and does therefore not meet the requirements.

Requires a drastically increasing number of connections when the number of participants increase.

**4.1.2.4. Multiple RTP sessions, one for each participant**

Text from all participants are transmitted directly to all others in one RTP session each, without a central bridge. Each session is established with a separate media description in SDP. The sources of the text in each RTP packet are identified by the source network address and the SSRC.

**Pros:**

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream. (normally primary and two redundant levels).

Complete loss of text can be indicated in the received stream.

This method can be implemented with most RTP implementations.

End-to-end encryption is achievable.

**Cons:**

This method is not described in IMS, NENA and ETSI specifications and does therefore not meet the requirements.

A lot of network resources are spent on setting up separate sessions for each participant.

## 5. Preferred RTP-based multi-party RTT transport method

For RTP transport of RTT using RTP-mixer technology, one method for multi-party mixing and transport stand out as fulfilling the goals best and is therefore recommended. That is: TBD

For RTP transport in separate streams or sessions, no current recommendation can be made. A bridging method in the process of standardisation with interesting characteristics is the end-to-end encryption model "perc" Section 4.1.2.2.

## 6. Session control of RTP-based multi-party RTT sessions

General session control aspects for multi-party sessions are described in RFC 4575 [RFC4575] A Session Initiation Protocol (SIP) Event Package for Conference State, and RFC 4579 [RFC4579] Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents. The nomenclature of these specifications are used here.

The procedures for a multi-party aware model for RTT-transmission shall only be applied if a capability exchange for multi-party aware real-time text transmission has been completed and a supported method for multi-party real-time text transmission can be negotiated.

A method for detection of conference-awareness for centralized SIP conferencing in general is specified in RFC 4579 [RFC4579]. The focus sends the "isfocus" feature tag in a SIP Contact header. This causes the conference-aware endpoint to subscribe to conference notifications from the focus. The focus then sends notifications to the endpoint about entering and disappearing conference participants and their media capabilities. The information is carried XML-formatted in a 'conference-info' block in the notification according to RFC 4575 [RFC4575]. The mechanism is described in detail in RFC 4575 [RFC4575].

Before a conference media server starts sending multi-party RTT to an endpoint, a verification of its ability to handle multi-party RTT must be made. A decision on which mechanism to use for identifying text from the different participants must also be taken, implicitly or explicitly. These verifications and decisions can be done in a number of ways. The most apparent ways are specified here and their pros and cons described. One of the methods is selected to be the one to be used by implementations of the centralized conference model according to this specification.

### 6.1. Implicit RTT multi-party capability indication

Capability for RTT multi-party handling can be decided to be implicitly indicated by session control items.

The focus may implicitly indicate multi-party RTT capability by including the media child with value "text" in the RFC 4575 [RFC4575] conference-info provided in conference notifications.

An endpoint may implicitly indicate multi-party RTT capability by including the text media in the SDP in the session control transactions with the conference focus after the subscription to the conference has taken place.

The implicit RTT capability indication means for the focus that it can handle multi-party RTT according to the preferred method indicated in the RTT multi-party methods section above.

The implicit RTT capability indication means for the endpoint that it can handle multi-party RTT according to the preferred method indicated in the RTT multi-party methods section above.

If the focus detects that an endpoint implicitly declared RTT multi-party capability, it SHALL provide RTT according to the preferred method.

If the focus detects that the endpoint does not indicate any RTT multi-party capability, then it shall either provide RTT multi-party text in the way specified for conference-unaware endpoint above, or refuse to set up the session.

If the endpoint detects that the focus has implicitly declared RTT multi-party capability, it shall be prepared to present RTT in a multi-party fashion according to the preferred method.

Pros:

Acceptance of implicit multi-party capability implies that no standardisation of explicit RTT multi-party capability exchange is required.

Cons:

If other methods for multi-party RTT are to be used in the same implementation environment as the preferred ones, then capability exchange needs to be defined for them.

Cannot be used outside a strictly applied SIP central conference model.

## 6.2. RTT multi-party capability declared by SIP media-tags

Specifications for RTT multi-party capability declarations can be agreed for use as SIP media feature tags, to be exchanged during SIP call control operation according to the mechanisms in RFC 3840 [RFC3840] and RFC 3841 [RFC3841]. Capability for the RTT Multi-party capability is then indicated by the media feature tag "rtt-mix", with a set of possible values for the different possible methods.

The possible values in the list may for example be:

rtt-mixer

perc

rtt-mixer indicates capability for using the RTP-mixer based presentation of multi-party text.

perc indicates capability for using the perc based transmission of multi-party text.

Example: Contact: <sip:a2@beco.example.com>

```
;methods="INVITE,ACK,OPTIONS,BYE,CANCEL"
```

```
;+sip.rtt-mix="rtt-mixer"
```

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the media tag can be reduced to a single tag with no list of values.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that UA.

When no common method is declared, then only the fallback method for multi-party unaware participants can be used, or the session dropped.

If more than one text media section is included in SDP, all must be capable of using the declared RTT multi-party method.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can guide call routing to a suitable capable focus.

Cons:

Requires standardization and IANA registration.

Is not stream specific. If more than one text stream is specified, all must have the same type of multi-party capability.

Cannot be used in the WebRTC environment.

### 6.3. SDP media attribute for RTT multi-party capability indication

An attribute can be specified on media level, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The attribute can have the name "rtt-mix".

More than one attribute can be included in one media description.

The attribute can have a value. The value can for example be:

rtt-mix:rtp-mixer

rtt-mix:rtp-translator

rtt-mix:perc

rtp-mixer indicates capability for using the RTP-mixer and CSRC-list based mixing of multi-party text.

rtp-translator indicates capability for using the RTP-translator based mixing

perc indicates capability for using the perc based transmission of multi-party text.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that endpoint.

When no common method is declared, then only the fallback method for multi-party unaware endpoints can be used.

Example: a=rtt-mix:rtp-mixer

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the attribute can be reduced to a single attribute with no list of values.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can be used on specific text media.

Can be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

Cannot guide SIP routing.

#### 6.4. Simplified SDP media attribute for RTT multi-party capability indication

An attribute can be specified on media level, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The attribute can have the name "rtt-mix" with no value. It would be selected and used if only one method for multi-party rtt is brought forward from this specification, and the other suppressed or found to be possible to negotiate in another way.

An offer-answer exchange should take place and if both parties specify "rtt-mix" capability, the selected mixing method shall be used.

When no common method is declared, then only the fallback method for multi-party unaware endpoints can be used, or the session not accepted for multi-party use.

Example: a=rtt-mix

Pros:

Provides a clear decision method.

Very simple syntax and semantics.

Can be used on specific text media.

Could possibly be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

If another RTT mixing method is also specified in the future, then that method may also need to specify and register its own attribute, instead of if an attribute with a parameter value is used, when only an addition of a new possible value is needed.

Cannot guide SIP routing.

#### 6.5. SDP format parameter for RTT multi-party capability indication

An FMTP format parameter can be specified for the RFC 4103 [RFC4103]media, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The parameter can have the name "rtt-mix", with one or more of its possible values.

The possible values in the list are:

rtp-mixer

perc

rtp-mixer indicates capability for using the RTP-mixer based mixing and presentation of multi-party text using the CSRC-list.

perc indicates capability for using the perc based transmission of multi-party text.

Example: a=fmtp 96 98/98/98 rtt-mix=rtp-mixer

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the parameter can be reduced to a single parameter with no list of values.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that UA.

When no common method is declared, then only the fallback method can be used, or the session denied.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can be used on specific text media.

Can be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

May cause interop problems with current RFC4103 [RFC4103] implementations not expecting a new fntp-parameter.

Cannot guide SIP routing.

#### 6.6. A text media subtype for support of multi-party rtt

Indicating a specific text media subtype in SDP is a straightforward way for negotiating multi-party capability. Especially if there are format differences from the "text/red" and "text/t140" formats of RFC4103 [RFC4103], then this is a natural way to do the negotiation for multi-party rtt.

Pros:

No extra efforts if a new format is needed anyway.

Cons:

None specific to using the format indication for negotiation of multi-party capability. But only feasible if a new format is needed anyway.

#### 6.7. Preferred capability declaration method for RTP-based transport.

If the preferred transport method is one with a specific media subtype in sdp, then specification by media subtype is preferred.

If this would not be the case, then the preferred capability declaration method would be the one with a simplified SDP attribute "a=rtt-mix" Section 6.4 because it is straightforward and partially usable also for WebRTC if so needed.

#### 6.8. Identification of the source of text for RTP-based solutions

The main way to identify the source of text in the RTP based solution is by the SSRC of the sending participant. In the RTP-mixer solution, this SSRC is included in the CSRC list of the transmitted packets. Further identification that may be needed for better labelling of received text may be achieved from a number of sources. It may be the RTCP SDES CNAME and NAME reports, and in the conference notification data (RFC 4575) [RFC4575].

As soon as a new member is added to the RTP session, its characteristics should be transmitted in RTCP SDES CNAME and NAME reports according to section 6.5 in RFC 3550 [RFC3550]. The information about the participant should also be included in the conference data including the text media member in a notification according to RFC 4575 [RFC4575].

The RTCP SDES report, SHOULD contain identification of the source represented by the SSRC/CSRC identifier. This identification MUST contain the CNAME field and MAY contain the NAME field and other defined fields of the SDES report.

A focus UA SHOULD primarily convey SDES information received from the sources of the session members. When such information is not available, the focus UA SHOULD compose SSRC/CSRC, CNAME and NAME information from available information from the SIP session with the participant.

### 7. RTT bridging in WebRTC

Within WebRTC, real-time text is specified to be carried in WebRTC data channels as specified in [I-D.ietf-mmusic-t140-usage-data-channel]. A few ways to handle multi-party RTT are mentioned briefly. They are repeated below.

#### 7.1. RTT bridging in WebRTC with one data channel per source

A straightforward way to handle multi-party RTT is for the bridge to open one T.140 data channel per source towards the receiving participants.

The stream-id forms a unique stream identification.

The identification of the source is made through the Label property of the channel, and session information belonging to the source. The endpoint can compose a readable label for the presentation from this information.

Pros:

This is a straightforward solution.

The load per source is low.

Cons:

With a high number of participants, the overhead of establishing and maintaining the high number of data channels required may be high, even if the load per channel is low.

## 7.2. RTT bridging in WebRTC with one common data channel

A way to handle multi-party RTT in WebRTC is for the bridge combine text from all sources into one data channel and insert the sources in the stream by a T.140 control code for source.

This method is described in a corresponding section for RTP transmission above in Section 4.1.1.9.

The identification of the source is made through insertion in the beginning of each text transmission from a source of a control code extension "c" followed by a string representing the source, framed by the control code start and end flags SOS and ST (See ITU-T T.140 [T140]).

A receiving endpoint is supposed to separate text items from the different sources and identify and display them in a suitable way.

The endpoint does not always display the source identification in the received text at the place where it is received, but has the information as a guide for planning the presentation of received text. A label corresponding to the source identification is presented when needed depending on the selected presentation style.

Pros:

This solution has relatively low overhead on session and network level

Cons:

This solution has higher overhead on the media contents level than the WebRTC solution above.

Standardisation of the new control code "c" in ITU-T T.140 [T140] is required.

The conference server need to be allowed to decrypt/encrypt the data channel contents.

### 7.3. Preferred rtt multi-party method for WebRTC

For WebRTC, one method is to prefer because of the simplicity. So, for WebRTC, the method to implement for multi-party RTT with multi-party aware parties when no other method is explicitly agreed between implementing parties is: "RTT bridging in WebRTC with one data channel per source" Section 7.1.

## 8. Presentation of multi-party text

All session participants with RTP based transport MUST observe the SSRC/CSRC field of incoming text RTP packets, and make note of which source they came from in order to be able to present text in a way that makes it easy to read text from each participant in a session, and get information about the source of the text.

In the WebRTC case, the Label parameter and other provided endpoint information should be used for the same purpose.

### 8.1. Associating identities with text streams

A source identity SHOULD be composed from available information sources and displayed together with the text as indicated in ITU-T T.140 Appendix[T140].

The source identity should primarily be the NAME field from incoming SDES packets. If this information is not available, and the session is a two-party session, then the T.140 source identity SHOULD be composed from the SIP session participant information. For multi-party sessions the source identity may be composed by local information if sufficient information is not available in the session.

Applications may abbreviate the presented source identity to a suitable form for the available display.

Applications may also replace received source information with internally used nicknames.

## 8.2. Presentation details for multi-party aware endpoints.

The multi-party aware endpoint should after any action for recovery of data from lost packets, separate the incoming streams and present them according to the style that the receiving application supports and the user has selected. The decisions taken for presentation of the multi-party interchange shall be purely on the receiving side. The sending application must not insert any item in the stream to influence presentation that is not requested by the sending participant.

### 8.2.1. Bubble style presentation

One often used style is to present real-time text in chunks in readable bubbles identified by labels containing names of sources. Bubbles are placed in one column in the presentation area and are closed and moved upwards in the presentation area after certain items or events, when there is also newer text from another source that would go into a new bubble. The text items that allows bubble closing are any character closing a phrase or sentence followed by a space or a timeout of a suitable time (about 10 seconds).

Real-time active text sent from the local user should be presented in a separate area. When there is a reason to close a bubble from the local user, the bubble should be placed above all real-time active bubbles, so that the time order that real-time text entries were completed is visible.

Scrolling is usually provided for viewing of recent or older text. When scrolling is done to an earlier point in the text, the presentation shall not move the scroll position by new received text. It must be the decision of the local user to return to automatic viewing of latest text actions. It may be useful with an indication that there is new text to read after scrolling to an earlier position has been activated.

The presentation area may become too small to present all text in all real-time active bubbles. Various techniques can be applied to provide a good overview and good reading opportunity even in such situations. The active real-time bubble may have a limited number of lines and if their contents need more lines, then a scrolling opportunity within the real-time active bubble is provided. Another method can be to only show the label and the last line of the active real-time bubble contents, and make it possible to expand or compress the bubble presentation between full view and one line view.

Erasures require special consideration. Erasure within a real-time active bubble is straightforward. But if erasure from one participant affects the last character before a bubble, the whole previous bubble becomes the actual bubble for real-time action by that participant and is placed below all other bubbles in the presentation area. If the border between bubbles was caused by the CRLF characters (instead of the normal "Line Separator"), only one erasure action is required to erase this bubble border. When a bubble is closed, it is moved up, above all real-time active bubbles.

A three-party view is shown in this example .

	^
	-
[Alice] Hi, Alice here.	
[Bob] Bob as well.	
[Eve] Hi, this is Eve, calling from Paris. I thought you should be here.	
[Alice] I am coming on Thursday, my performance is not until Friday morning.	
[Bob] And I on Wednesday evening.	
[Alice] Can we meet on Thursday evening?	
[Eve] Yes, definitely. How about 7pm. at the entrance of the restaurant Le Lion Blanc?	
[Eve] we can have dinner and then take a walk	
<Eve-typing> But I need to be back to the hotel by 11 because I need	
<Bob-typing> I wou	-
of course, I underst	v

Figure 1: Three-party call with bubble style.

Figure 1: Example of a three-party call presented in the bubble style.

### 8.2.2. Other presentation styles

Other presentation styles than the bubble style may be arranged and appreciated by the users. In a video conference one way may be to have a real-time text area below the video view of each participant. Another view may be to provide one column in a presentation area for each participant and place the text entries in a relative vertical position corresponding to when text entry in them was completed. The labels can then be placed in the column header. The considerations for ending and moving and erasure of entered text discussed above for the bubble style are valid also for these styles.

This figure shows how a coordinated column view MAY be presented.

Bob	Eve	Alice
My flight is to Orly		I will arrive by TGV. Convenient to the main station.
Eve, will you do your presentation on Friday?	Hi all, can we plan for the seminar?	
Fine, wo	Yes, Friday at 10.	We need to meet befo

Figure 2: A coordinated column-view of a three-party session with entries ordered in approximate time-order.

## 9. Presentation details for multi-party unaware endpoints.

Multi-party unaware endpoints are prepared only for presentation of two sources of text, the local user and a remote user. If mixing for multi-party unaware endpoints is to be supported, in order to enable some multi-party communication with such endpoint, the mixer need to plan the presentation and insert labels and line breaks before labels. Many limitations appear for this presentation mode, and it must be seen as a fallback and a last resort.

A procedure for presenting RTT to a conference-unaware endpoint is included in [I-D.ietf-avtcore-multi-party-rtt-mix]

## 10. Security Considerations

The security considerations valid for RFC 4103 [RFC4103] and RFC 3550 [RFC3550] are valid also for the multi-party sessions with text.

## 11. IANA Considerations

The items for indication and negotiation of capability for multi-party rtt should be registered with IANA in the specifications where they are specified in detail.

## 12. Congestion considerations

The congestion considerations described in RFC 4103 [RFC4103] are valid also for the recommended RTP-based multi-party use of the real-time text transport. A risk for congestion may appear if a number of conference participants are active transmitting text simultaneously, because the recommended RTP-based multi-party transmission method does not allow multiple sources of text to contribute to the same packet.

In situations of risk for congestion, the Focus UA MAY combine packets from the same source to increase the transmission interval per source up to one second. Local conference policy in the Focus UA may be used to decide which streams shall be selected for such transmission frequency reduction.

## 13. Acknowledgements

Arnoud van Wijk for contributions to an earlier, expired draft of this memo.

## 14. Change history

### 14.1. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-02

Added detail in the section on RTP translator model alternative 4.1.2.1.

### 14.2. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-01

Added three more methods for RTP-mixer mixing. Two RFC 5109 FEC based and another with modified data header to detect source of completely lost text.

Separated RTP-based and WebRTC based solutions.

Deleted the multi-party-unaware mixing procedure appendix. It is now included in the draft draft-ietf-avtcore-multi-party-rtt-mix. Kept a section with a reference to the new place.

#### 14.3. Changes from draft-hellstrom-mmusic-multi-party-rtt-02 to draft-hellstrom-avtcore-multi-party-rtt-solutions-00

Add discussion about switching performance, as discussed in avtcore on March 13.

Added that a decrease of transmission interval to 100 ms increases switching performance by a factor 3, but still not sufficient.

Added that the CSRC-list method also uses 100 milliseconds transmission interval.

Added the method with multiple primary text in each packet.

Added the timestamp-based method for rtp-mixing proposed by James Hamlin on March 14.

Corrected the chat style presentation example picture. Delete a few "[mix]".

#### 14.4. Changes from version draft-hellstrom-mmusic-multi-party-rtt-01 to -02

Change from a general overview to overview with clear recommendations.

Splits text coordination methods in three groups.

Recommends rtt-mixer with sources in CSRC-list but referenes to its spec for details.

Shortened Appendix with conference-unaware example.

Cleaned up preferences.

Inserted pictures of screen-views.

### 15. References

#### 15.1. 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>>.

#### 15.2. Informative References

- [EN301549] ETSI, "EN 301 549. Accessibility requirements for ICT products and services", November 2019, <[https://www.etsi.org/deliver/etsi\\_en/301500\\_301599/301549/03.01.01\\_60/en\\_301549v030101p.pdf](https://www.etsi.org/deliver/etsi_en/301500_301599/301549/03.01.01_60/en_301549v030101p.pdf)>.
- [I-D.ietf-avtcore-multi-party-rtt-mix] Hellstrom, G., "RTP-mixer formatting of multi-party Real-time text", Work in Progress, Internet-Draft, draft-ietf-avtcore-multi-party-rtt-mix-06, 11 June 2020, <<https://tools.ietf.org/html/draft-ietf-avtcore-multi-party-rtt-mix-06>>.
- [I-D.ietf-avtcore-multiplex-guidelines] Westerlund, M., Burman, B., Perkins, C., Alvestrand, H., and R. Even, "Guidelines for using the Multiplexing Features of RTP to Support Multiple Media Streams", Work in Progress, Internet-Draft, draft-ietf-avtcore-multiplex-guidelines-12, 16 June 2020, <<https://tools.ietf.org/html/draft-ietf-avtcore-multiplex-guidelines-12>>.
- [I-D.ietf-mmusic-t140-usage-data-channel] Holmberg, C. and G. Hellstrom, "T.140 Real-time Text Conversation over WebRTC Data Channels", Work in Progress, Internet-Draft, draft-ietf-mmusic-t140-usage-data-channel-14, 10 April 2020, <<https://tools.ietf.org/html/draft-ietf-mmusic-t140-usage-data-channel-14>>.
- [I-D.ietf-perc-private-media-framework] Jones, P., Benham, D., and C. Groves, "A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing (PERC)", Work in Progress, Internet-Draft, draft-ietf-perc-private-media-framework-12, 5 June 2019, <<https://tools.ietf.org/html/draft-ietf-perc-private-media-framework-12>>.
- [NENAi3] NENA, "NENA-STA-010.2-2016. Detailed Functional and Interface Standards for the NENA i3 Solution", October 2016, <[https://www.nena.org/page/i3\\_Stage3](https://www.nena.org/page/i3_Stage3)>.
- [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>>.

- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <<https://www.rfc-editor.org/info/rfc3261>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.
- [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>>.
- [RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, DOI 10.17487/RFC3840, August 2004, <<https://www.rfc-editor.org/info/rfc3840>>.
- [RFC3841] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Caller Preferences for the Session Initiation Protocol (SIP)", RFC 3841, DOI 10.17487/RFC3841, August 2004, <<https://www.rfc-editor.org/info/rfc3841>>.
- [RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <<https://www.rfc-editor.org/info/rfc4103>>.
- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, DOI 10.17487/RFC4353, February 2006, <<https://www.rfc-editor.org/info/rfc4353>>.
- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, Ed., "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, DOI 10.17487/RFC4575, August 2006, <<https://www.rfc-editor.org/info/rfc4575>>.
- [RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, DOI 10.17487/RFC4579, August 2006, <<https://www.rfc-editor.org/info/rfc4579>>.

- [RFC4597] Even, R. and N. Ismail, "Conferencing Scenarios", RFC 4597, DOI 10.17487/RFC4597, August 2006, <<https://www.rfc-editor.org/info/rfc4597>>.
- [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>>.
- [RFC5194] van Wijk, A., Ed. and G. Gybels, Ed., "Framework for Real-Time Text over IP Using the Session Initiation Protocol (SIP)", RFC 5194, DOI 10.17487/RFC5194, June 2008, <<https://www.rfc-editor.org/info/rfc5194>>.
- [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>>.
- [RFC6443] Rosen, B., Schulzrinne, H., Polk, J., and A. Newton, "Framework for Emergency Calling Using Internet Multimedia", RFC 6443, DOI 10.17487/RFC6443, December 2011, <<https://www.rfc-editor.org/info/rfc6443>>.
- [RFC6881] Rosen, B. and J. Polk, "Best Current Practice for Communications Services in Support of Emergency Calling", BCP 181, RFC 6881, DOI 10.17487/RFC6881, March 2013, <<https://www.rfc-editor.org/info/rfc6881>>.
- [RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <<https://www.rfc-editor.org/info/rfc7667>>.
- [T140] ITU-T, "Recommendation ITU-T T.140 (02/1998), Protocol for multimedia application text conversation", February 1998, <<https://www.itu.int/rec/T-REC-T.140-199802-I/en>>.
- [T140ad1] ITU-T, "Recommendation ITU-T.140 Addendum 1 - (02/2000), Protocol for multimedia application text conversation", February 2000, <<https://www.itu.int/rec/T-REC-T.140-200002-I!Add1/en>>.
- [TS103479] ETSI, "TS 103 479. Emergency communications (EMTEL); Core elements for network independent access to emergency services", December 2019, <[https://www.etsi.org/deliver/etsi\\_ts/103400\\_103499/103479/01.01.01\\_60/ts\\_103479v010101p.pdf](https://www.etsi.org/deliver/etsi_ts/103400_103499/103479/01.01.01_60/ts_103479v010101p.pdf)>.

- [TS22173] 3GPP, "IP Multimedia Core Network Subsystem (IMS) Multimedia Telephony Service and supplementary services; Stage 1", 3GPP TS 22.173 17.1.0, 20 December 2019, <<http://www.3gpp.org/ftp/Specs/html-info/22173.htm>>.
- [TS24147] 3GPP, "Conferencing using the IP Multimedia (IM) Core Network (CN) subsystem; Stage 3", 3GPP TS 24.147 16.0.0, 19 December 2019, <<http://www.3gpp.org/ftp/Specs/html-info/24147.htm>>.

## Author's Address

Gunnar Hellstrom  
Gunnar Hellstrom Accessible Communication  
Esplanaden 30  
SE-136 70 Vendelso  
Sweden

Phone: +46 708 204 288  
Email: [gunnar.hellstrom@ghaccess.se](mailto:gunnar.hellstrom@ghaccess.se)

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: 3 July 2022

G. Hellstrom  
GHAccess  
30 December 2021

Real-time text solutions for multi-party sessions  
draft-hellstrom-avtcore-multi-party-rtt-solutions-08

Abstract

This document specifies methods for Real-Time Text (RTT) media handling in multi-party calls. The main discussed transport is to carry Real-Time text by the RTP protocol in a time-sampled mode according to RFC 4103. The mechanisms enable the receiving application to present the received real-time text media, separated per source, in different ways according to user preferences. Some presentation related features are also described explaining suitable variations of transmission and presentation of text.

Call control features are described for the SIP environment. A number of alternative methods for providing the multi-party negotiation, transmission and presentation are discussed and a recommendation for the main ones is provided. The main solution for SIP based centralized multi-party handling of real-time text is achieved through a media control unit coordinating multiple RTP text streams into one RTP stream.

Alternative methods using a single RTP stream and source identification inline in the text stream are also described, one of them being provided as a lower functionality fallback method for endpoints with no multi-party awareness for RTT.

Bridging methods where the text stream is carried without the contents being dealt with in detail by the bridge are also discussed.

Brief information is also provided for multi-party RTT in the WebRTC environment.

The intention is to provide background for decisions, specification and implementation of selected methods. The recommendations have resulted in the published RFC 9071. This document is maintained mainly to present the reasoning behind the recommendations and provide material for any further work in other application areas.

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 <https://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 3 July 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

- 1. Introduction . . . . . 4
  - 1.1. Requirements Language . . . . . 5
- 2. Centralized conference model . . . . . 5
- 3. Requirements on multi-party RTT . . . . . 6
  - 3.1. General requirements . . . . . 6
  - 3.2. Performance requirements . . . . . 7
- 4. RTP based solutions . . . . . 8
  - 4.1. Coordination of text RTP streams . . . . . 8
    - 4.1.1. RTP-based solutions with a central mixer . . . . . 9
      - 4.1.1.1. RTP Mixer using default RFC 4103 methods . . . . . 9
      - 4.1.1.2. RTP Mixer using the default method but decreased transmission interval . . . . . 9
      - 4.1.1.3. RTP Mixer with frequent transmission and indicating sources in CSRC-list . . . . . 10

- 4.1.1.4. RTP Mixer interleaving packets, receiver using timestamp to recover from loss . . . . . 12
- 4.1.1.5. RTP Mixer with multiple primary data in each packet and individual sequence numbers . . . . . 13
- 4.1.1.6. RTP Mixer with multiple primary data in each packet . . . . . 14
- 4.1.1.7. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy in the packets . . . . . 15
- 4.1.1.8. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy and separate sequence number in the packets . . . . 17
- 4.1.1.9. RTP Mixer indicating participants by a control code in the stream . . . . . 19
- 4.1.1.10. Mixing for multi-party unaware user agents . . . 21
- 4.1.2. RTP-based bridging with minor RTT media contents reformatting by the bridge . . . . . 22
  - 4.1.2.1. One RTP stream for RTT per participant from the mixer . . . . . 22
  - 4.1.2.2. Selective Forwarding Middlebox . . . . . 25
  - 4.1.2.3. Distributing packets in an end-to-end encryption structure . . . . . 27
  - 4.1.2.4. Mesh of RTP endpoints . . . . . 27
  - 4.1.2.5. Multiple RTP sessions, one for each participant . . . . . 28
- 5. Preferred RTP-based multi-party RTT transport method . . . . 29
- 6. Session control of RTP-based multi-party RTT sessions . . . . 29
  - 6.1. Implicit RTT multi-party capability indication . . . . . 30
  - 6.2. RTT multi-party capability declared by SIP media-tags . . 31
  - 6.3. SDP media attribute for RTT multi-party capability indication . . . . . 32
  - 6.4. Simplified SDP media attribute for RTT multi-party capability indication . . . . . 33
  - 6.5. SDP format parameter for RTT multi-party capability indication . . . . . 34
  - 6.6. A text media subtype for support of multi-party rtt . . . 35
  - 6.7. Preferred capability declaration method for RTP-based transport. . . . . 35
  - 6.8. Identification of the source of text for RTP-based solutions . . . . . 36
- 7. RTT bridging in WebRTC . . . . . 36
  - 7.1. RTT bridging in WebRTC with one data channel per source . . . . . 36
- 8. Presentation of multi-party text . . . . . 37
  - 8.1. Associating identities with text streams . . . . . 37
  - 8.2. Presentation details for multi-party aware endpoints. . . 38
    - 8.2.1. Bubble style presentation . . . . . 38
    - 8.2.2. Other presentation styles . . . . . 40
- 9. Presentation details for multi-party unaware endpoints. . . . 41
- 10. Security Considerations . . . . . 41

- 11. IANA Considerations . . . . . 41
- 12. Congestion considerations . . . . . 41
- 13. Acknowledgements . . . . . 42
- 14. Change history . . . . . 42
  - 14.1. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-08 . 42
  - 14.2. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-07 . 42
  - 14.3. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-06 . 42
  - 14.4. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-05 . 42
  - 14.5. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-04 . 42
  - 14.6. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-03 . 42
  - 14.7. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-02 . 43
  - 14.8. Changes to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-01 . 43
  - 14.9. Changes from draft-hellstrom-mmusic-multi-party-rtt-02 to
    - draft-hellstrom-avtcore-multi-party-rtt-solutions-00 . 43
  - 14.10. Changes from version
    - draft-hellstrom-mmusic-multi-party-rtt-01 to -02 . . . 43
- 15. References . . . . . 44
  - 15.1. Normative References . . . . . 44
  - 15.2. Informative References . . . . . 44
- Author's Address . . . . . 47

1. Introduction

Real-time text (RTT) is a medium in real-time conversational sessions. Text entered by participants in a session is transmitted in a time-sampled fashion, so that no specific user action is needed to cause transmission. This gives a direct flow of text in the rate it is created, that is suitable in a real-time conversational setting. The real-time text medium can be combined with other media in multimedia sessions.

Media from a number of multimedia session participants can be combined in a multi-party session. The present document specifies how the real-time text streams can be handled in multi-party sessions. Recommendations are provided for preferred methods. The intention is to provide background for decisions, specification and implementation of selected methods. The recommendations have resulted in the published RFC 9071. This document is maintained mainly to present the reasoning behind the recommendations and provide material for any further work in other application areas.

The description is mainly focused on the transport level, but also describes a few session and presentation level aspects.

Transport of real-time text is specified in RFC 4103 [RFC4103] RTP Payload for text conversation. It makes use of RFC 3550 [RFC3550] Real Time Protocol, for transport. Robustness against network transmission problems is normally achieved through redundant transmission based on the principle from RFC 2198 [RFC2198], with one primary and two redundant transmission of each text element. Primary and redundant transmissions are combined in packets and described by a redundancy header. This transport is usually used in the SIP Session Initiation Protocol RFC 3261 [RFC3261] environment.

A very brief overview of functions for real-time text handling in multi-party sessions is described in RFC 4597 [RFC4597] Conferencing Scenarios, sections 4.8 and 4.10. The present specification builds on that description and indicates which protocol mechanisms should be used to implement multi-party handling of real-time text.

Real-time text can also be transported in the WebRTC environment, by using WebRTC data channels according to [RFC8865]. Multi-party aspects for WebRTC solutions are briefly covered.

### 1.1. Requirements Language

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

## 2. Centralized conference model

In the centralized conference model for SIP, introduced in RFC 4353 [RFC4353] "A Framework for Conferencing with the Session Initiation Protocol (SIP)", one function co-ordinates the communication with participants in the multi-party session. This function also controls media mixer functions for the media appearing in the session. The central function is common for control of all media, while the media mixers may work differently for each media.

The central function is called the Focus UA. Many variants exist for setting up sessions including the multipoint control centre. It is not within scope of this description to describe these, but rather the media specific handling in the mixer required to handle multi-party calls with RTT.

The main principle for handling real-time text media in a centralized conference is that one RTP session for real-time text is established including the multipoint media control centre and the participating endpoints which are going to have real-time text exchange with the others.

The different possible mechanisms for mixing and transporting RTT differs in the way they multiplex the text streams and how they identify the sources of the streams. RFC 7667 [RFC7667] describes a number of possible use cases for RTP. This specification refers to different sections of RFC 7667 for further reading of the situations caused by the different possible design choices.

The recommended method for using RTP based RTT in a centralized conference model is specified in [RFC9071] based on the recommendations in this document.

Real-time text can also be transported in the WebRTC environment, by using WebRTC data channels according to [RFC8865]. Ways to handle multi-party calls in that environment are also specified.

### 3. Requirements on multi-party RTT

#### 3.1. General requirements

The following general requirements are placed on multi-party RTT:

A solution shall be applicable to IMS (3GPP TS 22.173) [TS22173], SIP based VoIP [RFC5194] and Next Generation Emergency Services (NENA i3 [NENAi3], ETSI TS 103 479 [TS103479], RFC 6443 [RFC6443]).

The transmission interval for text should not be longer than 500 milliseconds when there is anything available to send. Ref ITU-T T.140 [T140].

If text loss is detected or suspected, a missing text marker should be inserted in the text stream. Ref ITU-T T.140 Amendment 1 [T140ad1]. ETSI EN 301 549 [EN301549]

The display of text from the members of the conversation shall be arranged so that the text from each participant is clearly readable, and its source and the relative timing of entered text is visualized in the display. Mechanisms for looking back in the contents from the current session should be provided. The text should be displayed as soon as it is received. Ref ITU-T T.140 [T140]

Bridges must be multimedia capable (voice, video, text). Ref NENA i3 STA-010.3. [NENAI3]

It MUST be possible to use real-time text in conferences both as a medium of discussion between individual participants (for example, for sidebar discussions in real-time text while listening to the main conference audio) and for central support of the conference with real-time text interpretation of speech. Ref (R7) in RFC 5194. [RFC5194]

It should be possible to protect RTT contents with usual means for privacy and integrity. Ref RFC 6881 section 16. [RFC6881]

Conferencing procedures are documented in RFC 4579 [RFC4579]. Ref NENA i3 STA-010.3. [NENAI3]

Conferencing applies to any kind of media stream by which users may want to communicate. Ref 3GPP TS 24.147 [TS24147]

The framework for SIP conferences is specified in RFC 4353 [RFC4353]. Ref 3GPP TS 24.147 [TS24147]

### 3.2. Performance requirements

The mixer performance requirements can be expressed in one number, extracted from the user requirements on real-time text expressed in ITU-T F.700, where it is stated that for "good" usability, text characters should not be delayed more than 1 second from creation to presentation. For "usable" usability the figure is 2 seconds. The main factor behind these limits is from when taking turns in a conversation gets disturbed by a delay of when a response gets visible to the receiving part. If that times get too long, the receiving part gets unsure if the previous utterance was well perceived and the receiving part maybe prepares for repetition. This is similar to the same effect in voice communication, where the usability limit is 400 ms delay.

Another important factor in a multi-party conference is the opportunity for a participant using real-time text to provide timely comments and get a chance to enter the discussion if the majority of participants use voice in the conference. A complicating factor when stating the requirements is that some transport methods do not cause a total delay, but instead an increasing jerkiness when the number of simultaneously sending participants is increased.

It should however be remembered that the expected number of participants sending real-time text simultaneously is low. Just as with voice or sign language, the capability of the participants to

perceive utterances from more than one participant at a time is very limited. Therefore the normal case in multi-party situations is that one participant at a time is the main provider of text. Others might usually just provide very brief comments such as "yes" or "no" or "may I comment?". Only at very rare situations two participants provide more information simultaneously.

- \* The number of expected simultaneously transmitting users is different for different applications. In all cases, just one transmitting user is the normal case. Two simultaneously transmitting participants can occasionally be expected in emergency services, relay services, small unmanaged conferences and group calls and large managed conferences. Three simultaneously transmitting participants may appear occasionally in large unmanaged conferences. The following can therefore express the performance requirement.
- \* The mean delay of text passing the mixer introduced when only one participant is sending text should be kept to a minimum and should not be more than 400 ms.
- \* The mean delay of text passing the mixer should not be more than 1 second during moments when up to three users are sending text simultaneously.
- \* For the very rare case that more than three participants send text simultaneously, the mixer may take action to limit the introduced delay of the text passing the mixer to 7 seconds e.g. by discarding text from some participants and instead inserting a general warning about possible text loss in the stream.
- \* The load on network and nodes should be limited. This is usually achieved by setting a limit for how many packets per second that may be sent from a mixer to each participant. While two-party use by RFC 4103, limits the load to 3.3 packets per second, a realistic limit for mixers could be 10 packets per second. This is still just a small fraction of what is commonly transmitted in real-time video and audio, so in known environments it may be possible to increase the packet rate if needed to keep latency low.

#### 4. RTP based solutions

##### 4.1. Coordination of text RTP streams

Coordinating and sending text RTP streams in the multi-party session can be done in a number of ways. The most suitable methods are specified here with pros and cons.

A receiving and presenting endpoint MUST separate text from the different sources and identify and display them accordingly.

#### 4.1.1. RTP-based solutions with a central mixer

A set of solutions can be based on the central RTP mixer. They are described here and a preferred method selected.

##### 4.1.1.1. RTP Mixer using default RFC 4103 methods

Without any extra specifications, a mixer would transmit with 300 milliseconds intervals, and use RFC 4103 [RFC4103] with the default redundancy of one original and two redundant transmissions. The source of the text would be indicated by a single member in the CSRC list. Text from different sources cannot be transmitted in the same packet. Therefore, from the time when the mixer sent one piece of new text from one source, it will need to transmit that text again twice as redundant data, before it can send text from another source. The jerkiness = time between transmission of new text is 900 ms. This is clearly insufficient.

Pros:

Only a capability negotiation method is needed. No other update of standards are needed, just a general remark that traditional RTP-mixing is used.

Cons:

Clearly insufficient mixer switching performance.

A bit complex handling of transmission when there is new text available from more than one source. The mixer needs to send two packets more with redundant text from the current source before starting to send anything from the other source.

##### 4.1.1.2. RTP Mixer using the default method but decreased transmission interval

This method makes use of the default RTP-mixing method briefly described in Section 4.1.1.1. The only difference is that the transmission interval is decreased to 100 milliseconds when there is text from more than one source available for transmission. The jerkiness is 300 ms. The mean delay with two simultaneously sending participants is 250 ms, and with three simultaneously sending participants 500 ms. This is acceptable performance.

Pros:

Minor influence on standards

Can be relatively rapidly be introduced in the intended technical environments.

Can be declared in sdp as the already existing "text/red" format with a multi-party attribute for capability negotiation.

Cons:

The introduced jerkiness of new text from more than the required three simultaneously sending sources is high.

Slightly higher risk for loss of text at bursty packet loss than for the recommended transmission interval (300 ms) for RFC 4103.

When complete loss of packets occur (beyond recovery), it is not possible to deduce from which source text was lost.

A bit complex handling of transmission when there is new text available from more than one source. The mixer needs to send two packets more with redundant text from the current source before starting to send anything from the other source.

#### 4.1.1.3. RTP Mixer with frequent transmission and indicating sources in CSRC-list

An RTP media mixer combines text from participants into one RTP stream, thus all using the same destination address/port combination, the same RTP SSRC, and one sequence number series as described in Section 7.1 and 7.3 of RTP RFC 3550 [RFC3550] about the Mixer function. This method is also briefly described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

The sources of the text in each RTP packet are identified by the CSRC list in the RTP packets, containing the SSRC of the initial sources of text. The order of the CSRC parameters is with the SSRC of the source of the primary text first, followed by the SSRC of the first level redundancy, and then the second level redundancy.

The transmission interval should be 100 milliseconds when there is text to transmit from more than one source, and otherwise 300 ms.

The identification of the sources is made through the CSRC fields and can be made more readable at the receiver through the RTCP SDES CNAME and NAME packets as described in RTP [RFC3550].

Information provided through the notification according to RFC 4575 [RFC4575] when the participant joined the conference provides also suitable information and a reference to the SSRC.

A receiving endpoint is supposed to separate text items from the different sources and identify and display them accordingly.

The ordered CSRC lists in the RFC 4103 [RFC4103] packets make it possible to recover from loss of one and two packets in sequence and assign the recovered text to the right source. For more loss, a marker for possible loss should be inserted or presented.

The conference server needs to have authority to decrypt the payload in the received RTP packets in order to be able to recover text from redundant data or insert the missing text marker in the stream, and repack the text in new packets.

Even if the format is very similar to "text/red" of RFC 4103, it needs to be declared as a new media subtype, e.g. "text/rex".

#### Pros:

This method has low overhead and less complexity than the methods in Section 4.1.1.1, Section 4.1.1.2, Section 4.1.1.4 and Section 4.1.1.6.

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream (normally primary and two redundant levels).

This method can be implemented with most RTP implementations.

The source switching performance is sufficient for well-behaving conference participants. The jerkiness is 100 ms.

#### Cons:

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduce the sources of the totally lost data.

Slightly higher risk for loss of text at bursty packet loss than for the recommended transmission interval for RFC 4103.

Requires a different sub media format, e.g. "text/rex". This takes a long time in standardisation and releases of target technical environments.

The conference server needs to be allowed to decrypt/encrypt the packet payload. This is however normal for media mixers for other media.

#### 4.1.1.4. RTP Mixer interleaving packets, receiver using timestamp to recover from loss

This method has text only from one source per packet, as the original RFC 4103 [RFC4103] specifies. Packets with text from different sources are instead allowed to be interleaved. The recovery procedure in the receiver makes use of the RTP timestamp and timestamp offsets in the redundancy headers to evaluate if a piece of redundant data was received earlier or not as a base for decision if the redundant data should be recovered or not in case of packet loss.

In this method, the transmission is immediate when new text from a source is available for transmission. Otherwise the transmission interval for redundant transmission of text from each source is 320 ms when no new text is available. At congestion, the transmission interval is allowed to be longer.

##### Pros:

The format of each packet is equal to what is specified in RFC 4103 [RFC4103].

The source switching performance is sufficient and good. Text from five participants can be transmitted simultaneously with 300 milliseconds interval per source.

New text from five simultaneous sources can be transmitted within 300 milliseconds. This is sufficient.

Recovery from packet loss with five simultaneous sources takes 1 second. This is good and implies good protection against bursty packet loss causing resulting text loss.

##### Cons:

The recovery time in case of packet loss can be long with more than ten simultaneously intensively sending participants. Then it will be more than 2 seconds.

The recovery procedure is different from what is described in RFC 4103 [RFC4103].

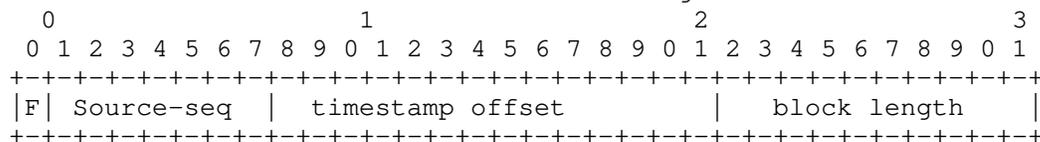
It will in many cases of loss of multiple packets not be possible to deduce if there was any resulting loss of text. A mark for possible loss should be inserted in cases when there might have been resulting loss.

Because of the benefits, this method is preferred and standardised as [RFC9071]

4.1.1.5. RTP Mixer with multiple primary data in each packet and individual sequence numbers

This method allows primary as well as redundant text from more than one source per packet. The packet payload contains an ordered set of redundant and primary data with the same number of generations of redundancy as once agreed in the SDP negotiation. The data header reflects these parts of the payload. The CSRC list contains one CSRC member per source in the payload and in the same order. An individual sequence number per source is included in the data header replacing the t140 payload type number that is instead assumed to be constant in this format. This allows an individual extra sequence number per source with maximum value 127, suitable for checking for which source loss of text appeared when recovery was not possible.

The data header would contain the following fields:



Where "Source-seq" is the sequence number per source.

The maximum number of members in the CSRC-list is 15, and that is therefore the maximum number of sources that can be represented in each packet provided that all data can be fitted into the size allowable in one packet.

Transmission is done as soon as there is new text available, but not with shorter interval than 150 ms and not longer than 300 ms while there is anything to send.

A new media subtype is needed, e.g. "text/rex".

This is an SDP offer example for both traditional "text/red" and multi-party "text/rex" format:

```
m=text 11000 RTP/AVP 101 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=rtpmap:101 rex/1000
a=fmtp:100 98/98/98
a=fmtp:101 98/98/98
```

Pros:

The source switching performance is good. Text from 15 participants can be transmitted simultaneously.

New text from 15 simultaneous sources can be transmitted within 300 milliseconds. This is good performance.

When more consecutive packet loss than the number of generations of redundant data appears, it is still possible to deduce the sources of the totally lost data, when next text from these sources arrive.

Cons:

The format of each packet is different from what is specified in RFC 4103 [RFC4103].

The processing time in standard organisations will be long.

A new media subtype is needed, causing a bit complex negotiation.

The recovery procedure is a bit complex.

#### 4.1.1.6. RTP Mixer with multiple primary data in each packet

This method allows primary as well as redundant text from more than one source per packet. The packet payload contains an ordered set of redundant and primary data with the same number of generations of redundancy as once agreed in the SDP negotiation. The data header reflects these parts of the payload. The CSRC list contains one CSRC member per source in the payload and in the same order.

The maximum number of members in the CSRC-list is 15, and that is therefore the maximum number of sources that can be represented in each packet provided that all data can be fitted into the size allowable in one packet.

Transmission is done as soon as there is new text available, but not with shorter interval than 150 ms and not longer than 300 ms while there is anything to send.

A new media subtype is needed, e.g. "text/rex".

SDP would be the same as in Section 4.1.1.6.

Pros:

The source switching performance is good. Text from 15 participants can be transmitted simultaneously.

New text from 15 simultaneous sources can be transmitted within 150 milliseconds. This is good performance.

Cons:

The format of each packet is different from what is specified in RFC 4103 [RFC4103].

A new media subtype is needed.

A new media subtype is needed, causing a bit complex negotiation.

The processing time in standard organisation will be long.

The recovery procedure is a bit complex [RFC4103].

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduce the sources of the totally lost data.

#### 4.1.1.7. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy in the packets

This method allows primary data from one source and redundant text from other sources in each packet. The packet payload contains primary data in "text/t140" format, and redundant data in RFC 5109 FEC [RFC5109] format called "text/ulpfec". That means that the redundant data contains the sequence number and the CSRC and other characteristics from the RTP header when the data was sent as primary. The redundancy can be sent at a selected number of packets after when it was sent as primary, in order to improve the protection against bursty packet loss. The redundancy level is recommended to be the same as in original RFC 4103.

RFC 4103 says that the protection against loss can be made by other methods than plain redundancy, so this method is in line with that statement.

Transmission is done as soon as there is new text available, but not with shorter interval than 100 ms and not longer than 300 ms while there is anything to send (new or redundant text).

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduce the sources of the totally lost data.

The sdp can indicate the format as "text/red" with "text/ulpfec" redundant data in this way. with traditional RFC 4103 with "text/red" with "text/t140" as redundant data as a fallback.

```
m=text 49170 RTP/AVP 98 101 100 102
a=rtpmap:98 red/1000
a=fmtp:98 100/102/102
a=rtpmap:102 ulpfec/1000
a=rtpmap:100 t140/1000
a=rtpmap:101 red/1000
a=fmtp:101 100/100/100
a=fmtp:100 cps=200
```

The "text/ulpfec" format includes an indication of how far back the redundancy belongs, making it possible to cover bursty packet loss better than the other formats with short transmission intervals. For real-time text, it is recommended to send three packets between the primary and the redundant transmissions of text. That makes the transmission cover between 500 and 1500 ms of bursty packet loss. The variation is because of the varying packet interval between many and one simultaneously transmitting source.

The "text/ulpfec" format has a number of parameters. One is the length of the data to be protected which in this case must be the whole t140block.

Pros:

The source switching performance is good. Text from 5 participants can be transmitted within 500 ms.

Good recovery from bursty packet loss.

The method is based on existing standards. No new registrations are needed.

#### Cons:

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduce the sources of the totally lost data.

Even if the switching performance is good, it is not as good as for the method called "RTP Mixer with multiple primary data in each packet" Section 4.1.1.6. With more than 5 simultaneously sending sources, there will be a noticeable delay of text of over 500 ms, with 100 ms added per simultaneous source. This is however beyond the requirements and would be a concern only in congestion situations.

The recovery procedure is a bit complex [RFC5109].

There is more overhead in terms of extra data and extra packets sent than in the other methods. With the recommended two redundant generations of data, each packet will be 36 bytes longer than with traditional RFC 4103, and at each pause in transmission five extra packets with only redundant data will be sent compared to two extra packets for the traditional RFC 4103 case.

#### 4.1.1.8. RTP Mixer with RFC 5109 FEC and RFC 2198 redundancy and separate sequence number in the packets

This method allows primary data from one source and redundant text from other sources in each packet. The packet payload contains primary data in a new "text/t140e" format, and redundant data in RFC 5109 FEC [RFC5109] format called "text/ulpfec". That means that the redundant data contains the sequence number and the CSRC and other characteristics from the RTP header when the data was sent as primary. The redundancy can be sent at a selected number of packets after when it was sent as primary, in order to improve the protection against bursty packet loss. The redundancy level is recommended to be the same as in original RFC 4103. The "text/t140e" format contains a source-specific sequence number and the t140block.

RFC 4103 says that the protection against loss can be made by other methods than plain redundancy, so this method is in line with that statement.

Transmission is done as soon as there is new text available, but not with shorter interval than 100 ms and not longer than 300 ms while there is anything to send (new or redundant text).

When more consecutive packet loss than the number of generations of redundant data appears, it is possible to deduce which sources lost data when new data arrives from the sources. This is done by monitoring the received source specific sequence numbers preceding the text.

This is an example of how can indicate the format as "text/red" with "text/t140e" as primary and "text/ulpfec" redundant data, with traditional RFC 4103 with "text/red" with "text/t140" as redundant data as a fallback.

```
m=text 49170 RTP/AVP 98 101 100 102 103
a=rtpmap:98 red/1000
a=fmtp:98 100/102/102
a=rtpmap:102 ulpfec/1000
a=rtpmap:103 t140/1000
a=rtpmap:100 t140e/1000
a=rtpmap:101 red/1000
a=fmtp:101 103/103/103
a=fmtp:100 cps=200
```

The "text/ulpfec" format includes an indication of how far back the redundancy belongs, making it possible to cover bursty packet loss better than the other formats with short transmission intervals. For real-time text, it is recommended to send three packets between the primary and the redundant transmissions of text. That makes the transmission cover between 500 and 1500 ms of bursty packet loss. The variation is because of the varying packet interval between many and one simultaneously transmitting source.

The "text/ulpfec" format has a number of parameters. One is the length of the data to be protected which in this case must be the whole t140block.

Pros:

The source switching performance is good. Text from 5 participants can be transmitted within 500 ms.

Good recovery from bursty packet loss.

The method is based on an existing standard for FEC.

When more consecutive packet loss than the number of generations of redundant data appears, it is possible to deduce the source of the lost data when new text arrives from the source.

Cons:

Even if the switching performance is good, it is not as good as for the method called "RTP Mixer with multiple primary data in each packet" Section 4.1.1.6. With more than 5 simultaneously sending sources, there will be a noticeable delay of text of over 500 ms, with 100 ms added per simultaneous source. This is however beyond the requirements and would be a concern only in congestion situations.

The recovery procedure is a bit complex [RFC5109].

There is more overhead in terms of extra data and extra packets sent than in the other methods. With the recommended two redundant generations of data, each packet will be 40 bytes longer than with traditional RFC 4103, and at each pause in transmission five extra packets with only redundant data will be sent compared to two extra packets for the traditional RFC 4103 case.

A new text media subtype "text/t140e" needs to be registered.

The processing time in standard organisation will be long.

#### 4.1.1.9. RTP Mixer indicating participants by a control code in the stream

Text from all participants except the receiving one is transmitted from the media mixer in the same RTP session and stream, thus all using the same destination address/port combination, the same RTP SSRC and , one sequence number series as described in Section 7.1 and 7.3 of RTP RFC 3550 [RFC3550] about the Mixer function. The sources of the text in each RTP packet are identified by a new defined T.140 control code "c" followed by a unique identification of the source in UTF-8 string format.

The receiver can use the string for presenting the source of text. This method is on the RTP level described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

The inline coding of the source of text is applied in the data stream itself, and an RTP mixer function is used for coordinating the sources of text into one RTP stream.

Information uniquely identifying each user in the multi-party session is placed as the parameter value "n" in the T.140 application protocol function with the function code "c". The identifier shall thus be formatted like this: SOS c n ST, where SOS and ST are coded as specified in ITU-T T.140 [T140]. The "c" is the letter "c". The n parameter value is a string uniquely identifying the source. This parameter shall be kept short so that it can be repeated in the transmission without concerns for network load.

A receiving endpoint is supposed to separate text items from the different sources and identify and display them accordingly.

The conference server need to be allowed to decrypt/encrypt the packet payload in order to check the source and repack the text.

Pros:

If loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103]stream. (normally primary and two redundant levels.

This method can be implemented with most RTP implementations.

The method can also be used with other transports than RTP

Cons:

The method implies a moderate load by the need to insert the source often in the stream.

If more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduce the source of the totally lost data.

The mixer needs to be able to generate suitable and unique source identifications which are suitable as labels for the sources.

Requires an extension on the ITU-T T.140 standard, best made by the ITU.

There is a risk that the control code indicating the change of source is lost and the result is false source indication of text.

The conference server need to be allowed to decrypt/encrypt the packet payload.

## 4.1.1.10. Mixing for multi-party unaware user agents

Multi-party real-time text contents can be transmitted to multi-party unaware user agents if source labelling and formatting of the text is performed by a mixer. This method has the limitations that the layout of the presentation and the format of source identification is purely controlled by the mixer, and that only one source at a time is allowed to present in real-time. Other sources need to be stored temporarily waiting for an appropriate moment to switch the source of transmitted text. The mixer controls the switching of sources and inserts a source identifier in text format at the beginning of text after switch of source. The logic of the mixer to detect when a switch is appropriate should detect a number of places in text where a switch can be allowed, including new line, end of sentence, end of phrase, a period of inactivity, and a word separator after a long time of active transmission.

This method MAY be used when no support for multi-party awareness is detected in the receiving endpoint. The base for this method is described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

See [RFC9071] for a procedure for mixing RTT for a conference-unaware endpoint.

## Pros:

Can be transmitted to conference-unaware endpoints.

Can be used with other transports than RTP

## Cons:

Does not allow full real-time presentation of more than one source at a time. Text from other sources will be delayed.

The only realistic presentation format is a style with the text from the different sources presented with a text label indicating source, and the text collected in a chat style presentation but with more frequent turn-taking.

Endpoints often have their own system for adding labels to the RTT presentation. In that case there will be two levels of labels in the presentation, one for the mixer and one for the sources.

If loss of more packets than can be recovered by the redundancy appears, it is not possible to detect which source was struck by the loss. It is also possible that a source switch occurred during the loss, and therefore a false indication of the source of text can be provided to the user after such loss.

Because of all these cons, this method is not recommended be used as the main method, but only as fallback and the last resort for backwards interoperability with multi-party unaware endpoints.

The conference server need to be allowed to decrypt/encrypt the packet payload.

#### 4.1.2. RTP-based bridging with minor RTT media contents reformatting by the bridge

It may be desirable to send text in a multi-party setting in a way that allows the text stream contents to be distributed without being dealt with in detail in any central server. This approach may enable end-to-end encryption. A number of such methods are described. However, when writing this specification, no one of these methods has a specified way of establishing the session by sdp. A reference for inspiration may be [TS26114] Annex S.

##### 4.1.2.1. One RTP stream for RTT per participant from the mixer

Within the RTP session, text from each participant is transmitted from the RTP media bridge in a separate RTP stream, thus using the same destination address/port combination, the same payload type number (PT) but separate RTP SSRC parameters and sequence number series as described in Section 7.1 and 7.2 of RTP RFC 3550 [RFC3550] about the Translator function. The source of the text in each RTP packet is identified by the SSRC parameter in the RTP packets, containing the SSRC of the initial source of text.

A receiving and presenting endpoint is supposed to separate text items from the different sources and identify and display them in a suitable way.

This method is described in RFC 7667, section 3.5.1 Relay-transport translator or 3.5.2 Media translator [RFC7667].

The identification of the source is made through the SSRC. The translation to a readable label can be done by mapping to information from the RTCP SDES CNAME and NAME packets as described in RTP[RFC3550], and also through information in the text media member in the conference notification described in RFC 4575 [RFC4575].

The sdp exchange for establishing this mixing type can be equal to what is used for basic two-party use of RFC 4103 with just an added attribute for indicating multi-party capability.

```
m=text 49170 RTP/AVP 98 103
a=rtpmap:98 red/1000
a=fmtp:98 103/103/103
a=rtpmap:103 t140/1000
a=fmtp:103 cps=150
a=RTT-mixing:RTP-translator
```

A similar answer including the same RTT-mixing attribute would indicate that multi-party coding can begin. An answer without the same RTT-mixing attribute could result in diversion to use of the mixing method for multi-party unaware endpoints Section 4.1.1.10 if more than two parties are involved in the session.

The bridge can add new sources in the communication to a participant by first sending a conference notification according to RFC 4575 [RFC4575] with the SSRC of the new source included in the corresponding "text" media member, or by sending an RTCP message with the new SSRC in an SDES packet.

A receiver should be prepared to receive such indications of new streams being added to the multi-party session, so that the new SSRC is not taken for a change in SSRC value for an already established RTP stream.

Transmission, reception, packet loss recovery and text loss indication is performed per source in the separate RTP streams in the same way as in two-party sessions with RFC 4103 [RFC4575].

Text is recommended to be sent by the bridge as soon as it is available for transmission, but not less than 250 ms after a previous transmission. This will in many cases result in close to 0 added delay by the bridge, because most RTT senders use a 300 ms transmission interval.

It is sometimes said that this configuration is not supported by current media declarations in sdp. RFC 3264 [RFC3264] specifies in some places that one media description is supposed to describe just one RTP media stream. However this is not directly referencing an RTP stream, and use of multiple RTP streams in the same RTP session is recommended in many other RFCs.

This confusion is clarified in RFC 5576 [RFC5576] section 3 by the following statements:

"The term "media stream" does not appear in the SDP specification itself, but is used by a number of SDP extensions, for instance, Interactive Connectivity Establishment (ICE) [ICE], to denote the object described by an SDP media description. This term is unfortunately rather confusing, as the RTP specification [RFC3550] uses the term "media stream" to refer to an individual media source or RTP packet stream, identified by an SSRC, whereas an SDP media stream describes an entire RTP session, which can contain any number of RTP sources."

In most cases, it will be sufficient that new sources are introduced with a conference notification or RTCP message. However, RFC 5576 [RFC5576] specifies attributes which may be used to more explicitly announce new sources or restart of earlier established RTP streams.

This method is encouraged by RFC 8872 [RFC8872] section 5.2.

One way of operation will be that the bridge receives text packets from the source and handles any text recovery and indication of loss needed before queueing the resulting clean text for transmission from the bridge to the receivers. However, that method requires the mixer to decrypt the payload of the packets and makes end-to-end encryption impossible.

It may however also be possible for the bridge to just convey the packet contents as received from the sources, with minor adjustments in the RTP header, and let the receiving endpoint handle all aspects of recovery and indication of loss, even for the source to bridge path. In that case also the sequence number sequence must be maintained as it was at reception in the bridge at least regarding gaps in the sequence. This mode needs further study before application.

Pros:

This method may be designed so that end-to-end encryption is enabled.

This method is a natural way to do multi-party bridging with RFC 4103 based RTT.

This method has moderate overhead in terms of work for the mixer, but high in terms of packet transmission rate. Five sources sending simultaneously cause the bridge to send 15 packets per second to each receiver.

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream (normally primary and two redundant levels).

More loss than what can be recovered, can be detected and the marker for text loss can be inserted in the correct stream.

It may be possible in some scenarios to keep the text encrypted through the Translator.

Minimal delay. The delay can often be kept close to 0 with at least 5 simultaneous sending participants.

Cons:

There are RTP implementations not supporting the Translator model. They will need to use the fall-back to multi-party-unaware mixing or another method based on RTP-mixer. An investigation about how common this lack of support is is needed before the method is used.

The processing time in standard organisation will be long.

With many simultaneous sending sources, the total rate of packets will be high, and can cause congestion. The requirement to handle 3 simultaneous sources in this specification will cause 10 packets per second that is manageable in most cases, e.g. considering that audio usually use 50 packets per second.

#### 4.1.2.2. Selective Forwarding Middlebox

From some points of view, use of multiple RTP streams, one for each source, sent in the same RTP session would be efficient, and would use exactly the same packet format as [RFC4103] and the same payload type.

A couple of relevant scenarios using multiple RTP-streams are specified in "RTP Topologies" [RFC7667]. One is described in the previous section. Another possibility of special interest is the Selective Forwarding Middlebox (SFM) topology specified in RFC 7667 section 3.7 that could enable end to end encryption. The idea of SFM is that the mixer selects a limited number of sources to be conveyed to the participants while other media streams are discarded. This causes very good efficiency for the audio and video media which are transmitted continuously from the sources.

In contrast to audio and video, real-time text is only transmitted when the users actually transmit information. Thus an SFM solution would not need to exclude any party from transmission under all normal conditions. It needs however be able to vary which sources are conveyed depending on which users are active transmitting at the moment.

In order to allow the mixer to convey the packets with the payload preserved and encrypted, an SFM solution would need to act on some specific characteristics of the "text/red" format. The redundancy headers are part of the payload, so the receiver would need to just assume that the payload type number in the redundancy header is for "text/t140". The characters per second parameter (CPS) would need to act per stream. The relation between the SSRC and the source would need to be conveyed in some specified way, e.g. in the CSRC. Recovery and loss detection would preferably be based on sequence number gap detection. Thus sequence number gaps in the incoming stream to the mixer would need to be reflected in the stream to the participant and no new gaps created by the mixer, even if the sequence number series may be different.

#### Pros:

This method may be designed so that end-to-end encryption is enabled.

This method is a natural way to do multi-party bridging with RFC 4103 based RTT.

This method has moderate overhead in terms of work for the mixer, but high in terms of packet transmission rate. Five sources sending simultaneously cause the bridge to send 15 packets per second to each receiver.

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream (normally primary and two redundant levels).

More loss than what can be recovered, can be detected and the marker for text loss can be inserted in the correct stream.

Minimal delay. The delay can often be kept close to 0 with at least 5 simultaneous sending participants.

#### Cons:

There are RTP implementations not supporting the SFM method. They will need to use the fall-back to multi-party-unaware mixing or another method based on RTP-mixer.

With very rarely occurring high number of simultaneous sending sources, the SFM will need to discard text from some sources in order to keep the total rate of packets at a suitable level. That can cause confusion.

This method requires a lot of further specification.

#### 4.1.2.3. Distributing packets in an end-to-end encryption structure

In order to achieve end-to-end encryption, it is possible to let the packets from the sources just pass through a central distributor, and handle the security agreements between the participants. Specifications exist for a framework with this functionality for application on RTP based conferences in [RFC8871]. The RTP flow and mixing characteristics has similarities with the method described under "RTP Translator sending one RTT stream per participant" above. RFC 4103 RTP streams [RFC4103] would fit into the structure and it would provide a base for end-to-end encrypted rtt multi-party conferencing.

Pros:

Good security

Straightforward multi-party handling.

Cons:

Does not operate under the usual SIP central conferencing architecture.

Requires the participants to perform a lot of key handling.

Is work in progress when this is written.

#### 4.1.2.4. Mesh of RTP endpoints

Text from all participants are transmitted directly to all others in one RTP session, without a central bridge. The sources of the text in each RTP packet are identified by the source network address and the SSRC.

This method is described in RFC 7667, section 3.4 Point to multi-point using mesh [RFC7667].

**Pros:**

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream. (normally primary and two redundant levels).

This method can be implemented with most RTP implementations.

Transmitted text can also be used with other transports than RTP

**Cons:**

This model is not described in IMS, NENA and EENA specifications, and does therefore not meet the requirements.

Requires a drastically increasing number of connections when the number of participants increase.

**4.1.2.5. Multiple RTP sessions, one for each participant**

Text from all participants are transmitted directly to all others in one RTP session each, without a central bridge. Each session is established with a separate media description in SDP. The sources of the text in each RTP packet are identified by the source network address and the SSRC.

**Pros:**

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream. (normally primary and two redundant levels).

Complete loss of text can be indicated in the received stream.

This method can be implemented with most RTP implementations.

End-to-end encryption is achievable.

**Cons:**

This method is not described in IMS, NENA and ETSI specifications and does therefore not meet the requirements.

A lot of network resources are spent on setting up separate sessions for each participant.

## 5. Preferred RTP-based multi-party RTT transport method

For RTP transport of RTT using RTP-mixer technology, one method for multi-party mixing and transport stand out as fulfilling the goals best and is therefore recommended. That is: "RTP Mixer interleaving packets, receiver using timestamp to recover from loss" Section 4.1.1.4. Of this reason, that method is brought forward to standardization and is now specified in [RFC9071].

For RTP transport in separate streams or sessions, no current recommendation can be made. A bridging method with interesting characteristics is the end-to-end encryption model "perc" Section 4.1.2.3, while also the method specified in [TS26114] Annex S also seems to have benefits.

## 6. Session control of RTP-based multi-party RTT sessions

General session control aspects for multi-party sessions are described in RFC 4575 [RFC4575] A Session Initiation Protocol (SIP) Event Package for Conference State, and RFC 4579 [RFC4579] Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents. The nomenclature of these specifications are used here.

The procedures for a multi-party aware model for RTT-transmission shall only be applied if a capability exchange for multi-party aware real-time text transmission has been completed and a supported method for multi-party real-time text transmission can be negotiated.

A method for detection of conference-awareness for centralized SIP conferencing in general is specified in RFC 4579 [RFC4579]. The focus sends the "isfocus" feature tag in a SIP Contact header. This causes the conference-aware endpoint to subscribe to conference notifications from the focus. The focus then sends notifications to the endpoint about entering and disappearing conference participants and their media capabilities. The information is carried XML-formatted in a 'conference-info' block in the notification according to RFC 4575 [RFC4575]. The mechanism is described in detail in RFC 4575 [RFC4575].

Before a conference media server starts sending multi-party RTT to an endpoint, a verification of its ability to handle multi-party RTT must be made. A decision on which mechanism to use for identifying text from the different participants must also be taken, implicitly or explicitly. These verifications and decisions can be done in a number of ways. The most apparent ways are specified here and their pros and cons described. One of the methods is selected to be the one to be used by implementations of the centralized conference model according to this specification.

### 6.1. Implicit RTT multi-party capability indication

Capability for RTT multi-party handling can be decided to be implicitly indicated by session control items.

The focus may implicitly indicate multi-party RTT capability by including the media child with value "text" in the RFC 4575 [RFC4575] conference-info provided in conference notifications.

An endpoint may implicitly indicate multi-party RTT capability by including the text media in the SDP in the session control transactions with the conference focus after the subscription to the conference has taken place.

The implicit RTT capability indication means for the focus that it can handle multi-party RTT according to the preferred method indicated in the RTT multi-party methods section above.

The implicit RTT capability indication means for the endpoint that it can handle multi-party RTT according to the preferred method indicated in the RTT multi-party methods section above.

If the focus detects that an endpoint implicitly declared RTT multi-party capability, it SHALL provide RTT according to the preferred method.

If the focus detects that the endpoint does not indicate any RTT multi-party capability, then it shall either provide RTT multi-party text in the way specified for conference-unaware endpoint above, or refuse to set up the session.

If the endpoint detects that the focus has implicitly declared RTT multi-party capability, it shall be prepared to present RTT in a multi-party fashion according to the preferred method.

#### Pros:

Acceptance of implicit multi-party capability implies that no standardisation of explicit RTT multi-party capability exchange is required.

#### Cons:

If other methods for multi-party RTT are to be used in the same implementation environment as the preferred ones, then capability exchange needs to be defined for them.

Cannot be used outside a strictly applied SIP central conference model.

## 6.2. RTT multi-party capability declared by SIP media-tags

Specifications for RTT multi-party capability declarations can be agreed for use as SIP media feature tags, to be exchanged during SIP call control operation according to the mechanisms in RFC 3840 [RFC3840] and RFC 3841 [RFC3841]. Capability for the RTT Multi-party capability is then indicated by the media feature tag "rtt-mix", with a set of possible values for the different possible methods.

The possible values in the list may for example be:

rtt-mixer

perc

rtt-mixer indicates capability for using the RTP-mixer based presentation of multi-party text.

perc indicates capability for using the perc based transmission of multi-party text.

Example: Contact: <sip:a2@beco.example.com>

```
;methods="INVITE,ACK,OPTIONS,BYE,CANCEL"
```

```
;+sip.rtt-mix="rtt-mixer"
```

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the media tag can be reduced to a single tag with no list of values.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that UA.

When no common method is declared, then only the fallback method for multi-party unaware participants can be used, or the session dropped.

If more than one text media section is included in SDP, all must be capable of using the declared RTT multi-party method.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can guide call routing to a suitable capable focus.

Cons:

Requires standardization and IANA registration.

Is not stream specific. If more than one text stream is specified, all must have the same type of multi-party capability.

Cannot be used in the WebRTC environment.

### 6.3. SDP media attribute for RTT multi-party capability indication

An attribute can be specified on media level, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The attribute can have the name "rtt-mixing".

More than one attribute can be included in one media description.

The attribute can have a value. The value can for example be:

rtt-mixer

rtt-translator

perc

rtt-mixer indicates capability for using the RTP-mixer and CSRC-list based mixing of multi-party text.

rtt-translator indicates capability for using the RTP-translator based mixing

perc indicates capability for using the perc based transmission of multi-party text.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that endpoint.

When no common method is declared, then only the fallback method for multi-party unaware endpoints can be used.

Example: a=rtt-mixing:rtp-mixer

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the attribute can be reduced to a single attribute with no list of values.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can be used on specific text media.

Can be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

Cannot guide SIP routing.

#### 6.4. Simplified SDP media attribute for RTT multi-party capability indication

An attribute can be specified on media level, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The attribute can have a name suitable for the selected method and no value. It would be selected and used if only one method for multi-party rtt is brought forward from this specification, and the other left unspecified for now or found to be possible to negotiate in another way.

An offer-answer exchange should take place and if both parties specify rtt-mixing capability with the same attribute, the selected mixing method shall be used.

When no common method is declared, then only the fallback method for multi-party unaware endpoints can be used, or the session not accepted for multi-party use.

Example: a=rtt-mixer

Pros:

Provides a clear decision method.

Very simple syntax and semantics.

Can be used on specific text media.

Cons:

Requires standardization and IANA registration.

If another RTT mixing method is also specified in the future, then that method may also need to specify and register its own attribute, instead of if an attribute with a parameter value is used, when only an addition of a new possible value is needed.

Cannot guide SIP routing.

#### 6.5. SDP format parameter for RTT multi-party capability indication

An FMTP format parameter can be specified for the RFC 4103 [RFC4103]media, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The parameter can have the name "rtt-mixing", with one or more of its possible values.

The possible values in the list are:

rtp-mixer

perc

rtp-mixer indicates capability for using the RTP-mixer based mixing and presentation of multi-party text using the CSRC-list.

perc indicates capability for using the perc based transmission of multi-party text.

Example: a=fmtp 96 98/98/98 rtt-mixing=rtp-mixer

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the parameter can be reduced to a single parameter with no list of values.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that UA.

When no common method is declared, then only the fallback method can be used, or the session denied.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can be used on specific text media.

Can be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

May cause interop problems with current RFC4103 [RFC4103] implementations not expecting a new fntp-parameter.

Cannot guide SIP routing.

#### 6.6. A text media subtype for support of multi-party rtt

Indicating a specific text media subtype in SDP is a straightforward way for negotiating multi-party capability. Especially if there are format differences from the "text/red" and "text/t140" formats of RFC4103 [RFC4103], then this is a natural way to do the negotiation for multi-party rtt.

Pros:

No extra efforts if a new format is needed anyway.

Cons:

None specific to using the format indication for negotiation of multi-party capability. But only feasible if a new format is needed anyway.

#### 6.7. Preferred capability declaration method for RTP-based transport.

The preferred capability declaration method is the simplified one with a specific SDP attribute for the selected mixing method Section 6.4 because it is straightforward. It is named "a=rtt-mixer" and included in [RFC9071].

## 6.8. Identification of the source of text for RTP-based solutions

The main way to identify the source of text in the RTP based solution is by the SSRC of the sending participant. In the RTP-mixer solution, this SSRC is included in the CSRC list of the transmitted packets. Further identification that may be needed for better labelling of received text may be achieved from a number of sources. It may be the RTCP SDES CNAME and NAME reports, and in the conference notification data (RFC 4575) [RFC4575].

As soon as a new member is added to the RTP session, its characteristics should be transmitted in RTCP SDES CNAME and NAME reports according to section 6.5 in RFC 3550 [RFC3550]. The information about the participant should also be included in the conference data including the text media member in a notification according to RFC 4575 [RFC4575].

The RTCP SDES report, SHOULD contain identification of the source represented by the SSRC/CSRC identifier. This identification MUST contain the CNAME field and MAY contain the NAME field and other defined fields of the SDES report.

A focus UA SHOULD primarily convey SDES information received from the sources of the session members. When such information is not available, the focus UA SHOULD compose SSRC/CSRC, CNAME and NAME information from available information from the SIP session with the participant.

Provision of detailed information in the NAME field has security implications, especially if provided without encryption.

## 7. RTT bridging in WebRTC

Within WebRTC, real-time text is specified to be carried in WebRTC data channels as specified in [RFC8865]. A few ways to handle multi-party RTT are mentioned briefly. The most straightforward one is referenced here.

### 7.1. RTT bridging in WebRTC with one data channel per source

A straightforward way to handle multi-party RTT is for the bridge to open one T.140 data channel per source towards the receiving participants.

The stream-id forms a unique stream identification.

The identification of the source is made through the Label property of the channel, and session information belonging to the source. The endpoint can compose a readable label for the presentation from this information.

This is the recommended solution.

Pros:

This is a straightforward solution.

The load per source is low.

Cons:

With a high number of participants, the overhead of establishing and maintaining the high number of data channels required may be high, even if the load per channel is low.

## 8. Presentation of multi-party text

All session participants with RTP based transport MUST observe the SSRC/CSRC field of incoming text RTP packets, and make note of which source they came from in order to be able to present text in a way that makes it easy to read text from each participant in a session, and get information about the source of the text.

In the WebRTC case, the Label parameter and other provided endpoint information should be used for the same purpose.

### 8.1. Associating identities with text streams

A source identity SHOULD be composed from available information sources and displayed together with the text as indicated in ITU-T T.140 Appendix[T140].

The source identity should primarily be the NAME field from incoming SDES packets. If this information is not available, and the session is a two-party session, then the T.140 source identity SHOULD be composed from the SIP session participant information. For multi-party sessions the source identity may be composed by local information if sufficient information is not available in the session.

Applications may abbreviate the presented source identity to a suitable form for the available display.

Applications may also replace received source information with internally used nicknames.

## 8.2. Presentation details for multi-party aware endpoints.

The multi-party aware endpoint should after any action for recovery of data from lost packets, separate the incoming streams and present them according to the style that the receiving application supports and the user has selected. The decisions taken for presentation of the multi-party interchange shall be purely on the receiving side. The sending application must not insert any item in the stream to influence presentation that is not requested by the sending participant.

### 8.2.1. Bubble style presentation

One often used style is to present real-time text in chunks in readable bubbles identified by labels containing names of sources. Bubbles are placed in one column in the presentation area and are closed and moved upwards in the presentation area after certain items or events, when there is also newer text from another source that would go into a new bubble.

The text items that allows bubble closing are any character closing a phrase or sentence followed by a space or a timeout of a suitable time (about 10 seconds).

Real-time active text sent from the local user should be presented in a separate area. When there is a reason to close a bubble from the local user, the bubble should be placed above all real-time active bubbles, so that the time order that real-time text entries were completed is visible.

Scrolling is usually provided for viewing of recent or older text. When scrolling is done to an earlier point in the text, the presentation shall not move the scroll position by new received text. It must be the decision of the local user to return to automatic viewing of latest text actions. It may be useful with an indication that there is new text to read after scrolling to an earlier position has been activated.

The presentation area may become too small to present all text in all real-time active bubbles. Various techniques can be applied to provide a good overview and good reading opportunity even in such situations. The active real-time bubble may have a limited number of lines and if their contents need more lines, then a scrolling opportunity within the real-time active bubble is provided. Another method can be to only show the label and the last line of the active real-time bubble contents, and make it possible to expand or compress the bubble presentation between full view and one line view.

Erasures require special consideration. Erasure within a real-time active bubble is straightforward. But if erasure from one participant affects the last character before a bubble, the whole previous bubble becomes the actual bubble for real-time action by that participant and is placed below all other bubbles in the presentation area. If the border between bubbles was caused by the CRLF characters (instead of the normal "Line Separator"), only one erasure action is required to erase this bubble border. When a bubble is closed, it is moved up, above all real-time active bubbles.

A three-party view is shown in this example .

	^
	-
[Alice] Hi, Alice here.	
[Bob] Bob as well.	
[Eve] Hi, this is Eve, calling from Paris. I thought you should be here.	
[Alice] I am coming on Thursday, my performance is not until Friday morning.	
[Bob] And I on Wednesday evening.	
[Alice] Can we meet on Thursday evening?	
[Eve] Yes, definitely. How about 7pm. at the entrance of the restaurant Le Lion Blanc?	
[Eve] we can have dinner and then take a walk	
<Eve-typing> But I need to be back to the hotel by 11 because I need	
<Bob-typing> I wou	-
of course, I underst	v

Figure 1: Three-party call with bubble style.

Figure 1: Example of a three-party call presented in the bubble style.

### 8.2.2. Other presentation styles

Other presentation styles than the bubble style may be arranged and appreciated by the users. In a video conference one way may be to have a real-time text area below the video view of each participant. Another view may be to provide one column in a presentation area for each participant and place the text entries in a relative vertical position corresponding to when text entry in them was completed. The labels can then be placed in the column header. The considerations for ending and moving and erasure of entered text discussed above for the bubble style are valid also for these styles.

This figure shows how a coordinated column view MAY be presented.

Bob	Eve	Alice
My flight is to Orly		I will arrive by TGV. Convenient to the main station.
Eve, will you do your presentation on Friday?	Hi all, can we plan for the seminar?	
Fine, wo	Yes, Friday at 10.	We need to meet befo

Figure 2: A coordinated column-view of a three-party session with entries ordered in approximate time-order.

#### 9. Presentation details for multi-party unaware endpoints.

Multi-party unaware endpoints are prepared only for presentation of two sources of text, the local user and a remote user. If mixing for multi-party unaware endpoints is to be supported, in order to enable some multi-party communication with such endpoint, the mixer need to plan the presentation and insert labels and line breaks before lables. Many limitations appear for this presentation mode, and it must be seen as a fallback and a last resort.

A procedure for presenting RTT to a conference-unaware endpoint is included in [RFC9071]

#### 10. Security Considerations

The security considerations valid for RFC 4103 [RFC4103] and RFC 3550 [RFC3550] are valid also for the multi-party sessions with text.

#### 11. IANA Considerations

The items for indication and negotiation of capability for multi-party rtt should be registered with IANA in the specifications where they are specified in detail.

#### 12. Congestion considerations

The congestion considerations described in RFC 4103 [RFC4103] are valid also for the recommended RTP-based multi-party use of the real-time text transport. A risk for congestion may appear if a number of conference participants are active transmitting text simultaneously, because the recommended RTP-based multi-party transmission method does not allow multiple sources of text to contribute to the same

packet.

In situations of risk for congestion, the Focus UA MAY combine packets from the same source to increase the transmission interval per source up to one second. Local conference policy in the Focus UA may be used to decide which streams shall be selected for such transmission frequency reduction.

### 13. Acknowledgements

Arnoud van Wijk for contributions to an earlier, expired draft of this memo.

### 14. Change history

#### 14.1. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-08

Update to align with published RFCs 8865, 8871, 8872, 9071.

#### 14.2. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-07

Adjustment of section 4.1.1.4 to match the specification in draft-ietf-avtcore-multi-party-rtt-mix-20.

#### 14.3. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-06

Addition of the Selective Forwarding Middlebox SFM among the methods with multiple RTP streams, to match the contents of draft-ietf-avtcore-multi-party-rtt-mix-12.

#### 14.4. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-05

Modify the solution changing source in every packet in the RTP-mixer solution, and base recovery on analyzing timestamp and make it the recommended one. Aligned with the recommendation in draft-ietf-avtcore-multi-party-rtt-mix-10.

#### 14.5. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-04

Change name of simplified sdp attribute to "rtt-mix" to match a change in the draft draft-ietf-avtcore-multi-party-rtt-mix-09.

#### 14.6. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-03

Modified info on the method with RFC 4103 format and sdp attribute "rtt-mix-rtp-mixer".

Increased the performance requirements section.

Inserted recommendations, with emphasis on ease of implementation and ease of standardisation.

14.7. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-02

Added detail in the section on RTP translator model alternative 4.1.2.1.

14.8. Changes to draft-hellstrom-avtcore-multi-party-rtt-solutions-01

Added three more methods for RTP-mixer mixing. Two RFC 5109 FEC based and another with modified data header to detect source of completely lost text.

Separated RTP-based and WebRTC based solutions.

Deleted the multi-party-unaware mixing procedure appendix. It is now included in the draft draft-ietf-avtcore-multi-party-rtt-mix. Kept a section with a reference to the new place.

14.9. Changes from draft-hellstrom-mmusic-multi-party-rtt-02 to draft-hellstrom-avtcore-multi-party-rtt-solutions-00

Add discussion about switching performance, as discussed in avtcore on March 13.

Added that a decrease of transmission interval to 100 ms increases switching performance by a factor 3, but still not sufficient.

Added that the CSRC-list method also uses 100 milliseconds transmission interval.

Added the method with multiple primary text in each packet.

Added the timestamp-based method for rtp-mixing proposed by James Hamlin on March 14.

Corrected the chat style presentation example picture. Delete a few "[mix]".

14.10. Changes from version draft-hellstrom-mmusic-multi-party-rtt-01 to -02

Change from a general overview to overview with clear recommendations.

Splits text coordination methods in three groups.

Recommends rtt-mixer with sources in CSRC-list but refers to its spec for details.

Shortened Appendix with conference-unaware example.

Cleaned up preferences.

Inserted pictures of screen-views.

## 15. References

### 15.1. 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>>.

### 15.2. Informative References

- [EN301549] ETSI, "EN 301 549. Accessibility requirements for ICT products and services", November 2019, <[https://www.etsi.org/deliver/etsi\\_en/301500\\_301599/301549/03.01.01\\_60/en\\_301549v030101p.pdf](https://www.etsi.org/deliver/etsi_en/301500_301599/301549/03.01.01_60/en_301549v030101p.pdf)>.
- [NENAi3] NENA, "NENA-STA-010.3a-2021. NENA i3 Standard for Next Generation 9-1-1", July 2021, <[https://www.nena.org/page/i3\\_Stage3](https://www.nena.org/page/i3_Stage3)>.
- [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>>.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <<https://www.rfc-editor.org/info/rfc3261>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.

- [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>>.
- [RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, DOI 10.17487/RFC3840, August 2004, <<https://www.rfc-editor.org/info/rfc3840>>.
- [RFC3841] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Caller Preferences for the Session Initiation Protocol (SIP)", RFC 3841, DOI 10.17487/RFC3841, August 2004, <<https://www.rfc-editor.org/info/rfc3841>>.
- [RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <<https://www.rfc-editor.org/info/rfc4103>>.
- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, DOI 10.17487/RFC4353, February 2006, <<https://www.rfc-editor.org/info/rfc4353>>.
- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, Ed., "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, DOI 10.17487/RFC4575, August 2006, <<https://www.rfc-editor.org/info/rfc4575>>.
- [RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, DOI 10.17487/RFC4579, August 2006, <<https://www.rfc-editor.org/info/rfc4579>>.
- [RFC4597] Even, R. and N. Ismail, "Conferencing Scenarios", RFC 4597, DOI 10.17487/RFC4597, August 2006, <<https://www.rfc-editor.org/info/rfc4597>>.
- [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>>.
- [RFC5194] van Wijk, A., Ed. and G. Gybels, Ed., "Framework for Real-Time Text over IP Using the Session Initiation Protocol (SIP)", RFC 5194, DOI 10.17487/RFC5194, June 2008, <<https://www.rfc-editor.org/info/rfc5194>>.

- [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>>.
- [RFC6443] Rosen, B., Schulzrinne, H., Polk, J., and A. Newton, "Framework for Emergency Calling Using Internet Multimedia", RFC 6443, DOI 10.17487/RFC6443, December 2011, <<https://www.rfc-editor.org/info/rfc6443>>.
- [RFC6881] Rosen, B. and J. Polk, "Best Current Practice for Communications Services in Support of Emergency Calling", BCP 181, RFC 6881, DOI 10.17487/RFC6881, March 2013, <<https://www.rfc-editor.org/info/rfc6881>>.
- [RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <<https://www.rfc-editor.org/info/rfc7667>>.
- [RFC8865] Holmberg, C. and G. Hellström, "T.140 Real-Time Text Conversation over WebRTC Data Channels", RFC 8865, DOI 10.17487/RFC8865, January 2021, <<https://www.rfc-editor.org/info/rfc8865>>.
- [RFC8871] Jones, P., Benham, D., and C. Groves, "A Solution Framework for Private Media in Privacy-Enhanced RTP Conferencing (PERC)", RFC 8871, DOI 10.17487/RFC8871, January 2021, <<https://www.rfc-editor.org/info/rfc8871>>.
- [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, January 2021, <<https://www.rfc-editor.org/info/rfc8872>>.
- [RFC9071] Hellström, G., "RTP-Mixer Formatting of Multiparty Real-Time Text", RFC 9071, DOI 10.17487/RFC9071, July 2021, <<https://www.rfc-editor.org/info/rfc9071>>.
- [T140] ITU-T, "Recommendation ITU-T T.140 (02/1998), Protocol for multimedia application text conversation", February 1998, <<https://www.itu.int/rec/T-REC-T.140-199802-I/en>>.
- [T140ad1] ITU-T, "Recommendation ITU-T.140 Addendum 1 - (02/2000), Protocol for multimedia application text conversation", February 2000, <<https://www.itu.int/rec/T-REC-T.140-200002-I!Add1/en>>.

- [TS103479] ETSI, "TS 103 479. Emergency communications (EMTEL); Core elements for network independent access to emergency services", December 2019, <[https://www.etsi.org/deliver/etsi\\_ts/103400\\_103499/103479/01.01.01\\_60/ts\\_103479v010101p.pdf](https://www.etsi.org/deliver/etsi_ts/103400_103499/103479/01.01.01_60/ts_103479v010101p.pdf)>.
- [TS22173] 3GPP, "IP Multimedia Core Network Subsystem (IMS) Multimedia Telephony Service and supplementary services; Stage 1", 3GPP TS 22.173 17.1.0, 20 December 2019, <<http://www.3gpp.org/ftp/Specs/html-info/22173.htm>>.
- [TS24147] 3GPP, "Conferencing using the IP Multimedia (IM) Core Network (CN) subsystem; Stage 3", 3GPP TS 24.147 16.0.0, 19 December 2019, <<http://www.3gpp.org/ftp/Specs/html-info/24147.htm>>.
- [TS26114] 3GPP, "IP Multimedia Subsystem (IMS) Multimedia Telephony; Media handling and interaction", 3GPP TS 26.114 17.3.0, 25 December 2021, <<http://www.3gpp.org/ftp/Specs/html-info/26114.htm>>.

## Author's Address

Gunnar Hellstrom  
Gunnar Hellstrom Accessible Communication  
SE-136 70 Vendelso  
Sweden

Email: [gunnar.hellstrom@ghaccess.se](mailto:gunnar.hellstrom@ghaccess.se)

IETF RMCAT Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: December 12, 2020

Z. Sarker  
Ericsson AB  
C. Perkins  
University of Glasgow  
V. Singh  
callstats.io  
M. Ramalho  
June 10, 2020

RTP Control Protocol (RTCP) Feedback for Congestion Control  
draft-ietf-avtcore-cc-feedback-message-07

Abstract

This document describes an RTCP feedback message intended to enable congestion control for interactive real-time traffic using RTP. The feedback message is designed for use with a sender-based congestion control algorithm, in which the receiver of an RTP flow sends RTCP feedback packets to the sender containing the information the sender needs to perform congestion control.

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 <https://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 December 12, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
3. RTCP Feedback for Congestion Control . . . . .	3
3.1. RTCP Congestion Control Feedback Report . . . . .	4
4. Feedback Frequency and Overhead . . . . .	7
5. Response to Loss of Feedback Packets . . . . .	7
6. SDP Signalling . . . . .	8
7. Relation to RFC 6679 . . . . .	8
8. Design Rationale . . . . .	9
9. Acknowledgements . . . . .	10
10. IANA Considerations . . . . .	11
11. Security Considerations . . . . .	11
12. References . . . . .	11
12.1. Normative References . . . . .	12
12.2. Informative References . . . . .	13
Authors' Addresses . . . . .	14

## 1. Introduction

For interactive real-time traffic, such as video conferencing flows, the typical protocol choice is the Real-time Transport Protocol (RTP) running over the User Datagram Protocol (UDP). RTP does not provide any guarantee of Quality of Service (QoS), reliability, or timely delivery, and expects the underlying transport protocol to do so. UDP alone certainly does not meet that expectation. However, the RTP Control Protocol (RTCP) provides a mechanism by which the receiver of an RTP flow can periodically send transport and media quality metrics to the sender of that RTP flow. This information can be used by the sender to perform congestion control. In the absence of standardized messages for this purpose, designers of congestion control algorithms have developed proprietary RTCP messages that convey only those parameters needed for their respective designs. As a direct result, the different congestion control (i.e., rate adaptation) designs are not interoperable. To enable algorithm evolution as well as interoperability across designs (e.g., different rate adaptation algorithms), it is highly desirable to have generic congestion control feedback format.

To help achieve interoperability for unicast RTP congestion control, this memo proposes a common RTCP feedback packet format that can be

used by NADA [RFC8698], SCReAM [RFC8298], Google Congestion Control [I-D.ietf-rmcat-gcc] and Shared Bottleneck Detection [RFC8382], and hopefully also by future RTP congestion control algorithms.

## 2. Terminology

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

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

## 3. RTCP Feedback for Congestion Control

Based on an analysis of NADA [RFC8698], SCReAM [RFC8298], Google Congestion Control [I-D.ietf-rmcat-gcc] and Shared Bottleneck Detection [RFC8382], the following per-RTP packet congestion control feedback information has been determined to be necessary:

- o RTP sequence number: The receiver of an RTP flow needs to feedback the sequence numbers of the received RTP packets to the sender, so the sender can determine which packets were received and which were lost. Packet loss is used as an indication of congestion by many congestion control algorithms.
- o Packet Arrival Time: The receiver of an RTP flow needs to feedback the arrival time of each RTP packet to the sender. Packet delay and/or delay variation (jitter) is used as a congestion signal by some congestion control algorithms.
- o Packet Explicit Congestion Notification (ECN) Marking: If ECN [RFC3168], [RFC6679] is used, it is necessary to feedback the 2-bit ECN mark in received RTP packets, indicating for each RTP packet whether it is marked not-ECT, ECT(0), ECT(1), or ECN-CE. If the path used by the RTP traffic is ECN capable the sender can use Congestion Experienced (ECN-CE) marking information as a congestion control signal.

Every RTP flow is identified by its Synchronization Source (SSRC) identifier. Accordingly, the RTCP feedback format needs to group its reports by SSRC, sending one report block per received SSRC.

As a practical matter, we note that host operating system (OS) process interruptions can occur at inopportune times. Accordingly, recording RTP packet send times at the sender, and the corresponding RTP packet arrival times at the receiver, needs to be done with deliberate care. This is because the time duration of host OS

interruptions can be significant relative to the precision desired in the one-way delay estimates. Specifically, the send time needs to be recorded at the last opportunity prior to transmitting the RTP packet at the sender, and the arrival time at the receiver needs to be recorded at the earliest available opportunity.

### 3.1. RTCP Congestion Control Feedback Report

Congestion control feedback can be sent as part of a regular scheduled RTCP report, or in an RTP/AVPF early feedback packet. If sent as early feedback, congestion control feedback MAY be sent in a non-compound RTCP packet [RFC5506] if the RTP/AVPF profile [RFC4585] or the RTP/SAVPF profile [RFC5124] is used.

Irrespective of how it is transported, the congestion control feedback is sent as a Transport Layer Feedback Message (RTCP packet type 205). The format of this RTCP packet is shown in Figure 1:

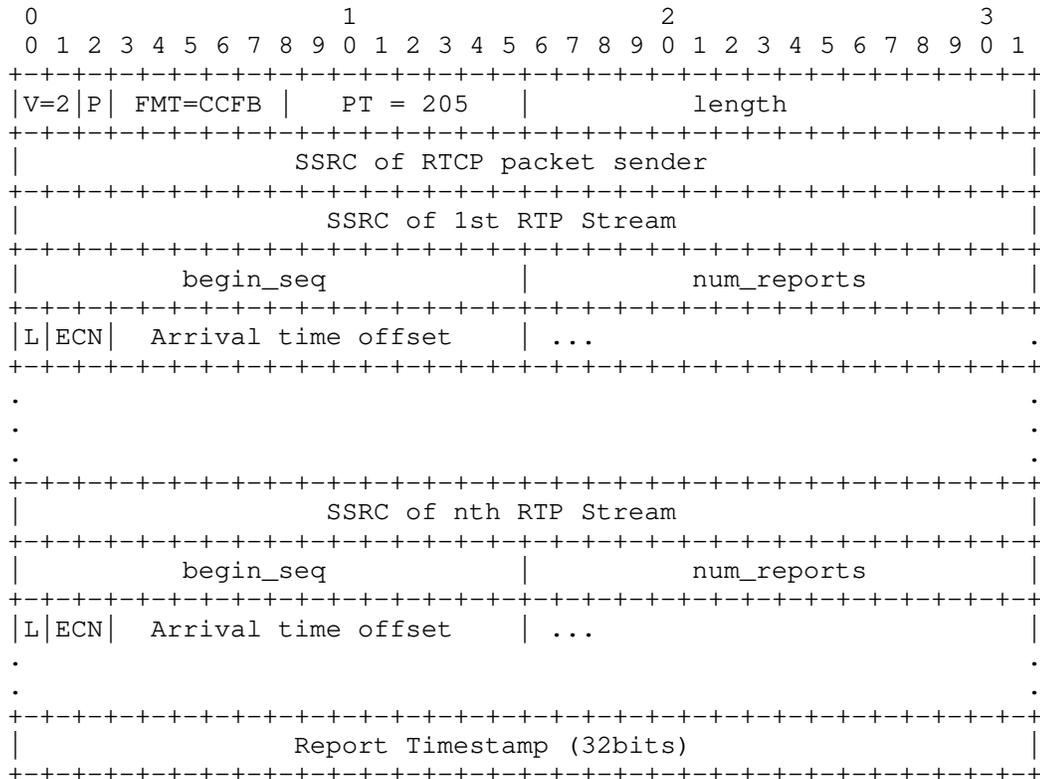


Figure 1: RTCP Congestion Control Feedback Packet Format

The first eight octets comprise a standard RTCP header, with PT=205 and FMT=CCFB indicating that this is a congestion control feedback packet, and with the SSRC set to that of the sender of the RTCP packet. (NOTE TO RFC EDITOR: please replace CCFB here and in the above diagram with the IANA assigned RTCP feedback packet type, and remove this note)

Section 6.1 of [RFC4585] requires the RTCP header to be followed by the SSRC of the RTP flow being reported upon. Accordingly, the RTCP header is followed by a report block for each SSRC from which RTP packets have been received, followed by a Report Timestamp.

Each report block begins with the SSRC of the received RTP Stream on which it is reporting. Following this, the report block contains a 16-bit packet metric block for each RTP packet with sequence number in the range `begin_seq` to `begin_seq+num_reports` inclusive (calculated using arithmetic modulo 65536 to account for possible sequence number wrap-around). If the number of 16-bit packet metric blocks included in the report block is not a multiple of two, then 16 bits of zero padding MUST be added after the last packet metric block, to align the end of the packet metric blocks with the next 32 bit boundary. The value of `num_reports` MAY be zero, indicating that there are no packet metric blocks included for that SSRC. Each report block MUST NOT include more than 16384 packet metric blocks (i.e., it MUST NOT report on more than one quarter of the sequence number space in a single report).

The contents of each 16-bit packet metric block comprises the L, ECN, and ATO fields are as follows:

- o L (1 bit): is a boolean to indicate if the packet was received. 0 represents that the packet was not yet received and all the subsequent bits (ECN and ATO) are also set to 0. 1 represent the packet was received and the subsequent bits in the block need to be parsed.
- o ECN (2 bits): is the echoed ECN mark of the packet. These are set to 00 if not received, or if ECN is not used.
- o Arrival time offset (ATO, 13 bits): is the arrival time of the RTP packet at the receiver, as an offset before the time represented by the Report Timestamp (RTS) field of this RTCP congestion control feedback report. The ATO field is in units of 1/1024 seconds (this unit is chosen to give exact offsets from the RTS field) so, for example, an ATO value of 512 indicates that the corresponding RTP packet arrived exactly half a second before the time instant represented by the RTS field. If the measured value is greater than 8189/1024 seconds (the value that would be coded

as 0x1FFD), the value 0x1FFE MUST be reported to indicate an over-range measurement. If the measurement is unavailable, or if the arrival time of the RTP packet is after the time represented by the RTS field, then an ATO value of 0x1FFF MUST be reported for the packet.

The RTCP congestion control feedback report packet concludes with the Report Timestamp field (RTS, 32 bits). This denotes the time instant on which this packet is reporting, and is the instant from which the arrival time offset values are calculated. The value of RTS field is derived from the same clock used to generate the NTP timestamp field in RTCP Sender Report (SR) packets. It is formatted as the middle 32 bits of an NTP format timestamp, as described in Section 4 of [RFC3550].

RTCP congestion control feedback packets SHOULD include a report block for every active SSRC. The sequence number ranges reported on in consecutive reports for a given SSRC will generally be contiguous, but overlapping reports MAY be sent (and need to be sent in cases where RTP packet reordering occurs across the boundary between consecutive reports). If reports covering overlapping sequence number ranges are sent, information in later reports updates that in sent previous reports for RTP packets included in both reports. If an RTP packet was reported as received in one report, that packet MUST also be reported as received in any overlapping reports sent later that cover its sequence number range.

RTCP congestion control feedback packets can be large if they are sent infrequently relative to the number of RTP data packets. If an RTCP congestion control feedback packet is too large to fit within the path MTU, its sender SHOULD split it into multiple feedback packets. The RTCP reporting interval SHOULD be chosen such that feedback packets are sent often enough that they are small enough to fit within the path MTU ([I-D.ietf-rmcat-rtp-cc-feedback] discusses how to choose the reporting interval; specifications for RTP congestion control algorithms can also provide guidance).

If duplicate copies of a particular RTP packet are received, then the arrival time of the first copy to arrive MUST be reported. If any of the copies of the duplicated packet are ECN-CE marked, then an ECN-CE mark MUST be reported that for packet; otherwise the ECN mark of the first copy to arrive is reported.

If no packets are received from an SSRC in a reporting interval, a report block MAY be sent with `begin_seq` set to the highest sequence number previously received from that SSRC and `num_reports` set to zero (or, the report can simply be omitted). The corresponding SR/RR packet will have a non-increased extended highest sequence number

received field that will inform the sender that no packets have been received, but it can ease processing to have that information available in the congestion control feedback reports too.

A report block indicating that certain RTP packets were lost is not to be interpreted as a request to retransmit the lost packets. The receiver of such a report might choose to retransmit such packets, provided a retransmission payload format has been negotiated, but there is no requirement that it do so.

#### 4. Feedback Frequency and Overhead

There is a trade-off between speed and accuracy of reporting, and the overhead of the reports. [I-D.ietf-rmcat-rtp-cc-feedback] discusses this trade-off, suggests desirable RTCP feedback rates, and provides guidance on how to configure the RTCP bandwidth fraction, etc., to make appropriate use of the reporting block described in this memo. Specifications for RTP congestion control algorithms can also provide guidance.

It is generally understood that congestion control algorithms work better with more frequent feedback. However, RTCP bandwidth and transmission rules put some upper limits on how frequently the RTCP feedback messages can be sent from an RTP receiver to the RTP sender. In many cases, sending feedback once per frame is an upper bound before the reporting overhead becomes excessive, although this will depend on the media rate and more frequent feedback might be needed with high-rate media flows [I-D.ietf-rmcat-rtp-cc-feedback]. Analysis [feedback-requirements] has also shown that some candidate congestion control algorithms can operate with less frequent feedback, using a feedback interval range of 50-200ms. Applications need to negotiate an appropriate congestion control feedback interval at session setup time, based on the choice of congestion control algorithm, the expected media bit rate, and the acceptable feedback overhead.

#### 5. Response to Loss of Feedback Packets

Like all RTCP packets, RTCP congestion control feedback packets might be lost. All RTP congestion control algorithms MUST specify how they respond to the loss of feedback packets.

If only a single congestion control feedback packet is lost, an appropriate response is to assume that the level of congestion has remained roughly the same as the previous report. However, if multiple consecutive congestion control feedback packets are lost, the sender SHOULD rapidly reduce its sending rate towards zero, as

this likely indicates a path failure. The RTP circuit breaker [RFC8083] provides further guidance.

## 6. SDP Signalling

A new "ack" feedback parameter, "ccfb", is defined for use with the "a=rtcp-fb:" SDP extension to indicate the use of the RTP Congestion Control feedback packet format defined in Section 3. The ABNF definition of this SDP parameter extension is:

```
rtcp-fb-ack-param = <See Section 4.2 of [RFC4585]>
rtcp-fb-ack-param =/ ccfb-par
ccfb-par           = SP "ccfb"
```

When used with "ccfb" feedback, the wildcard payload type ("\*") MUST be used. This implies that the congestion control feedback is sent for all payload types in use in the session, including any FEC and retransmission payload types. An example of the resulting SDP attribute is:

```
a=rtcp-fb:* ack ccfb
```

The offer/answer rules for these SDP feedback parameters are specified in Section 4.2 of the RTP/AVPF profile [RFC4585].

An SDP offer might indicate support for both the congestion control feedback mechanism specified in this memo and one or more alternative congestion control feedback mechanisms that offer substantially the same semantics. In this case, the answering party SHOULD include only one of the offered congestion control feedback mechanisms in its answer. If a re-invite offering the same set of congestion control feedback mechanisms is received, the generated answer SHOULD choose the same congestion control feedback mechanism as in the original answer where possible.

When the SDP BUNDLE extension [I-D.ietf-mmusic-sdp-bundle-negotiation] is used for multiplexing, the "a=rtcp-fb:" attribute has multiplexing category IDENTICAL-PER-PT [I-D.ietf-mmusic-sdp-mux-attributes].

## 7. Relation to RFC 6679

Use of Explicit Congestion Notification (ECN) with RTP is described in [RFC6679]. That specifies how to negotiate the use of ECN with RTP, and defines an RTCP ECN Feedback Packet to carry ECN feedback reports. It uses an SDP "a=ecn-capable-rtcp:" attribute to negotiate use of ECN, and the "a=rtcp-fb:" attributes with the "nack" parameter "ecn" to negotiate the use of RTCP ECN Feedback Packets.

The RTCP ECN Feedback Packet is not useful when ECN is used with the RTP Congestion Control Feedback Packet defined in this memo since it provides duplicate information. Accordingly, when congestion control feedback is to be used with RTP and ECN, the SDP offer generated MUST include an "a=ecn-capable-rtp:" attribute to negotiate ECN support, along with an "a=rtcp-fb:" attribute with the "ack" parameter "ccfb" to indicate that the RTP Congestion Control Feedback Packet is to be used for feedback. The "a=rtcp-fb:" attribute MUST NOT include the "nack" parameter "ecn", so the RTCP ECN Feedback Packet will not be used.

## 8. Design Rationale

The primary function of RTCP SR/RR packets is to report statistics on the reception of RTP packets. The reception report blocks sent in these packets contain information about observed jitter, fractional packet loss, and cumulative packet loss. It was intended that this information could be used to support congestion control algorithms, but experience has shown that it is not sufficient for that purpose. An efficient congestion control algorithm requires more fine grained information on per packet reception quality than is provided by SR/RR packets to react effectively. The feedback format defined in this memo provides such fine grained feedback.

Several other RTCP extensions also provide more detailed feedback than SR/RR packets:

**TMMBR:** The Codec Control Messages for the RTP/AVPF profile [RFC5104] include a Temporary Maximum Media Bit Rate (TMMBR) message. This is used to convey a temporary maximum bit rate limitation from a receiver of RTP packets to their sender. Even though it was not designed to replace congestion control, TMMBR has been used as a means to do receiver based congestion control where the session bandwidth is high enough to send frequent TMMBR messages, especially when used with non-compound RTCP packets [RFC5506]. This approach requires the receiver of the RTP packets to monitor their reception, determine the level of congestion, and recommend a maximum bit rate suitable for current available bandwidth on the path; it also assumes that the RTP sender can/will respect that bit rate. This is the opposite of the sender based congestion control approach suggested in this memo, so TMMBR cannot be used to convey the information needed for a sender based congestion control. TMMBR could, however, be viewed a complementary mechanism that can inform the sender of the receiver's current view of acceptable maximum bit rate. The Received Estimated Maximum Bit-rate (REMB) mechanism [I-D.alvestrand-remcat-remb] provides similar feedback.

RTCP Extended Reports (XR): Numerous RTCP extended report (XR) blocks have been defined to report details of packet loss, arrival times [RFC3611], delay [RFC6843], and ECN marking [RFC6679]. It is possible to combine several such XR blocks into a compound RTCP packet, to report the detailed loss, arrival time, and ECN marking information needed for effective sender-based congestion control. However, the result has high overhead both in terms of bandwidth and complexity, due to the need to stack multiple reports.

Transport-wide Congestion Control: The format defined in this memo provides individual feedback on each SSRC. An alternative is to add a header extension to each RTP packet, containing a single, transport-wide, packet sequence number, then have the receiver send RTCP reports giving feedback on these additional sequence numbers [I-D.holmer-rmcat-transport-wide-cc-extensions]. Such an approach adds the per-packet overhead of the header extension (8 octets per packet in the referenced format), but reduces the size of the feedback packets, and can simplify the rate calculation at the sender if it maintains a single rate limit that applies to all RTP packets sent irrespective of their SSRC. Equally, the use of transport-wide feedback makes it more difficult to adapt the sending rate, or respond to lost packets, based on the reception and/or loss patterns observed on a per-SSRC basis (for example, to perform differential rate control and repair for audio and video flows, based on knowledge of what packets from each flow were lost). Transport-wide feedback is also a less natural fit with the wider RTP framework, which makes extensive use of per-SSRC sequence numbers and feedback.

Considering these issues, we believe it appropriate to design a new RTCP feedback mechanism to convey information for sender based congestion control algorithms. The new congestion control feedback RTCP packet described in Section 3 provides such a mechanism.

## 9. Acknowledgements

This document is based on the outcome of a design team discussion in the RTP Media Congestion Avoidance Techniques (RMCAT) working group. The authors would like to thank David Hayes, Stefan Holmer, Randell Jesup, Ingemar Johansson, Jonathan Lennox, Sergio Mena, Nils Ohlmeier, Magnus Westerlund, and Xiaoqing Zhu for their valuable feedback.

## 10. IANA Considerations

The IANA is requested to register one new RTP/AVPF Transport-Layer Feedback Message in the table for FMT values for RTPFB Payload Types [RFC4585] as defined in Section 3.1:

Name: CCFB  
Long name: RTP Congestion Control Feedback  
Value: (to be assigned by IANA)  
Reference: (RFC number of this document, when published)

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

Value name: ccfb  
Long name: Congestion Control Feedback  
Usable with: ack  
Reference: (RFC number of this document, when published)

## 11. Security Considerations

The security considerations of the RTP specification [RFC3550], the applicable RTP profile (e.g., [RFC3551], [RFC3711], or [RFC4585]), and the RTP congestion control algorithm that is in use (e.g., [RFC8698], [RFC8298], [I-D.ietf-rmcat-gcc], or [RFC8382]) apply.

A receiver that intentionally generates inaccurate RTCP congestion control feedback reports might be able to trick the sender into sending at a greater rate than the path can support, thereby congesting the path. This will negatively impact the quality of experience of that receiver. Since RTP is an unreliable transport, a sender can intentionally leave a gap in the RTP sequence number space without causing harm, to check that the receiver is correctly reporting losses.

An on-path attacker that can modify RTCP congestion control feedback packets can change the reports to trick the sender into sending at either an excessively high or excessively low rate, leading to denial of service. The secure RTCP profile [RFC3711] can be used to authenticate RTCP packets to protect against this attack.

## 12. References

## 12.1. Normative References

- [I-D.ietf-mmusic-sdp-bundle-negotiation]  
Holmberg, C., Alvestrand, H., and C. Jennings,  
"Negotiating Media Multiplexing Using the Session  
Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-  
negotiation-54 (work in progress), December 2018.
- [I-D.ietf-mmusic-sdp-mux-attributes]  
Nandakumar, S., "A Framework for SDP Attributes when  
Multiplexing", draft-ietf-mmusic-sdp-mux-attributes-17  
(work in progress), February 2018.
- [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>>.
- [RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition  
of Explicit Congestion Notification (ECN) to IP",  
RFC 3168, DOI 10.17487/RFC3168, September 2001,  
<<https://www.rfc-editor.org/info/rfc3168>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.  
Jacobson, "RTP: A Transport Protocol for Real-Time  
Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550,  
July 2003, <<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>>.
- [RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed.,  
"RTP Control Protocol Extended Reports (RTCP XR)",  
RFC 3611, DOI 10.17487/RFC3611, November 2003,  
<<https://www.rfc-editor.org/info/rfc3611>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K.  
Norrman, "The Secure Real-time Transport Protocol (SRTP)",  
RFC 3711, DOI 10.17487/RFC3711, March 2004,  
<<https://www.rfc-editor.org/info/rfc3711>>.
- [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>>.

- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/info/rfc5124>>.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <<https://www.rfc-editor.org/info/rfc5506>>.
- [RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <<https://www.rfc-editor.org/info/rfc6679>>.
- [RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", RFC 8083, DOI 10.17487/RFC8083, March 2017, <<https://www.rfc-editor.org/info/rfc8083>>.

## 12.2. Informative References

- [feedback-requirements]  
"RMCAT Feedback Requirements",  
<[://www.ietf.org/proceedings/95/slides/slides-95-rmcat-1.pdf](http://www.ietf.org/proceedings/95/slides/slides-95-rmcat-1.pdf)>.
- [I-D.alvestrand-rmcat-remb]  
Alvestrand, H., "RTCP message for Receiver Estimated Maximum Bitrate", draft-alvestrand-rmcat-remb-03 (work in progress), October 2013.
- [I-D.holmer-rmcat-transport-wide-cc-extensions]  
Holmer, S., Flodman, M., and E. Sprang, "RTP Extensions for Transport-wide Congestion Control", draft-holmer-rmcat-transport-wide-cc-extensions-01 (work in progress), October 2015.
- [I-D.ietf-rmcat-gcc]  
Holmer, S., Lundin, H., Carlucci, G., Cicco, L., and S. Mascolo, "A Google Congestion Control Algorithm for Real-Time Communication", draft-ietf-rmcat-gcc-02 (work in progress), July 2016.

- [I-D.ietf-rmcat-rtcp-cc-feedback]  
Perkins, C., "RTP Control Protocol (RTCP) Feedback for Congestion Control in Interactive Multimedia Conferences", draft-ietf-rmcat-rtcp-cc-feedback-05 (work in progress), November 2019.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.
- [RFC6843] Clark, A., Gross, K., and Q. Wu, "RTP Control Protocol (RTCP) Extended Report (XR) Block for Delay Metric Reporting", RFC 6843, DOI 10.17487/RFC6843, January 2013, <<https://www.rfc-editor.org/info/rfc6843>>.
- [RFC8298] Johansson, I. and Z. Sarker, "Self-Clocked Rate Adaptation for Multimedia", RFC 8298, DOI 10.17487/RFC8298, December 2017, <<https://www.rfc-editor.org/info/rfc8298>>.
- [RFC8382] Hayes, D., Ed., Ferlin, S., Welzl, M., and K. Hiorth, "Shared Bottleneck Detection for Coupled Congestion Control for RTP Media", RFC 8382, DOI 10.17487/RFC8382, June 2018, <<https://www.rfc-editor.org/info/rfc8382>>.
- [RFC8698] Zhu, X., Pan, R., Ramalho, M., and S. Mena, "Network-Assisted Dynamic Adaptation (NADA): A Unified Congestion Control Scheme for Real-Time Media", RFC 8698, DOI 10.17487/RFC8698, February 2020, <<https://www.rfc-editor.org/info/rfc8698>>.

#### Authors' Addresses

Zaheduzzaman Sarker  
Ericsson AB  
Torshamnsgatan 21  
Stockholm 164 40  
Sweden

Phone: +46107173743  
Email: [zaheduzzaman.sarker@ericsson.com](mailto:zaheduzzaman.sarker@ericsson.com)

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

Email: [csp@csp Perkins.org](mailto:csp@csp Perkins.org)

Varun Singh  
CALLSTATS I/O Oy  
Annankatu 31-33 C 42  
Helsinki 00100  
Finland

Email: [varun.singh@iki.fi](mailto:varun.singh@iki.fi)  
URI: <http://www.callstats.io/>

Michael A. Ramalho  
6310 Watercrest Way Unit 203  
Lakewood Ranch, FL 34202-5122  
USA

Phone: +1 732 832 9723  
Email: [mar42@cornell.edu](mailto:mar42@cornell.edu)  
URI: <http://ramalho.webhop.info/>

AVTCORE WG  
Internet-Draft  
Updates: 3550, 3551 (if approved)  
Intended status: Standards Track  
Expires: June 20, 2016

M. Westerlund  
Ericsson  
C. Perkins  
University of Glasgow  
J. Lennox  
Vidyo  
December 18, 2015

Sending Multiple Types of Media in a Single RTP Session  
draft-ietf-avtccore-multi-media-rtp-session-13

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

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 June 20, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction . . . . . 2  
 2. Terminology . . . . . 3  
 3. Background and Motivation . . . . . 3  
 4. Applicability . . . . . 4  
 5. Using Multiple Media Types in a Single RTP Session . . . . . 6  
   5.1. Allowing Multiple Media Types in an RTP Session . . . . . 6  
   5.2. Demultiplexing media types within an RTP session . . . . . 7  
   5.3. Per-SSRC Media Type Restrictions . . . . . 8  
   5.4. RTCP Considerations . . . . . 8  
 6. Extension Considerations . . . . . 9  
   6.1. RTP Retransmission Payload Format . . . . . 9  
   6.2. RTP Payload Format for Generic FEC . . . . . 10  
   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 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

[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. Section 5 discusses issues arising from the base RTP and RTCP specification 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 an 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 [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 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 specification, because the design of 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. Various RTP extensions also make assumptions about 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 meets 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 quality of service (QoS) or RTCP configuration for different RTP streams are better suited by sending those RTP streams in separate RTP session, using separate transport layer

flows for each, since that gives greater flexibility. Further guidance on how to provide differential treatment for some media is given in [I-D.ietf-avtcore-multiplex-guidelines] 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 rate 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 reporting interval, is 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 [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 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 type of media in a single RTP session. It is not possible, in the general case, to implement a gateway that can interconnect an endpoint using 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 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 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 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 limiting when using different media types and RTP payload formats within a single RTP session.

Avoids 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 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 specification's purpose is to override 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 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 [RFCXXXX] is used, and the application conforms to the applicability constraints. 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 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 based on their SSRC into the different RTP streams. 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, then the RTP payload type is 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 media type during its lifetime. For example, an SSRC can change between different audio formats, but 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 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 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

The RTP Retransmission Payload Format [RFC4588] can operate in either SSRC-multiplexed mode or session-multiplex 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 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 original and retransmission RTP stream. Another alternative is the use of different RTP payload types, allowing the signalled "apt" (associated payload type) parameter 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 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 [I-D.ietf-mmusic-sdp-bundle-negotiation] and FID grouping [RFC5888] extensions. 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. In case 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 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:100 apt=31;rtx-time=3000
a=mid:kelp
m=video 40000 RTP/AVPF 100
a=rtpmap:100 rtx/90000
a=fmtp:100 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) [RFC5109] (and its predecessor [RFC2733]) can either send the FEC stream as a separate RTP stream, or it can 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 for [RFC5109], and the "application/parityfec" media type 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 unnecessarily uses up RTP payload type values, and adds no value for demultiplexing since 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 [I-D.ietf-mmusic-sdp-bundle-negotiation]. 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 [RFC5956] grouping or, for backwards compatibility, the FEC [RFC4756] grouping. This is similar to the situation that arises for RTP retransmission with session multiplexing discussed in Section 6.1.

The Source-Specific Media Attributes [RFC5576] specification 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 was work on an Unequal Layer Protection (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].

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 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 RTP session level parameters, for example the RTCP RR and RS bandwidth modifiers, the RTP/AVPF trr-int parameter, 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 [I-D.ietf-mmusic-sdp-bundle-negotiation] 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 set-up time, etc.

## 9. IANA Considerations

This memo makes no request of IANA.

## 10. Acknowledgements

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

## 11. References

### 11.1. Normative References

- [I-D.ietf-avtcore-rtp-multi-stream]  
Lennox, J., Westerlund, M., Wu, Q., and C. Perkins,  
"Sending Multiple RTP Streams in a Single RTP Session",  
draft-ietf-avtcore-rtp-multi-stream-11 (work in progress),  
December 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-23 (work in progress), July 2015.
  
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119,  
DOI 10.17487/RFC2119, March 1997,  
<<http://www.rfc-editor.org/info/rfc2119>>.
  
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.  
Jacobson, "RTP: A Transport Protocol for Real-Time  
Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550,  
July 2003, <<http://www.rfc-editor.org/info/rfc3550>>.
  
- [RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and  
Video Conferences with Minimal Control", STD 65, RFC 3551,  
DOI 10.17487/RFC3551, July 2003,  
<<http://www.rfc-editor.org/info/rfc3551>>.

## 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.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.
- [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,  
DOI 10.17487/RFC2198, September 1997,  
<<http://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,  
<<http://www.rfc-editor.org/info/rfc2733>>.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session  
Description Protocol", RFC 4566, DOI 10.17487/RFC4566,  
July 2006, <<http://www.rfc-editor.org/info/rfc4566>>.
- [RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R.  
Hakenberg, "RTP Retransmission Payload Format", RFC 4588,  
DOI 10.17487/RFC4588, July 2006,  
<<http://www.rfc-editor.org/info/rfc4588>>.
- [RFC4756] Li, A., "Forward Error Correction Grouping Semantics in  
Session Description Protocol", RFC 4756,  
DOI 10.17487/RFC4756, November 2006,  
<<http://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, <<http://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,  
<<http://www.rfc-editor.org/info/rfc5576>>.

- [RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, DOI 10.17487/RFC5888, June 2010, <<http://www.rfc-editor.org/info/rfc5888>>.
- [RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in the Session Description Protocol", RFC 5956, DOI 10.17487/RFC5956, September 2010, <<http://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, <<http://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, <<http://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, <<http://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, <<http://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, <<http://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, <<http://www.rfc-editor.org/info/rfc7657>>.

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@csp Perkins.org

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

Email: jonathan@vidyo.com

AVTCore  
Internet-Draft  
Updates: RFC 4102, RFC 4103 (if approved)  
Intended status: Standards Track  
Expires: 13 January 2021

G. Hellstrom  
Gunnar Hellstrom Accessible Communication  
12 July 2020

RTP-mixer formatting of multi-party Real-time text  
draft-ietf-avtcore-multi-party-rtt-mix-07

Abstract

Real-time text mixers for multi-party sessions need to identify the source of each transmitted group of text so that the text can be presented by endpoints in suitable grouping with other text from the same source.

Regional regulatory requirements specify provision of real-time text in multi-party calls. RFC 4103 mixer implementations can use traditional RTP functions for source identification, but the mixer source switching performance is limited when using the default transmission with redundancy.

Enhancements for RFC 4103 real-time text mixing is provided in this document, suitable for a centralized conference model that enables source identification and source switching. The intended use is for real-time text mixers and multi-party-aware participant endpoints. Two mechanisms are provided. The mechanisms builds on use of the CSRC list in the RTP packet for source identification. One method makes use of the same "text/red" format as for two-party sessions, while the other makes use of an extended packet format "text/rex" for more efficient transmission.

A capability exchange is specified so that it can be verified that a participant can handle the multi-party coded real-time text stream. The capability for one method is by use of a media attribute a=rtt-mix-rtp-mixer. The other method is indicated by the media subtype "text/rex".

The document updates RFC 4102[RFC4102] and RFC 4103[RFC4103]

A brief description about how a mixer can format text for the case when the endpoint is not multi-party aware is also provided.

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 <https://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 13 January 2021.

#### Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

#### Table of Contents

1. Introduction . . . . .	3
1.1. Selected solution and considered alternative . . . . .	5
1.2. Nomenclature . . . . .	6
1.3. Intended application . . . . .	7
2. Specified solutions . . . . .	7
2.1. Negotiated use of the RFC 4103 format for multi-party in a single RTP stream . . . . .	7
2.2. Use of an extended packet format "text/rex" with text from multiple sources . . . . .	17
2.3. Mixing for multi-party unaware endpoints . . . . .	35
3. Presentation level considerations . . . . .	36
3.1. Presentation by multi-party aware endpoints . . . . .	36
3.2. Multi-party mixing for multi-party unaware endpoints . . . . .	38
4. Gateway Considerations . . . . .	44
4.1. Gateway considerations with Textphones (e.g. TTYs). . . . .	44
4.2. Gateway considerations with WebRTC. . . . .	45
5. Updates to RFC 4102 and RFC 4103 . . . . .	45
6. Congestion considerations . . . . .	46
7. Acknowledgements . . . . .	46
8. IANA Considerations . . . . .	46

8.1.	Registration of the "rtt-mix-rtp-mixer" sdp media attribute . . . . .	46
8.2.	Registration of "text/rex" media subtype . . . . .	47
9.	Security Considerations . . . . .	47
10.	Change history . . . . .	47
10.1.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-07 . . . . .	47
10.2.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-06 . . . . .	47
10.3.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-05 . . . . .	48
10.4.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-04 . . . . .	48
10.5.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-03 . . . . .	48
10.6.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-02 . . . . .	49
10.7.	Changes to draft-ietf-avtcore-multi-party-rtt-mix-01 . . . . .	49
10.8.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-03 to draft-ietf-avtcore-multi-party-rtt-mix-00 . . . . .	50
10.9.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-02 to -03 . . . . .	50
10.10.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-01 to -02 . . . . .	50
10.11.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-00 to -01 . . . . .	51
11.	References . . . . .	51
11.1.	Normative References . . . . .	51
11.2.	Informative References . . . . .	53
	Author's Address . . . . .	53

## 1. Introduction

RFC 4103[RFC4103] specifies use of RFC 3550 RTP [RFC3550] for transmission of real-time text (RTT) and the "text/t140" format. It also specifies a redundancy format "text/red" for increased robustness. RFC 4102 [RFC4102] registers the "text/red" format. Regional regulatory requirements specify provision of real-time text in multi-party calls.

Real-time text is usually provided together with audio and sometimes with video in conversational sessions.

The redundancy scheme of RFC 4103 [RFC4103] enables efficient transmission of redundant text in packets together with new text. However the redundancy header format has no source indicators for the redundant transmissions. An assumption has had to be made that the redundant parts in a packet are from the same source as the new text. The recommended transmission is one new and two redundant generations of text (T140blocks) in each packet and the recommended transmission interval is 300 ms.

A mixer, selecting between text input from different sources and transmitting it in a common stream needs to make sure that the receiver can assign the received text to the proper sources for presentation. Therefore, using RFC 4103 without any extra rule for source identification, the mixer needs to stop sending new text from one source and then make sure that all text so far has been sent with all intended redundancy levels (usually two) before switching to another source. That causes the long time of one second to switch between transmission of text from one source to text from another source when using the default transmission interval 300 ms. Both the total throughput and the switching performance in the mixer would be too low for most applications. However by shorting the transmission interval to 100 ms, good performance is achieved for up to 3 simultaneously sending sources and usable performance for up to 5 simultaneously sending sources. This method is negotiated through an sdp media attribute "rtt-mix-rtp-mixer".

A more efficient source identification scheme requires that each redundant T140block has its source individually preserved. This document introduces a source indicator by specific rules for populating the CSRC-list and the data header in the RTP-packet.

An extended packet format "text/rex" is specified for this purpose, providing the possibility to include text from up to 15 sources in each packet in order to enhance mixer source switching performance. By these extensions, the performance requirements on multi-party mixing for real-time text are exceeded by the "text/rex" solution in this document.

A negotiation mechanism can therefore be based on selection of the "text/red" with media attribute "rtt-mix-rtp-mixer" or the "text/rex" media format for verification that the parties are able to handle a multi-party coded stream and agreeing on which method to use.

A fall-back mixing procedure is specified for cases when the negotiation results in "text/red" without the "rtt-mix" attribute" being the only common submedia format.

The document updates RFC 4102[RFC4102] and RFC 4103[RFC4103] by introducing an attribute for indicating multi-party capability, and an extended packet format for the multi-party mixing case and more strict rules for the source indications.

### 1.1. Selected solution and considered alternative

A number of alternatives were considered when searching an efficient multi-party method for real-time text. This section explains a few of them briefly.

One RTP stream per source, sent in the same RTP session with "text/red" format. From some points of view, use of multiple RTP streams, one for each source, sent in the same RTP session, called the RTP translator model in RFC 3550 [RFC3550], would be efficient, and use exactly the same packet format as RFC 4103, the same payload type and a simple SDP declaration. However, there is currently lack of support for multi-stream RTP in certain implementation technologies. The multi-stream solution would also cause more overhead than a single RTP stream solution "text/rex" specified in this document and more the more simultaneous sending participants there are.

The "text/red" format in RFC 4103 with shorter transmission interval, and indicating source in CSRC. The "text/red" format with "text/t140" payload in a single RTP stream can be sent with 100 ms packet intervals instead of the regular 300 ms. The source is indicated in the CSRC field. Source switching can then be done every 300 ms while simultaneous transmission occurs. With two participants sending text simultaneously, the switching and transmission performance is good. With three or more simultaneously sending participants, there will be a noticeable jerkiness in text presentation, more the more participants who send text simultaneously. With three sending participants, the jerkiness will be about 450 ms, and with five, about 1350 ms. Text sent from a source at the end of the period its text is sent by the mixer will have close to zero extra delay. Recent text will be presented with no or low delay. The 1350 ms jerkiness will be noticeable and slightly unpleasant, but corresponds in time to what typing humans often cause by hesitation or changing position while typing. A benefit of this method is that no new packet format needs to be introduced and implemented. Since simultaneous typing by more than two parties is rare, and in many applications also more than three parties in a call is rare, this method can be used successfully without its limitations becoming annoying. Negotiation is based on a new sdp media attribute "rtt-mix-rtp-mixer".

The "text/rex" packet format with up to 15 sources in one packet. The mechanism called "text/rex" specified in this document makes use of the RTP mixer model specified in RFC3550[RFC3550]. Text from up to 15 sources can be included in each packet. Packets are normally sent every 300 ms. The mean delay will be 150 ms. The sources are indicated in the CSRC list of the RTP packets. A new redundancy packet format is specified, named "text/rex".

The presentation planned by the mixer for multi-party unaware endpoints. It is desirable to have a method that does not require any modifications in existing user devices implementing RFC 4103 for RTT without explicit support of multi-party sessions. This is possible by having the mixer insert a new line and a text formatted source label before each switch of text source in the stream. Switch of source can only be done in places in the text where it does not disturb the perception of the contents. Text from only one source can be presented in real time at a time. The delay will therefore be varying. The method has also other limitations, but is included in this document as a fallback method. In calls where parties take turns properly by ending their entries with a new line, the limitations will have limited influence on the user experience. While only two parties send text, these two will see the text in real time with no delay.

## 1.2. Nomenclature

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].

The terms SDES, CNAME, NAME, SSRC, CSRC, CSRC list, CC, RTCP, RTP-mixer, RTP-translator are explained in [RFC3550]

The term "T140block" is defined in RFC 4103 [RFC4103] to contain one or more T.140 code elements.

"TTY" stands for a text telephone type used in North America.

"WebRTC" stands for web based communication specified by W3C and IETF.

"DTLS-SRTP" stands for security specified in RFC 5764 [RFC5764].

### 1.3. Intended application

The methods for multi-party real-time text are primarily intended for use in transmission between mixers and endpoints in centralised mixing configurations. It is also applicable between endpoints as well as between mixers. An often mentioned application is for emergency service calls with real-time text and voice, where a calltaker want to make an attended handover of a call to another agent, and stay observing in the session. Multimedia conference sessions with support for participants to contribute in text is another application. Conferences with central support for speech-to-text conversion is yet another mentioned application.

In all these applications, normally only one participant at a time will send long text utterances. In some cases, one other participant will occasionally contribute with a longer comment simultaneously. That may also happen in some rare cases when text is interpreted to text in another language in a conference. Apart from these cases, other participants are only expected to contribute with very brief utterings while others are sending text.

Text is supposed to be human generated, by some text input means, such as typing on a keyboard or using speech-to-text technology. Occasional small cut-and-paste operations may appear even if that is not the initial purpose of real-time text.

The real-time characteristics of real-time text is essential for the participants to be able to contribute to a conversation. If the text is too much delayed from typing a letter to its presentation, then, in some conference situations, the opportunity to comment will be gone and someone else will grab the turn. A delay of more than one second in such situations is an obstacle for good conversation.

## 2. Specified solutions

### 2.1. Negotiated use of the RFC 4103 format for multi-party in a single RTP stream

This section specifies use of the current format specified in [RFC4103] for true multi-party real-time text. It is an update of RFC 4103 by a clarification on one way to use it in the multi-party situation. It is done by completing a negotiation for this kind of multi-party capability and by indicating source in the CSRC element in the RTP packets. Please use [RFC4103] as reference when reading the following description.

### 2.1.1. Negotiation for use of this method

RFC 4103[RFC4103] specifies use of RFC 3550 RTP[RFC3550], and a redundancy format "text/red" for increased robustness of real-time text transmission. This document updates RFC 4102[RFC4102] and RFC 4103[RFC4103] by introducing a capability negotiation for handling multi-party real-time text. The capability negotiation is based on use of the sdp media attribute "rtt-mix-rtp-mixer".

The syntax is as follows:

```
a=rtt-mix-rtp-mixer
```

A transmitting party SHALL send text according to the multi-party format only when the negotiation for this method was successful and when the CC field in the RTP packet is 1. In all other cases, the packets SHALL be populated as for a two-party session.

### 2.1.2. Use of fields in the RTP packets

The CC field SHALL show the number of members in the CSRC list, which is one (1) in transmissions from a mixer involved in a multi-party session, and otherwise 0.

When transmitted from a mixer during a multi-party session, a CSRC list is included in the packet. The single member in the CSRC-list SHALL contain the SSRC of the source of the T140blocks in the packet. When redundancy is used, the recommended level of redundancy is to use one primary and two redundant generations of T140blocks. In some cases, a primary or redundant T140block is empty, but is still represented by a member in the redundancy header.

From other aspects, the contents of the RTP packets are equal to what is specified in RFC 4103.

### 2.1.3. Transmission of multi-party contents

As soon as a participant is known to participate in a session and being available for text reception, a Unicode BOM character SHALL be sent to it according to the procedures in this section. If the transmitter is a mixer, then the source of this character SHALL be indicated to be the mixer itself.

### 2.1.4. Keep-alive

After that, the transmitter SHALL send keep-alive traffic to the receivers at regular intervals when no other traffic has occurred during that interval if that is decided for the actual connection. Recommendations for keep-alive can be found in RFC 6263[RFC6263].

#### 2.1.5. Transmission interval

A "text/red" transmitter SHOULD send packets distributed in time as long as there is something (new or redundant T140blocks) to transmit. The maximum transmission interval SHOULD then be 300 ms. It is RECOMMENDED to send next packet to a receiver as soon as new text to that receiver is available, as long as the time after the latest sent packet to the same receiver is more than or equal to 100 ms, and also the maximum character rate to the receiver is not exceeded. The intention is to keep the latency low while keeping a good protection against text loss in bursty packet loss conditions. New and redundant text from one source MAY be transmitted in the same packet. Text from different sources MUST NOT be transmitted in the same packet.

#### 2.1.6. Do not send received text to the originating source

Text received from a participant SHOULD NOT be included in transmission to that participant.

#### 2.1.7. Clean incoming text

A mixer SHALL handle reception and recovery of packet loss, marking of possible text loss and deletion of 'BOM' characters from each participant before queueing received text for transmission to receiving participants.

#### 2.1.8. Redundancy

The transmitting party using redundancy SHALL send redundant repetitions of T140blocks already transmitted in earlier packets. The number of redundant generations of T140blocks to include in transmitted packets SHALL be deducted from the SDP negotiation. It SHOULD be set to the minimum of the number declared by the two parties negotiating a connection.

#### 2.1.9. Text placement in packets

At time of transmission, the mixer SHALL populate the RTP packet with all T140blocks queued for transmission originating from the source in turn for transmission as long as this is not in conflict with the allowed number of characters per second or the maximum packet size. The SSRC of the source shall be placed as the member in the CSRC-list.

#### 2.1.10. Maximum number of sources per packet

When text from more than one source is available for transmission, the mixer SHALL let the sources take turns in having their text transmitted. When switching from transmission of one source to allow another source to have its text sent, all intended redundant generations of the last text from the current source MUST be transmitted before text from another source can be transmitted. Actively transmitting sources SHOULD be allowed to take turns as frequently as possible to have their text transmitted. That implies that with the recommended redundancy, the mixer SHALL send primary text and two packets with redundant text from the current source before text from another source is transmitted. The source with the oldest received text in the mixer SHOULD be next in turn to get all its available text transmitted.

Note: The CSRC-list in an RTP packet only includes the participant who's text is included in text blocks. It is not the same as the total list of participants in a conference. With audio and video media, the CSRC-list would often contain all participants who are not muted whereas text participants that don't type are completely silent and thus are not represented in RTP packet CSRC-lists once their text have been transmitted as primary and the intended number of redundant generations.

#### 2.1.11. Empty T140blocks

If no unsent T140blocks were available for a source at the time of populating a packet, but T140blocks are available which have not yet been sent the full intended number of redundant transmissions, then the primary T140block for that source is composed of an empty T140block, and populated (without taking up any length) in a packet for transmission. The corresponding SSRC SHALL be placed as usual in its place in the CSRC-list.

#### 2.1.12. Creation of the redundancy

The primary T140block from a source in the latest transmitted packet is used to populate the first redundant T140block for that source. The first redundant T140block for that source from the latest transmission is placed as the second redundant T140block.

Usually this is the level of redundancy used. If a higher number of redundancy is negotiated, then the procedure SHALL be maintained until all available redundant levels of T140blocks are placed in the packet. If a receiver has negotiated a lower number of "text/red" generations, then that level shall be the maximum used by the transmitter.

#### 2.1.13. Timer offset fields

The timestamp offset values are inserted in the data header, with the time offset from the RTP timestamp in the packet when the corresponding T140block was sent from its original source as primary.

The timestamp offsets are expressed in the same clock tick units as the RTP timestamp.

The timestamp offset values for empty T140blocks have no relevance but SHOULD be assigned realistic values.

#### 2.1.14. Other RTP header fields

The number of members in the CSRC list ( 0 or 1) shall be placed in the "CC" header field. Only mixers place value 1 in the "CC" field.

The current time SHALL be inserted in the timestamp.

The SSRC of the mixer for the RTT session SHALL be inserted in the SSRC field of the RTP header.

The M-bit shall be handled as specified in [RFC4103].

#### 2.1.15. Pause in transmission

When there is no new T140block to transmit, and no redundant T140block that has not been retransmitted the intended number of times from any source, the transmission process can stop until either new T140blocks arrive, or a keep-alive method calls for transmission of keep-alive packets.

#### 2.1.16. RTCP considerations

A mixer SHALL send RTCP reports with SDES, CNAME and NAME information about the sources in the multi-party call. This makes it possible for participants to compose a suitable label for text from each source.

#### 2.1.17. Reception of multi-party contents

The "text/red" receiver included in an endpoint with presentation functions will receive RTP packets in the single stream from the mixer, and SHALL distribute the T140blocks for presentation in presentation areas for each source. Other receiver roles, such as gateways or chained mixers are also feasible, and requires consideration if the stream shall just be forwarded, or distributed based on the different sources.

#### 2.1.17.1. Multi-party vs two-party use

If the "CC" field value of a received packet is 1, it indicates that multi-party transmission is active, and the receiver MUST be prepared to act on the source according to its role. If the CC value is 0, the connection is point-to-point.

#### 2.1.17.2. Level of redundancy

The used level of redundancy generations SHALL be evaluated from the received packet contents. The number of generations (including the primary) is equal to the number of members in the redundancy header.

#### 2.1.17.3. Extracting text and handling recovery and loss

The RTP sequence numbers of the received packets SHALL be monitored for gaps and packets out of order.

As long as the sequence is correct, each packet SHALL be unpacked in order. The T140blocks SHALL be extracted from the primary area, and the corresponding SSRC SHALL be extracted from the CSRC list and used for assigning the new T140block to the correct presentation areas (or correspondingly for other receiver roles).

If a sequence number gap appears and is still there after some defined time for jitter resolution, T140data SHALL be recovered from redundant data. If the gap is wider than the number of generations of redundant T140blocks in the packet, then a t140block SHALL be created with a marker for possible text loss [T140ad1] and assigned to the SSRC of the transmitter as a general input from the mixer because in general it is not possible to deduct from which source(s) text was lost. It is in some cases possible to deduct that no text was lost even for a gap wider than the redundancy generations, and in some cases it can be concluded which source that likely had loss. Therefore, the receiver MAY insert the marker for possible text loss [T140ad1] in the presentation area corresponding to the source which may have had loss.

Then, the T140block in the received packet SHALL be retrieved beginning with the highest redundant generation, and assigning it to the presentation area of that source. Finally the primary T140block SHALL be retrieved from the packet and similarly assigned to the corresponding presentation area for the source.

If the sequence number gap was equal to or less than the number of redundancy generations in the received packet, a missing text marker SHALL NOT be inserted, and instead the T140block and the SSRC fully recovered from the redundancy information and the CSRC-list in the way indicated above.

#### 2.1.17.4. Delete BOM

Unicode character "BOM" is used as a start indication and sometimes used as a filler or keep alive by transmission implementations. These SHALL be deleted on reception.

#### 2.1.17.5. Empty T140blocks

Empty T140blocks are included as fillers for unused redundancy levels in the packets. They just do not provide any contents and do not contribute to the received streams.

#### 2.1.18. Performance considerations

This solution has good performance up to three participants simultaneously sending text. At higher numbers of participants simultaneously sending text, a jerkiness is visible in the presentation of text. With five participants simultaneously transmitting text, the jerkiness is about 1400 ms. Evenso, the transmission of text catches up, so there is no resulting delay introduced. The solution is therefore suitable for emergency service use, relay service use, and small or well-managed larger multimedia conferences. It is only less suitable for large conferences with a high number of participants sending text simultaneously. It should be noted that it is only the number of users sending text within the same moment that causes jerkiness, not the total number of users with RTT capability.

#### 2.1.19. Offer/answer considerations

A party which has negotiated the "rtt-mix-rtp-mixer" sdp media attribute MUST populate the CSRC-list and format the packets according to this section if it acts as an rtp-mixer and sends multi-party text.

A party which has negotiated the the "rtt-mix-rtp-mixer" sdp media attribute MUST interpret the contents of the CSRC-list and the packets according to this section in received rtp packets in the corresponding RTP stream.

A party performing as a mixer, which has not negotiated the "rtt-mix-rtp-mixer" sdp media attribute, but negotiated a "text/red" or "text/t140" format in a session with a participant SHOULD, if nothing else is specified for the application, format transmitted text to that participant to be suitable to present on a multi-party unaware endpoint as further specified in section Section 3.2.

A party not performing as a mixer MUST not include the CSRC list.

#### 2.1.20. Security for session control and media

Security SHOULD be applied on both session control and media. In applications where legacy endpoints without security may exist, a negotiation between security and no security SHOULD be applied. If no other security solution is mandated by the application, then RFC 8643 OSRTP[RFC8643] SHOULD be applied to negotiate SRTP media security with DTLS. Most SDP examples below are for simplicity expressed without the security additions. The principles (but not all details) for applying DTLS-SRTP security is shown in a couple of the following examples.

#### 2.1.21. SDP offer/answer examples

This sections shows some examples of SDP for session negotiation of the real-time text media in SIP sessions. Audio is usually provided in the same session, and sometimes also video. The examples only show the part of importance for the real-time text media.

Offer example for "text/red" format and multi-party support:

```
m=text 11000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mix-rtp-mixer
```

Answer example from a multi-party capable device

```
m=text 11000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mix-rtp-mixer
```

Offer example for "text/red" format including multi-party and security:

```
a=fingerprint: SHA-1 \  
4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB  
m=text 11000 RTP/AVP 100 98  
a=rtpmap:98 t140/1000  
a=rtpmap:100 red/1000  
a=fmtp:100 98/98/98  
a=rtt-mix-rtp-mixer
```

The "Fingerprint" is sufficient to offer DTLS-SRTP, with the media line still indicating RTP/AVP.

Answer example from multi-party capable device with security

```
a=fingerprint: SHA-1 \  
FF:FF:FF:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB  
m=text 11000 RTP/AVP 100 98  
a=rtpmap:98 t140/1000  
a=rtpmap:100 red/1000  
a=fmtp:100 98/98/98  
a=rtt-mix-rtp-mixer
```

With the "fingerprint" the device acknowledges use of SRTP/DTLS.

Answer example from a multi-party unaware device that also does not support security:

```
m=text 12000 RTP/AVP 100 98  
a=rtpmap:98 t140/1000  
a=rtpmap:100 red/1000  
a=fmtp:100 98/98/98
```

#### 2.1.22. Packet sequence example

This example shows a symbolic flow of packets from a mixer with loss and recovery. A and B are sources of RTT. P indicates primary data. R1 is first redundant generation data and R2 is second redundant generation data. A1, B1, A2 etc are text chunks (T140blocks) received from the respective sources. X indicates dropped packet between the mixer and a receiver.

```

-----
Seq no 1
CC=1
CSRC list A
R2: A1
R1: A2
P: A3
-----

```

Assuming that earlier packets ( with text A1 and A2) were received in sequence, text A3 is received from packet 1 and assigned to reception area A. The mixer is now assumed to have received text from source B and need to prepare for sending that text. First it must send the redundant generations of text A1.

```

-----
Seq no 2
CC=1
CSRC list A
R2 A2
R1: A3
P: Empty
-----

```

Nothing needs to be retrieved from this packet.

```

X-----
X Seq no 3
X CC=1
X CSRC list A
X R2: A3
X R1: Empty
X P: Empty
X-----

```

Packet 3 is assumed to be dropped in network problems

```

X-----
X Seq no 4
X CC=1
X CSRC list B
X R2: Empty
X R1: Empty
X P2: B1
X-----

```

Packet 4 contains text from B, assumed dropped in network problems. The mixer is assumed to have received text from A on turn to send. Sending of text from B must therefore be temporarily ended by sending redundancy twice.

```

X-----
X Seq no 5
X CC=1
X CSRC list B
X R2: Empty
X R1: B1
X P:  Empty
X-----

```

Packet 5 is assumed to be dropped in network problems

```

-----
Seq no 6
CC=1
CSRC list B
R2: B1
R1: Empty
P:  Empty
-----

```

Packet 6 is received. The latest received sequence number was 2. Recovery is therefore tried for 3,4,5. There is no coverage for seq no 3. But knowing that A1 must have been sent as R2 in packet 3, it can be concluded that nothing was lost.

For seqno 4, text B1 is recovered from the second generation redundancy and appended to the reception area of B. For seqno 5, nothing needs to be recovered. No primary text is available in packet 6.

After this sequence, A3 and B1 have been received. In this case no text was lost. Even if also packet 2 was lost, it can be concluded that no text was lost.

If also packets 1 and 2 were lost, there would be a need to create a marker for possibly lost text (U'FFFD) [T140ad1], inserted generally and possibly also in text sequences A and B.

## 2.2. Use of an extended packet format "text/rex" with text from multiple sources

The method specified in this section called "text/rex" has higher performance than the previous method. Text from up to 15 sources can be included in each packet. This may be of value in large non-managed conferences.

2.2.1. Use of fields in the RTP packets

RFC 4103[RFC4103] specifies use of RFC 3550 RTP[RFC3550], and a redundancy format "text/red" for increased robustness of real-time text transmission. This document updates RFC 4102[RFC4102] and RFC 4103[RFC4103] by introducing a format "text/rex" with a rule for populating and using the CSRC-list in the RTP packet and extending the redundancy header to be called a data header. This is done in order to enhance the performance in multi-party RTT sessions.

The "text/rex" format can be seen as an "n-tuple" variant of the "text/red" format intended to carry text information from up to 15 sources per packet.

The CC field SHALL show the number of members in the CSRC list, which is one per source represented in the packet.

When transmitted from a mixer, a CSRC list is included in the packet. The members in the CSRC-list SHALL contain the SSRCs of the sources of the T140blocks in the packet. The order of the CSRC members MUST be the same as the order of the sources of the data header fields and the T140blocks. When redundancy is used, text from all included sources MUST have the same number of redundant generations. The primary, first redundant, second redundant and possible further redundant generations of T140blocks MUST be grouped per source in the packet in "source groups". The recommended level of redundancy is to use one primary and two redundant generations of T140blocks. In some cases, a primary or redundant T140block is empty, but is still represented by a member in the data header.

The RTP header is followed by one or more source groups of data headers: one header for each text block to be included. Each of these data headers provides the timestamp offset and length of the corresponding data block, in addition to the payload type number corresponding to the payload format "text/t140". The data headers are followed by the data fields carrying T140blocks from the sources.

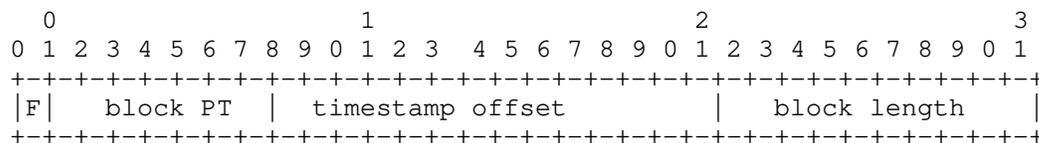


Figure 1: The bits in the data header.

The bits in the data header are specified as follows:

F: 1 bit First bit in header indicates whether another header block

follows. It has value 1 if further header blocks follow, and value 0 if this is the last header block.

block PT: 7 bits RTP payload type number for this block, corresponding to the t140 payload type from the RTPMAP SDP attribute.

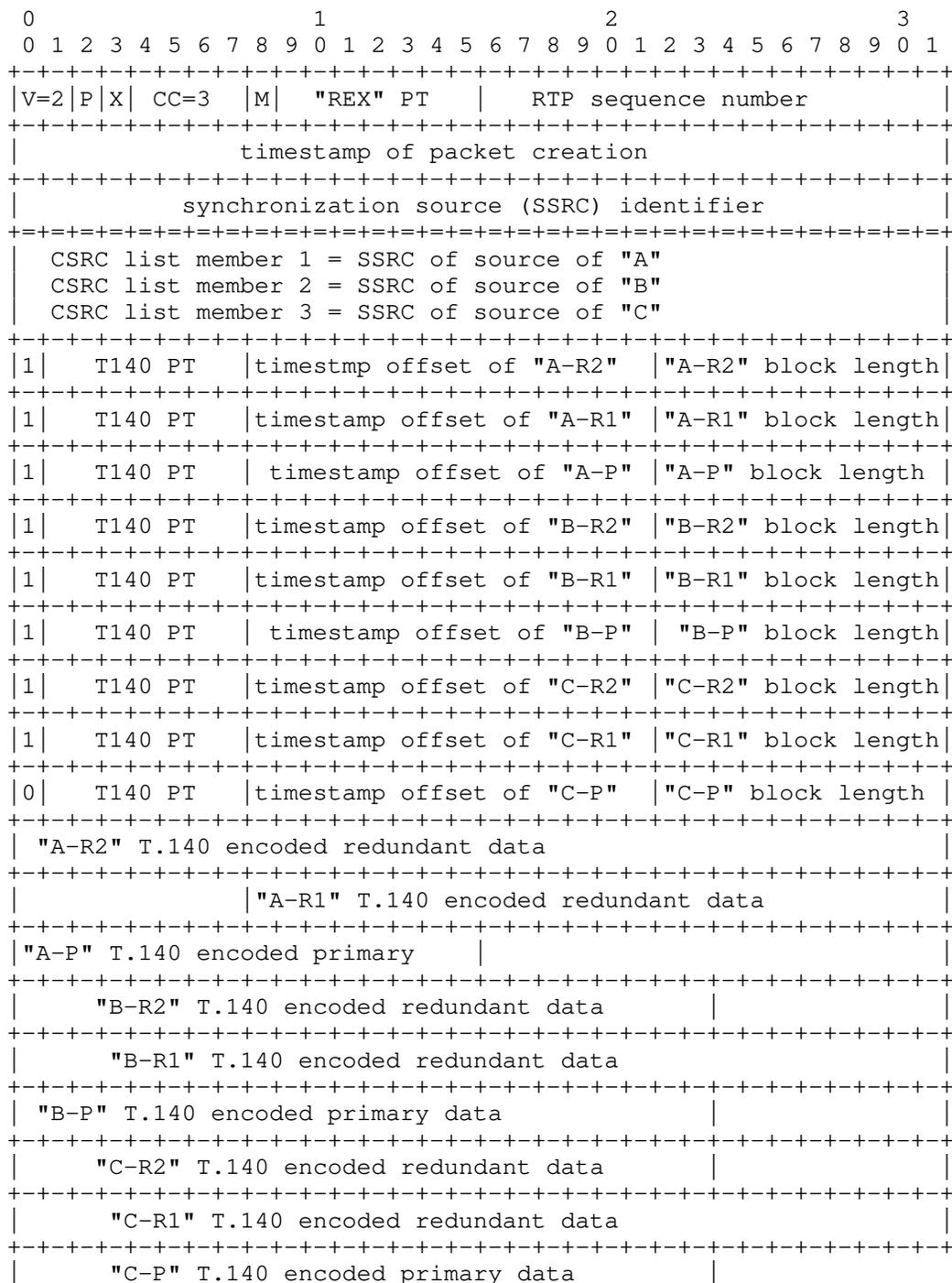
timestamp offset: 14 bits Unsigned offset of timestamp of this block relative to the timestamp given in the RTP header. The offset is a time to be subtracted from the current timestamp to determine the timestamp of the data when the latest part of this block was sent from the original source. If the timestamp offset would be >15 000, it SHALL be set to 15 000. For redundant data, the resulting time is the time when the data was sent as primary from the original source. If the value would be >15 000, then it SHALL be set to 15 000 plus 300 times the redundancy level of the data. The high values appear only in exceptional cases, e.g. when some data has been held in order to keep the text flow under the Characters Per Second (CPS) limit.

block length: 10 bits Length in bytes of the corresponding data block excluding the header.

The header for the final block has a zero F bit, and apart from that the same fields as other data headers.

Note: The "text/rex" packet format is similar to that of RFC 2198 [RFC2198] but is different from some aspects. RFC 2198 associates the whole of the CSRC-list with the primary data and assumes that the same list applies to reconstructed redundant data. In this section a T140block is associated with exactly one CSRC list member as described above. Also RFC 2198 [RFC2198] anticipates infrequent change to CSRCs; implementers should be aware that the order of the CSRC-list according to this section will vary during transitions between transmission from the mixer of text originated by different participants. Another difference is that the last member in the data header area in RFC 2198 [RFC2198] only contains the payload type number while in this section it has the same format as all other entries in the data header.

The picture below shows a typical "text/rex" RTP packet with multi-party RTT contents from three sources and coding according to this section.



+++++

Figure 2:A "text/rex" packet with text from three sources A, B, C.

A-P, B-P, C-P are primary data from A, B and C.

A-R1, B-R1, C-R1 are first redundant generation data from A, B and C.

A-R2, B-R2, C-R2 are first redundant generation data from A, B and C.

In a real case, some of the data headers will likely indicate a zero block length, and no corresponding T.140 data.

#### 2.2.2. Actions at transmission by a mixer

##### 2.2.2.1. Initial BOM transmission

As soon as a participant is known to participate in a session and being available for text reception, a Unicode "BOM" character SHALL be sent to it according to the procedures in this section. If the transmitter is a mixer, then the source of this character SHALL be indicated to be the mixer itself.

##### 2.2.2.2. Keep-alive

After that, the transmitter SHALL send keep-alive traffic to the receivers at regular intervals when no other traffic has occurred during that interval if that is decided for the actual connection. Recommendations for keep-alive can be found in RFC 6263[RFC6263].

##### 2.2.2.3. Transmission interval

A "text/rex" transmitter SHOULD send packets distributed in time as long as there is something (new or redundant T140blocks) to transmit. The maximum transmission interval SHOULD then be 300 ms. It is RECOMMENDED to send a packet to a receiver as soon as new text to that receiver is available, as long as the time after the latest sent packet to the same receiver is more than 150 ms, and also the maximum character rate to the receiver is not exceeded. The intention is to keep the latency low while keeping a good protection against text loss in bursty packet loss conditions.

##### 2.2.2.4. Do not send received text to the originating source

Text received from a participant SHOULD NOT be included in transmission to that participant.

#### 2.2.2.5. Clean incoming text

A mixer SHALL handle reception and recovery of packet loss, marking of possible text loss and deletion of 'BOM' characters from each participant before queueing received text for transmission to receiving participants.

#### 2.2.2.6. Redundancy

The transmitting party using redundancy SHALL send redundant repetitions of T140blocks already transmitted in earlier packets. The number of redundant generations of T140blocks to include in transmitted packets SHALL be deducted from the SDP negotiation. It SHOULD be set to the minimum of the number declared by the two parties negotiating a connection. The same number of redundant generations MUST be used for text from all sources when it is transmitted to a receiver. The number of generations sent to a receiver SHALL be the same during the whole session unless it is modified by session renegotiation.

#### 2.2.2.7. Text placement in packets

At time of transmission, the mixer SHALL populate the RTP packet with T140blocks combined from all T140blocks queued for transmission originating from each source as long as this is not in conflict with the allowed number of characters per second or the maximum packet size. These T140blocks SHALL be placed in the packet interleaved with redundant T140blocks and new T140blocks from other sources. The SSRC of each source shall be placed as a member in the CSRC-list at a place corresponding to the place of its T140blocks in the packet.

#### 2.2.2.8. Maximum number of sources per packet

Text from a maximum of 15 sources MAY be included in a packet. The reason for this limitation is the maximum number of CSRC list members allowed in a packet. If text from more sources need to be transmitted, the mixer MAY let the sources take turns in having their text transmitted. When stopping transmission of one source to allow another source to have its text sent, all intended redundant generations of the last text from the source to be stopped MUST be transmitted before text from another source can be transmitted. Actively transmitting sources SHOULD be allowed to take turns with short intervals to have their text transmitted.

Note: The CSRC-list in an RTP packet only includes participants who's text is included in text blocks. It is not the same as the total list of participants in a conference. With audio and video media, the CSRC-list would often contain all participants who are not muted

whereas text participants that don't type are completely silent and thus are not represented in RTP packet CSRC-lists once their text have been transmitted as primary and the intended number of redundant generations.

#### 2.2.2.9. Empty T140blocks

If no unsent T140blocks were available for a source at the time of populating a packet, but T140blocks are available which have not yet been sent the full intended number of redundant transmissions, then the primary T140block for that source is composed of an empty T140block, and populated (without taking up any length) in a packet for transmission. The corresponding SSRC SHALL be placed in its place in the CSRC-list.

#### 2.2.2.10. Creation of the redundancy

The primary T140block from each source in the latest transmitted packet is used to populate the first redundant T140block for that source. The first redundant T140block for that source from the latest transmission is placed as the second redundant T140block.

Usually this is the level of redundancy used. If a higher number of redundancy is negotiated, then the procedure SHALL be maintained until all available redundant levels of T140blocks and their sources are placed in the packet. If a receiver has negotiated a lower number of "text/rex" generations, then that level shall be the maximum used by the transmitter.

#### 2.2.2.11. Timer offset fields

The timer offset values are inserted in the data header, with the time offset from the RTP timestamp in the packet when the corresponding T140block was sent from its original source as primary.

The timer offsets are expressed in the same clock tick units as the RTP timestamp.

The timestamp offset values for empty T140blocks have no relevance but SHOULD be assigned realistic values.

#### 2.2.2.12. Other RTP header fields

The number of members in the CSRC list shall be placed in the "CC" header field. Only mixers place values >0 in the "CC" field.

The current time SHALL be inserted in the timestamp.

The SSRC of the mixer for the RTT session SHALL be inserted in the SSRC field of the RTP header.

The M-bit SHALL be set to 1 first in the session and after a pause.

#### 2.2.2.13. Pause in transmission

When there is no new T140block to transmit, and no redundant T140block that has not been retransmitted the intended number of times, the transmission process can stop until either new T140blocks arrive, or a keep-alive method calls for transmission of keep-alive packets.

#### 2.2.3. Actions at reception

The "text/rex" receiver included in an endpoint with presentation functions will receive RTP packets in the single stream from the mixer, and SHALL distribute the T140blocks for presentation in presentation areas for each source. Other receiver roles, such as gateways or chained mixers are also feasible, and requires consideration if the stream shall just be forwarded, or distributed based on the different sources.

##### 2.2.3.1. Multi-party vs two-party use

If the "CC" field value of a received packet is >0, it indicates that multi-party transmission is active, and the receiver MUST be prepared to act on the different sources according to its role. If the CC value is 0, the transmission is point-to-point.

##### 2.2.3.2. Level of redundancy

The used level of redundancy generations SHALL be evaluated from the received packet contents. If the CC value is 0, the number of generations (including the primary) is equal to the number of members in the data header. If the CC value is >0, the number of generations (including the primary) is equal to the number of members in the data header divided by the CC value. If the remainder from the division is >0, then the packet is malformed and SHALL cause an error indication in the receiver.

##### 2.2.3.3. Extracting text and handling recovery and loss

The RTP sequence numbers of the received packets SHALL be monitored for gaps and packets out of order.

As long as the sequence is correct, each packet SHALL be unpacked in order. The T140blocks SHALL be extracted from the primary areas, and the corresponding SSRCs SHALL be extracted from the corresponding positions in the CSRC list and used for assigning the new T140block to the correct presentation areas (or correspondingly).

If a sequence number gap appears and is still there after some defined time for jitter resolution, T140data SHALL be recovered from redundant data. If the gap is wider than the number of generations of redundant T140blocks in the packet, then a t140block SHALL be created with a marker for possible text loss [T140ad1] and assigned to the SSRC of the transmitter as a general input from the mixer because in general it is not possible to deduct from which sources text was lost. It is however likely that the sources which had loss were active in transmission just before or after the sequence number gap. Therefore, the receiver MAY insert the marker for possible text loss [T140ad1] in the presentation areas corresponding to the sources which had text in the packets just before and after the gap.

Then, the T140blocks in the received packet SHALL be retrieved beginning with the highest redundant generation, grouping them with the corresponding SSRC from the CSRC-list and assigning them to the presentation areas per source. Finally the primary T140blocks SHALL be retrieved from the packet and similarly their sources retrieved from the corresponding positions in the CSRC-list, and then assigned to the corresponding presentation areas for the sources.

If the sequence number gap was equal to or less than the number of redundancy generations in the received packet, a missing text marker SHALL NOT be inserted, and instead the T140blocks and their SSRCs fully recovered from the redundancy information and the CSRC-list in the way indicated above.

#### 2.2.3.4. Delete BOM

Unicode character "BOM" is used as a start indication and sometimes used as a filler or keep alive by transmission implementations. These SHALL be deleted on reception.

#### 2.2.3.5. Empty T140blocks

Empty T140blocks are included as fillers for unused redundancy levels in the packets. They just do not provide any contents and do not contribute to the received streams.

#### 2.2.4. RTCP considerations

A mixer SHALL send RTCP reports with SDES, CNAME and NAME information about the sources in the multi-party call. This makes it possible for participants to compose a suitable label for text from each source.

#### 2.2.5. Chained operation

By strictly applying the rules for "text/rex" packet format by all conforming devices, mixers MAY be arranged in chains.

#### 2.2.6. Usage without redundancy

The "text/rex" format SHALL be used also for multi-party communication when the redundancy mechanism is not used. That MAY be the case when robustness in transmission is provided by some other means than by redundancy. All aspects of this section SHALL be applied except the redundant generations in transmission.

The "text/rex" format SHOULD thus be used for multi-party operation, also when some other protection against packet loss is utilized, for example a reliable network or transport. The format is also suitable to be used for point-to-point operation.

#### 2.2.7. Use with SIP centralized conferencing framework

The SIP conferencing framework, mainly specified in RFC 4353[RFC4353], RFC 4579[RFC4579] and RFC 4575[RFC4575] is suitable for coordinating sessions including multi-party RTT. The RTT stream between the mixer and a participant is one and the same during the conference. Participants get announced by notifications when participants are joining or leaving, and further user information may be provided. The SSRC of the text to expect from joined users MAY be included in a notification. The notifications MAY be used both for security purposes and for translation to a label for presentation to other users.

#### 2.2.8. Conference control

In managed conferences, control of the real-time text media SHOULD be provided in the same way as other for media, e.g. for muting and unmuting by the direction attributes in SDP [RFC4566].

Note that floor control functions may be of value for RTT users as well as for users of other media in a conference.

### 2.2.9. Media Subtype Registration

This registration is done using the template defined in [RFC6838] and following [RFC4855].

Type name:  
text

Subtype name:  
rex

Required parameters:

rate:  
The RTP timestamp (clock) rate. The only valid value is 1000.

pt:  
a comma-separated list of RTP payload types. Because comma is a special character, the list must be a quoted-string (enclosed in double quotes). Each list element is a mapping of the dynamic payload type number to an embedded Content-type specification for the payload format corresponding to the payload type. The format of the mapping is:

payload-type-number "=" content-type

If the content-type string includes a comma, then the content-type string MUST be a quoted-string. If the content-type string does not include a comma, it MAY still be quoted. Since it is part of the list which must itself be a quoted-string, that means the quotation marks MUST be quoted with backslash quoting as specified in RFC 2045. If the content-type string itself contains a quoted-string, then the requirement for backslash quoting is recursively applied. To specify the text/rex payload format in SDP, the pt parameter is mapped to an a=fmtp attribute by eliminating the parameter name (pt) and changing the commas to slashes. For example:

```
pt = " = \"text/t140;cps=200,text/t140,text/t140\" "
```

Implies the following sdp

```
m=text 49170 RTP/AVP 98 100
a=rtpmap:98 rex/1000
a=fmtp:98 100/100/100
a=rtpmap:100 t140/1000
a=fmtp:100 cps=200
```

Encoding considerations:

binary; see Section 4.8 of [RFC6838].

Security considerations:

See Section 9 of RFC xxxx. [RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Interoperability considerations:

None.

Published specification:

RFC XXXX. [RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Applications which use this media type:

For example: Text conferencing tools, multimedia conferencing tools.Real-time conversational tools.

Fragment identifier considerations:

N/A.

Additional information:

None.

Person & email address to contact for further information:

Gunnar Hellstrom <gunnar.hellstrom@ghaccess.se>

Intended usage:

COMMON

Restrictions on usage:

This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550].

Author:

Gunnar Hellstrom <gunnar.hellstrom@ghaccess.se>

Change controller:

IETF AVTCORE Working Group delegated from the IESG.

#### 2.2.10. SDP considerations

There are receiving RTT implementations which implement RFC 4103 [RFC4103] but not the source separation by the CSRC. Sending mixed text according to the usual CSRC convention from RFC 2198 [RFC2198] to a device implementing only RFC 4103 [RFC4103] and no multi-party mechanism would risk to lead to unreadable presented text. Therefore, in order to negotiate RTT mixing capability according to the "text/rtx" method, all devices supporting "text/rex" for multi-party aware participants SHALL include an SDP media format "text/rex" in the SDP [RFC4566], indicating this format in offers and answers. Multi-party streams using the coding of this section intended for multi-party aware endpoints MUST NOT be sent to devices which have not indicated the "text/rex" format.

Implementations not understanding the "text/rex" format MUST ignore it according to common SDP rules.

The SDP media format defined here, is named "rex", for extended "red". It is intended to be used in "text" media descriptions with "text/rex" and "text/t140" formats. Both formats MUST be declared for the "text/rex" format to be used. It indicates capability to use source indications in the CSRC list and the packet format according to this section. It also indicates ability to receive 150 real-time text characters per second by default.

##### 2.2.10.1. Mapping of media parameters to sdp

The information carried in the media type registration has a specific mapping to fields in the Session Description Protocol (SDP), which is commonly used to describe RTP sessions. When SDP RFC 4566 [RFC4566] is used to specify sessions employing the "text/rex" format, the mapping is as follows:

- \* The media type ("text") goes in SDP "m=" as the media name.
- \* The media subtype (payload format name) goes in SDP "a=rtpmap" as the encoding name. The RTP clock rate in "a=rtpmap" MUST be 1000 for "text/rex".
- \* When the payload type is used with redundancy, the level of redundancy is shown by the number of elements in the slash-separated payload type list in the "fmt" parameter of the "text/rex" media format.

### 2.2.10.2. Security for session control and media

Security SHOULD be applied on both session control and media. In applications where legacy endpoints without security may exist, a negotiation between security and no security SHOULD be applied. If no other security solution is mandated by the application, then RFC 8643 OSRTP[RFC8643] SHOULD be applied to negotiate SRTP media security with DTLS. Most SDP examples below are for simplicity expressed without the security additions. The principles (but not all details) for applying DTLS-SRTP security is shown in a couple of the following examples.

### 2.2.10.3. SDP offer/answer examples

This sections shows some examples of SDP for session negotiation of the real-time text media in SIP sessions. Audio is usually provided in the same session, and sometimes also video. The examples only show the part of importance for the real-time text media.

Offer example for just "text/rex" multi-party capability :

```
m=text 11000 RTP/AVP 101 98
a=rtpmap:98 t140/1000
a=rtpmap:101 rex/1000
a=fmtp:101 98/98/98
```

Answer example from a multi-party capable device

```
m=text 12000 RTP/AVP 101 98
a=rtpmap:98 t140/1000
a=rtpmap:101 rex/1000
a=fmtp:101 98/98/98
```

Offer example for "text/red" and "text/rex" multi-party support:

```
m=text 11000 RTP/AVP 101 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=rtpmap:101 rex/1000
a=fmtp:100 98/98/98
a=fmtp:101 98/98/98
a=rtt-mix-rtp-mixer
```

Answer example from multi-party capable device using "text/rex".

```
m=text 11000 RTP/AVP 101 98
a=rtpmap:98 t140/1000
a=rtpmap:101 rex/1000
a=fmtp:101 98/98/98
```

Offer example for both traditional "text/red" and multi-party format including security:

```
a=fingerprint: SHA-1 \  
4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB  
m=text 11000 RTP/AVP 101 100 98  
a=rtpmap:98 t140/1000  
a=rtpmap:100 red/1000  
a=rtpmap:101 rex/1000  
a=fmtp:100 98/98/98  
a=fmtp:101 98/98/98  
a=rtt-mix-rtp-mixer
```

The "Fingerprint" is sufficient to offer DTLS-SRTP, with the media line still indicating RTP/AVP.

Answer example from a multi-party capable device including security

```
a=fingerprint: SHA-1 \  
FF:FF:FF:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB  
m=text 11000 RTP/AVP 101 98  
a=rtpmap:98 t140/1000  
a=rtpmap:101 rex/1000  
a=fmtp:101 98/98/98
```

With the "fingerprint" the device acknowledges use of SRTP/DTLS.

Answer example from a multi-party unaware device that also does not support security:

```
m=text 12000 RTP/AVP 100 98  
a=rtpmap:98 t140/1000  
a=rtpmap:100 red/1000  
a=fmtp:100 98/98/98
```

A party which has negotiated the "text/rex" format MUST populate the CSRC-list and format the packets according to this section if it acts as an rtp-mixer and sends multi-party text.

A party which has negotiated the "text/rex" capability MUST interpret the contents of the CSRC-list and the packets according to this section in received rtp packets using the corresponding payload type.

A party performing as a mixer, which has not negotiated the "text/rex" format, but negotiated a "text/red" or "text/t140" format in a session with a participant SHOULD, if nothing else is specified for the application, format transmitted text to that participant to be suitable to present on a multi-party unaware endpoint as further specified in section Section 3.2.

A party not performing as a mixer MUST not include the CSRC list if it has a single source of text.

#### 2.2.10.4. Packet examples

This example shows a symbolic flow of packets from a mixer with loss and recovery. A, B and C are sources of RTT. M is the mixer. Pn indicates primary data in source group "n". Rn1 is first redundant generation data and Rn2 is second redundant generation data in source group "n". A1, B1, A2 etc are text chunks (T140blocks) received from the respective sources. X indicates dropped packet between the mixer and a receiver.

```

-----
Seq no 1
CC=1
CSRC list A
R12: Empty
R11: Empty
P1: A1
-----

```

Assuming that earlier packets were received in sequence, text A1 is received from packet 1 and assigned to reception area A.

```

-----
Seq no 2
CC=3
CSRC list C,A
R12 Empty
R11:Empty
P1: C1
R22 Empty
R21: A1
P2: Empty
-----

```

Text C1 is received from packet 2 and assigned to reception area C.

```
X-----  
X Seq no 3  
X CC=2  
X CSRC list C,A  
X R12: Empty  
X R11: C1  
X P1: Empty  
X R22: A1  
X R21: Empty  
X P2: A2  
X-----
```

Packet 3 is assumed to be dropped in network problems

```
X-----  
X Seq no 4  
X CC=3  
X CSRC list C,B,A  
X R12: Empty  
X R11: Empty  
X P1: C2  
X R22: Empty  
X R21: Empty  
X P2: B1  
X R32: Empty  
X R31: A2  
X P3: A3  
X-----
```

Packet 4 is assumed to be dropped in network problems

```
X-----  
X Seq no 5  
X CC=3  
X CSRC list C,B,A  
X R12: Empty  
X R11: C2  
X P1: Empty  
X R22: Empty  
X R21: B1  
X P2: B2  
X R32: A2  
X R31: A3  
X P3: A4  
X-----
```

Packet 5 is assumed to be dropped in network problems

```
-----  
Seq no 6  
CC=3  
CSRC list C,B,A  
R12: C2  
R11: Empty  
P1: Empty  
R22: B1  
R21: B2  
P2: B3  
R32: A3  
R31: A4  
P3: A5  
-----
```

Packet 6 is received. The latest received sequence number was 2. Recovery is therefore tried for 3,4,5. But there is no coverage for seq no 3. A missing text mark (U'FFFD) [T140ad1] is created and appended to the common mixer reception area. A missing text mark (U'FFFD) MAY also be appended in all streams which had text in the packets before and after the gap. That is in this case after A1, and C1, and before B1.

For seqno 4, texts C2, B1 and A3 are recovered from the second generation redundancy and appended to their respective reception areas. For seqno 5, texts B2 and A4 are recovered from the first generation redundancy and appended to their respective reception areas. Primary text B3 and A5 are received and appended to their respective reception areas.

After this sequence, the following has been received: A1,A3, A4, A5; B1, B2, B3; C1, C2. A possible loss is indicated by the general missing text mark in time between A1 and A3, and in the streams after A1 and C1 and before B1.

With only one or two packets lost, there would not be any need to create a missing text marker, and all text would be recovered.

It will be a design decision how to present the missing text markers assigned to the mixer as a source.

#### 2.2.10.5. Performance considerations

This method allows new text from up to 15 sources per packet. A mixer implementing the specification will normally cause a latency of 0 to 150 milliseconds in text from up to 15 simultaneous sources. This performance meets well the realistic requirements for conference and conversational applications for which up to 5 simultaneous sources should not be delayed more than 500 milliseconds by a mixer. In order to achieve good performance, a receiver for multi-party calls SHOULD declare a sufficient CPS value for the "text/t140" format in SDP for the number of allowable characters per second.

As comparison, if the "text/red" format would be used for multi-party communication with its default timing and redundancy, 5 simultaneously sending parties would cause jerky presentation of the text from them in text spurts with 5 seconds intervals. With a reduction of the transmission interval to 150 ms, the time between text spurts for 5 simultaneous sending parties would be 2.5 seconds.

Five simultaneous sending parties may occasionally occur in a conference with one or two main sending parties and three parties giving very brief comments.

The default maximum rate of reception of "text/t140" real-time text is in RFC 4103 [RFC4103] specified to be 30 characters per second. The value MAY be modified in the CPS parameter of the FMTP attribute in the media section for the "text/t140" media. A mixer combining real-time text from a number of sources may have a higher combined flow of text coming from the sources. Endpoints SHOULD therefore specify a suitable higher value for the CPS parameter, corresponding to its real reception capability. A value for CPS of 150 is the default for the "text/t140" stream in the "text/rex" format. See RFC 4103 [RFC4103] for the format and use of the CPS parameter. The same rules apply for the "text/rex" format except for the default value.

#### 2.3. Mixing for multi-party unaware endpoints

A method is specified in this section for cases when the participating endpoint does not implement any solution for multi-party presentation of real-time text. The solution requires the mixer to insert text dividers and readable labels and only send text from one source at a time until a suitable point appears for source change. This solution is a fallback method with functional limitations that acts on the presentation level and is further specified in Section 3.2.

### 3. Presentation level considerations

ITU-T T.140 [T140] provides the presentation level requirements for the RFC 4103 [RFC4103] transport. T.140 [T140] has functions for erasure and other formatting functions and has the following general statement for the presentation:

"The display of text from the members of the conversation should be arranged so that the text from each participant is clearly readable, and its source and the relative timing of entered text is visualized in the display. Mechanisms for looking back in the contents from the current session should be provided. The text should be displayed as soon as it is received."

Strict application of T.140 [T140] is of essence for the interoperability of real-time text implementations and to fulfill the intention that the session participants have the same information of the text contents of the conversation without necessarily having the exact same layout of the conversation.

T.140 [T140] specifies a set of presentation control codes to include in the stream. Some of them are optional. Implementations **MUST** be able to ignore optional control codes that they do not support.

There is no strict "message" concept in real-time text. Line Separator **SHALL** be used as a separator allowing a part of received text to be grouped in presentation. The characters "CRLF" may be used by other implementations as replacement for Line Separator. The "CRLF" combination **SHALL** be erased by just one erasing action, just as the Line Separator. Presentation functions are allowed to group text for presentation in smaller groups than the line separators imply and present such groups with source indication together with text groups from other sources (see the following presentation examples). Erasure has no specific limit by any delimiter in the text stream.

#### 3.1. Presentation by multi-party aware endpoints

A multi-party aware receiving party, presenting real-time text **MUST** separate text from different sources and present them in separate presentation fields. The receiving party **MAY** separate presentation of parts of text from a source in readable groups based on other criteria than line separator and merge these groups in the presentation area when it benefits the user to most easily find and read text from the different participants. The criteria **MAY** e.g. be a received comma, full stop, or other phrase delimiters, or a long pause.

When text is received from multiple original sources simultaneously, the presentation SHOULD provide a view where text is added in multiple places simultaneously.

If the presentation presents text from different sources in one common area, the presenting endpoint SHOULD insert text from the local user ended at suitable points merged with received text to indicate the relative timing for when the text groups were completed. In this presentation mode, the receiving endpoint SHALL present the source of the different groups of text.

A view of a three-party RTT call in chat style is shown in this example .

[Alice] Hi, Alice here.	^
[Bob] Bob as well.	-
[Eve] Hi, this is Eve, calling from Paris. I thought you should be here.	
[Alice] I am coming on Thursday, my performance is not until Friday morning.	
[Bob] And I on Wednesday evening.	
[Alice] Can we meet on Thursday evening?	
[Eve] Yes, definitely. How about 7pm. at the entrance of the restaurant Le Lion Blanc?	
[Eve] we can have dinner and then take a walk	-
<Eve-typing> But I need to be back to the hotel by 11 because I need	v ^
<Bob-typing> I wou	-
of course, I underst	v

Figure 3: Example of a three-party RTT call presented in chat style seen at participant 'Alice's endpoint.

Other presentation styles than the chat style may be arranged.

This figure shows how a coordinated column view MAY be presented.

Bob	Eve	Alice
My flight is to Orly	Hi all, can we plan for the seminar?	I will arrive by TGV. Convenient to the main station.
Eve, will you do your presentation on Friday?	Yes, Friday at 10.	
Fine, wo		We need to meet befo

Figure 4: An example of a coordinated column-view of a three-party session with entries ordered vertically in approximate time-order.

### 3.2. Multi-party mixing for multi-party unaware endpoints

When the mixer has indicated multi-party capability by the "rtt-mix-rtp-mixer" sdp attribute or the "text/rex" format in an SDP negotiation, but the multi-party capability negotiation fails with an endpoint, then the agreed "text/red" or "text/t140" format SHALL be used and the mixer SHOULD compose a best-effort presentation of multi-party real-time text in one stream intended to be presented by an endpoint with no multi-party awareness.

This presentation format has functional limitations and SHOULD be used only to enable participation in multi-party calls by legacy deployed endpoints implementing only RFC 4103 without any multi-party extensions specified in this document.

The principles and procedures below do not specify any new protocol elements. They are instead composed from the information in ITU-T T.140 [T140] and an ambition to provide a best effort presentation on an endpoint which has functions only for two-party calls.

The mixer mixing for multi-party unaware endpoints SHALL compose a simulated limited multi-party RTT view suitable for presentation in one presentation area. The mixer SHALL group text in suitable groups and prepare for presentation of them by inserting a new line between them if the transmitted text did not already end with a new line. A presentable label SHOULD be composed and sent for the source initially in the session and after each source switch. With this procedure the time for source switching is depending on the actions of the users. In order to expedite source switch, a user can for example end its turn with a new line.

### 3.2.1. Actions by the mixer at reception from the call participants

When text is received by the mixer from the different participants, the mixer SHALL recover text from redundancy if any packets are lost. The mark for lost text [T140ad1] SHOULD be inserted in the stream if unrecoverable loss appears. Any Unicode "BOM" characters, possibly used for keep-alive shall be deleted. The time of creation of text (retrieved from the RTP timestamp) SHALL be stored together with the received text from each source in queues for transmission to the recipients.

### 3.2.2. Actions by the mixer for transmission to the recipients

The following procedure SHOULD be applied for each recipient of multi-part text from the mixer.

The text for transmission SHOULD be formatted by the mixer for each receiving user for presentation in one single presentation area. Text received from a participant SHOULD NOT be included in transmission to that participant. When there is text available for transmission from the mixer to a receiving party from more than one participant, the mixer SHOULD switch between transmission of text from the different sources at suitable points in the transmitted stream.

When switching source, the mixer SHOULD insert a line separator if the already transmitted text did not end with a new line (line separator or CRLF). A label SHOULD be composed from information in the CNAME and NAME fields in RTCP reports from the participant to have its text transmitted, or from other session information for that user. The label SHOULD be delimited by suitable characters (e.g. '[' ]') and transmitted. The CSRC SHOULD indicate the selected source. Then text from that selected participant SHOULD be transmitted until a new suitable point for switching source is reached.

Seeking a suitable point for switching source SHOULD be done when there is older text waiting for transmission from any party than the age of the last transmitted text. Suitable points for switching are:

- \* A completed phrase ended by comma
- \* A completed sentence
- \* A new line (line separator or CRLF)
- \* A long pause (e.g. > 10 seconds) in received text from the currently transmitted source

- \* If text from one participant has been transmitted with text from other sources waiting for transmission for a long time (e.g. > 1 minute) and none of the other suitable points for switching has occurred, a source switch MAY be forced by the mixer at next word delimiter, and also if even a word delimiter does not occur within a time (e.g. 15 seconds) after the scan for word delimiter started.

When switching source, the source which has the oldest text in queue SHOULD be selected to be transmitted. A character display count SHOULD be maintained for the currently transmitted source, starting at zero after the label is transmitted for the currently transmitted source.

The status SHOULD be maintained for the latest control code for Select Graphic Rendition (SGR) from each source. If there is an SGR code stored as the status for the current source before the source switch is done, a reset of SGR shall be sent by the sequence SGR 0 [009B 0000 006D] after the new line and before the new label during a source switch. See SGR below for an explanation. This transmission does not influence the display count.

If there is an SGR code stored for the new source after the source switch, that SGR code SHOULD be transmitted to the recipient before the label. This transmission does not influence the display count.

### 3.2.3. Actions on transmission of text

Text from a source sent to the recipient SHOULD increase the display count by one per transmitted character.

### 3.2.4. Actions on transmission of control codes

The following control codes specified by T.140 require specific actions. They SHOULD cause specific considerations in the mixer. Note that the codes presented here are expressed in UCS-16, while transmission is made in UTF-8 transform of these codes.

BEL 0007 Bell Alert in session, provides for alerting during an active session. The display count SHOULD not be altered.

NEW LINE 2028 Line separator. Check and perform a source switch if appropriate. Increase display count by 1.

CR LF 000D 000A A supported, but not preferred way of requesting a new line. Check and perform a source switch if appropriate. Increase display count by 1.

INT ESC 0061 Interrupt (used to initiate mode negotiation procedure). The display count SHOULD not be altered.

SGR 009B Ps 006D Select graphic rendition. Ps is rendition parameters specified in ISO 6429. The display count SHOULD not be altered. The SGR code SHOULD be stored for the current source.

SOS 0098 Start of string, used as a general protocol element introducer, followed by a maximum 256 bytes string and the ST. The display count SHOULD not be altered.

ST 009C String terminator, end of SOS string. The display count SHOULD not be altered.

ESC 001B Escape - used in control strings. The display count SHOULD not be altered for the complete escape code.

Byte order mark "BOM" (U+FEFF) "Zero width, no break space", used for synchronization and keep-alive. SHOULD be deleted from incoming streams. Shall be sent first after session establishment to the recipient. The display count shall not be altered.

Missing text mark (U+FFFD) "Replacement character", represented as a question mark in a rhombus, or if that is not feasible, replaced by an apostrophe ', marks place in stream of possible text loss. SHOULD be inserted by the reception procedure in case of unrecoverable loss of packets. The display count SHOULD be increased by one when sent as for any other character.

SGR If a control code for selecting graphic rendition (SGR), other than reset of the graphic rendition (SGR 0) is sent to a recipient, that control code shall also be stored as status for the source in the storage for SGR status. If a reset graphic rendition (SGR 0) originated from a source is sent, then the SGR status storage for that source shall be cleared. The display count shall not be increased.

BS (U+0008) Back Space, intended to erase the last entered character

by a source. Erasure by backspace cannot always be performed as the erasing party intended. If an erasing action erases all text up to the end of the leading label after a source switch, then the mixer must not transmit more backspaces. Instead it is RECOMMENDED that a letter "X" is inserted in the text stream for each backspace as an indication of the intent to erase more. A new line is usually coded by a Line Separator, but the character combination "CRLF" MAY be used instead. Erasure of a new line is in both cases done by just one erasing action (Backspace). If the display count has a positive value it is decreased by one when the BS is sent. If the display count is at zero, it is not altered.

### 3.2.5. Packet transmission

A mixer transmitting to a multi-party unaware terminal SHOULD send primary data only from one source per packet. The SSRC SHOULD be the SSRC of the mixer. The CSRC list SHOULD contain one member and be the SSRC of the source of the primary data.

### 3.2.6. Functional limitations

When a multi-party unaware endpoint presents a conversation in one display area in a chat style, it inserts source indications for remote text and local user text as they are merged in completed text groups. When an endpoint using this layout receives and presents text mixed for multi-party unaware endpoints, there will be two levels of source indicators for the received text; one generated by the mixer and inserted in a label after each source switch, and another generated by the receiving endpoint and inserted after each switch between local and remote source in the presentation area. This will waste display space and look inconsistent to the reader.

New text can be presented only from one source at a time. Switch of source to be presented takes place at suitable places in the text, such as end of phrase, end of sentence, line separator and inactivity. Therefore the time to switch to present waiting text from other sources may become long and will vary and depend on the actions of the currently presented source.

Erasure can only be done up to the latest source switch. If a user tries to erase more text, the erasing actions will be presented as letter X after the label.

Text loss because of network errors may hit the label between entries from different parties, causing risk for misunderstanding from which source a piece of text is.

These facts makes it strongly RECOMMENDED to implement multi-party awareness in RTT endpoints. The use of the mixing method for multi-party-unaware endpoints should be left for use with endpoints which are impossible to upgrade to become multi-party aware.

### 3.2.7. Example views of presentation on multi-party unaware endpoints

The following pictures are examples of the view on a participant's display for the multi-party-unaware case.

Conference	Alice
[Bob]:My flight is to Orly. [Eve]:Hi all, can we plan for the seminar.	I will arrive by TGV. Convenient to the main station.
[Bob]:Eve, will you do your presentation on Friday? [Eve]:Yes, Friday at 10.	
[Bob]: Fine, wo	We need to meet befo

Figure 5: Alice who has a conference-unaware client is receiving the multi-party real-time text in a single-stream. This figure shows how a coordinated column view MAY be presented on Alice's device.

[Alice] Hi, Alice here.	^
[mix][Bob] Bob as well.	-
[Eve] Hi, this is Eve, calling from Paris I thought you should be here.	
[Alice] I am coming on Thursday, my performance is not until Friday morning.	
[mix][Bob] And I on Wednesday evening.	
[Eve] we can have dinner and then walk	
[Eve] But I need to be back to the hotel by 11 because I need	
of course, I underst	-
	v

Figure 6: An example of a view of the multi-party unaware presentation in chat style. Alice is the local user.

#### 4. Gateway Considerations

##### 4.1. Gateway considerations with Textphones (e.g. TTYs).

Multi-party RTT sessions may involve gateways of different kinds. Gateways involved in setting up sessions SHALL correctly reflect the multi-party capability or unawareness of the combination of the gateway and the remote endpoint beyond the gateway.

One case that may occur is a gateway to PSTN for communication with textphones (e.g. TTYs). Textphones are limited devices with no multi-party awareness, and it SHOULD therefore be suitable for the gateway to not indicate multi-party awareness for that case. Another solution is that the gateway indicates multi-party capability towards the mixer, and includes the multi-party mixer function for multi-party unaware endpoints itself. This solution makes it possible to make adaptations for the functional limitations of the textphone (TTY).

More information on gateways to textphones (TTYs) is found in RFC 5194[RFC5194]

#### 4.2. Gateway considerations with WebRTC.

Gateway operation to real-time text in WebRTC may also be required. In WebRTC, RTT is specified in draft-ietf-mmusic-t140-usage-data-channel[I-D.ietf-mmusic-t140-usage-data-channel].

A multi-party bridge may have functionality for communicating by RTT both in RTP streams with RTT and WebRTC t140 data channels. Other configurations may consist of a multi-party bridge with either technology for RTT transport and a separate gateway for conversion of the text communication streams between RTP and t140 data channel.

In WebRTC, it is assumed that for a multi-party session, one t140 data channel is established for each source from a gateway or bridge to each participant. Each participant also has a data channel with two-way connection with the gateway or bridge.

The t140 channel used both ways is for text from the WebRTC user and from the bridge or gateway itself to the WebRTC user. The label parameter of this t140 channel is used as NAME field in RTCP to participants on the RTP side. The other t140 channels are only for text from other participants to the WebRTC user.

When a new participant has entered the session with RTP transport of rtt, a new t140 channel SHOULD be established to WebRTC users with the label parameter composed from the NAME field in RTCP on the RTP side.

When a new participant has entered the multi-party session with RTT transport in a WebRTC t140 data channel, the new participant SHOULD be announced by a notification to RTP users. The label parameter from the WebRTC side SHOULD be used as the NAME RTCP field on the RTP side, or other available session information.

#### 5. Updates to RFC 4102 and RFC 4103

This document updates RFC 4102[RFC4102] and RFC 4103[RFC4103] by introducing an sdp media attribute "rtt-mix-rtp-mixer" for negotiation of multi-party mixing capability with the [RFC4103] format and an extended packet format "text/rex" for the enhanced performance multi-party mixing case and more strict rules for the use of redundancy, and population of the CSRC list in the packets. Implications for the CSRC list use from RFC 2198[RFC2198] is not in effect for the "text/rex" format.

The update is in line with the statement in RFC 4103 section 4, saying that "Forward Error Correction mechanisms, ..., or any other mechanism with the purpose of increasing the reliability of text transmission, MAY be used as an alternative or complement to redundancy."

## 6. Congestion considerations

The congestion considerations and recommended actions from RFC 4103 [RFC4103] are valid also in multi-party situations.

The first action in case of congestion SHOULD be to temporarily increase the transmission interval up to two seconds.

## 7. Acknowledgements

James Hamlin for format input.

## 8. IANA Considerations

### 8.1. Registration of the "rtt-mix-rtp-mixer" sdp media attribute

[RFC EDITOR NOTE: Please replace all instances of RFCXXXX with the RFC number of this document.]

IANA is asked to register the new sdp attribute "rtt-mix-rtp-mixer".

Contact name: IESG

Contact email: iesg@ietf.org

Attribute name: rtt-mix-rtp-mixer

Attribute syntax: a=rtt-mix-rtp-mixer

Attribute semantics: See RFCXXXX Section 2.1.1

Attribute value: none

Usage level: media

Purpose: Indicate support by mixer or endpoint of multi-party mixing for real-time text transmission, using a common RTP-stream for transmission of text from a number of sources mixed with one source at a time and the source indicated in a single CSRC-list member.

Charset Dependent: no

O/A procedure: See RFCXXXX Section 2.1.19

Mux Category: normal

Reference: RFCXXXX

## 8.2. Registration of "text/rex" media subtype

The IANA is requested to register the media type "text/rex" as specified in Section 2.2.9. The media type is also requested to be added to the IANA registry for "RTP Payload Format Media Types" <<http://www.iana.org/assignments/rtp-parameters>>.

## 9. Security Considerations

The RTP-mixer model requires the mixer to be allowed to decrypt, pack and encrypt secured text from the conference participants. Therefore the mixer needs to be trusted. This is similar to the situation for central mixers of audio and video.

The requirement to transfer information about the user in RTCP reports in SDES, CNAME and NAME fields, and in conference notifications, for creation of labels may have privacy concerns as already stated in RFC 3550 [RFC3550], and may be restricted of privacy reasons. The receiving user will then get a more symbolic label for the source.

## 10. Change history

### 10.1. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-07

Added a method based on the "text/red" format and single source per packet, negotiated by the "rtt-mix-rtp-mixer" sdp attribute.

Added reasoning and recommendation about indication of loss.

The highest number of sources in one packet is 15, not 16. Changed.

Added in information on update to RFC 4103 that RFC 4103 explicitly allows addition of FEC method. The redundancy is a kind of forward error correction..

### 10.2. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-06

Improved definitions list format.

The format of the media subtype parameters is made to match the requirements.

The mapping of media subtype parameters to sdp is included.

The CPS parameter belongs to the t140 subtype and does not need to be registered here.

10.3. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-05  
nomenclature and editorial improvements

"this document" used consistently to refer to this document.

10.4. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-04

'Redundancy header' renamed to 'data header'.

More clarifications added.

Language and figure number corrections.

10.5. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-03

Mention possible need to mute and raise hands as for other media.  
---done ----

Make sure that use in two-party calls is also possible and explained.  
- may need more wording -

Clarify the RTT is often used together with other media. --done--

Tell that text mixing is N-1. A users own text is not received in  
the mix. -done-

In 3. correct the interval to: A "text/rex" transmitter SHOULD send  
packets distributed in time as long as there is something (new or  
redundant T140blocks) to transmit. The maximum transmission interval  
SHOULD then be 300 ms. It is RECOMMENDED to send a packet to a  
receiver as soon as new text to that receiver is available, as long  
as the time after the latest sent packet to the same receiver is more  
than 150 ms, and also the maximum character rate to the receiver is  
not exceeded. The intention is to keep the latency low while keeping  
a good protection against text loss in bursty packet loss conditions.  
-done-

In 1.3 say that the format is used both ways. -done-

In 13.1 change presentation area to presentation field so that reader  
does not think it shall be totally separated. -done-

In Performance and intro, tell the performance in number of simultaneous sending users and introduced delay 16, 150 vs requirements 5 vs 500. -done --

Clarify redundancy level per connection. -done-

Timestamp also for the last data header. To make it possible for all text to have time offset as for transmission from the source. Make that header equal to the others. -done-

Mixer always use the CSRC list, even for its own BOM. -done-

Combine all talk about transmission interval (300 ms vs when text has arrived) in section 3 in one paragraph or close to each other. -done-

Documents the goal of good performance with low delay for 5 simultaneous typers in the introduction. -done-

Describe better that only primary text shall be sent on to receivers. Redundancy and loss must be resolved by the mixer. -done-

#### 10.6. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-02

SDP and better description and visibility of security by OSRTP RFC 8634 needed.

The description of gatewaying to WebRTC extended.

The description of the data header in the packet is improved.

#### 10.7. Changes to draft-ietf-avtcore-multi-party-rtt-mix-01

2,5,6 More efficient format "text/rex" introduced and attribute a=rtt-mix deleted.

3. Brief about use of OSRTP for security included- More needed.

4. Brief motivation for the solution and why not rtp-translator is used added to intro.

7. More limitations for the multi-party unaware mixing method inserted.

8. Updates to RFC 4102 and 4103 more clearly expressed.

9. Gateway to WebRTC started. More needed.

10.8. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-03 to draft-ietf-avtcore-multi-party-rtt-mix-00

Changed file name to draft-ietf-avtcore-multi-party-rtt-mix-00

Replaced CDATA in IANA registration table with better coding.

Converted to xml2rfc version 3.

10.9. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-02 to -03

Changed company and e-mail of the author.

Changed title to "RTP-mixer formatting of multi-party Real-time text" to better match contents.

Check and modification where needed of use of RFC 2119 words SHALL etc.

More about the CC value in sections on transmitters and receivers so that 1-to-1 sessions do not use the mixer format.

Enhanced section on presentation for multi-party-unaware endpoints

A paragraph recommending CPS=150 inserted in the performance section.

10.10. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-01 to -02

In Abstract and 1. Introduction: Introduced wording about regulatory requirements.

In section 5: The transmission interval is decreased to 100 ms when there is text from more than one source to transmit.

In section 11 about SDP negotiation, a SHOULD-requirement is introduced that the mixer should make a mix for multi-party unaware endpoints if the negotiation is not successful. And a reference to a later chapter about it.

The presentation considerations chapter 14 is extended with more information about presentation on multi-party aware endpoints, and a new section on the multi-party unaware mixing with low functionality but SHOULD a be implemented in mixers. Presentation examples are added.

A short chapter 15 on gateway considerations is introduced.

Clarification about the text/t140 format included in chapter 10.

This sentence added to the chapter 10 about use without redundancy.  
"The text/red format SHOULD be used unless some other protection against packet loss is utilized, for example a reliable network or transport."

Note about deviation from RFC 2198 added in chapter 4.

In chapter 9. "Use with SIP centralized conferencing framework" the following note is inserted: Note: The CSRC-list in an RTP packet only includes participants who's text is included in one or more text blocks. It is not the same as the list of participants in a conference. With audio and video media, the CSRC-list would often contain all participants who are not muted whereas text participants that don't type are completely silent and so don't show up in RTP packet CSRC-lists.

#### 10.11. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-00 to -01

Editorial cleanup.

Changed capability indication from fntp-parameter to SDP attribute "rtt-mix".

Swapped order of redundancy elements in the example to match reality.

Increased the SDP negotiation section

## 11. References

### 11.1. Normative References

[I-D.ietf-mmusic-t140-usage-data-channel]

Holmberg, C. and G. Hellstrom, "T.140 Real-time Text Conversation over WebRTC Data Channels", Work in Progress, Internet-Draft, draft-ietf-mmusic-t140-usage-data-channel-14, 10 April 2020, <<https://tools.ietf.org/html/draft-ietf-mmusic-t140-usage-data-channel-14>>.

[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>>.

- [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>>.
- [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>>.
- [RFC4102] Jones, P., "Registration of the text/red MIME Sub-Type", RFC 4102, DOI 10.17487/RFC4102, June 2005, <<https://www.rfc-editor.org/info/rfc4102>>.
- [RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <<https://www.rfc-editor.org/info/rfc4103>>.
- [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>>.
- [RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <<https://www.rfc-editor.org/info/rfc4855>>.
- [RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <<https://www.rfc-editor.org/info/rfc5764>>.
- [RFC6263] Marjou, X. and A. Sollaud, "Application Mechanism for Keeping Alive the NAT Mappings Associated with RTP / RTP Control Protocol (RTCP) Flows", RFC 6263, DOI 10.17487/RFC6263, June 2011, <<https://www.rfc-editor.org/info/rfc6263>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/info/rfc6838>>.

- [RFC8643] Johnston, A., Aboba, B., Hutton, A., Jesske, R., and T. Stach, "An Opportunistic Approach for Secure Real-time Transport Protocol (OSRTP)", RFC 8643, DOI 10.17487/RFC8643, August 2019, <<https://www.rfc-editor.org/info/rfc8643>>.
- [T140] ITU-T, "Recommendation ITU-T T.140 (02/1998), Protocol for multimedia application text conversation", February 1998, <<https://www.itu.int/rec/T-REC-T.140-199802-I/en>>.
- [T140ad1] ITU-T, "Recommendation ITU-T.140 Addendum 1 - (02/2000), Protocol for multimedia application text conversation", February 2000, <<https://www.itu.int/rec/T-REC-T.140-200002-I!Add1/en>>.

## 11.2. Informative References

- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, DOI 10.17487/RFC4353, February 2006, <<https://www.rfc-editor.org/info/rfc4353>>.
- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, Ed., "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, DOI 10.17487/RFC4575, August 2006, <<https://www.rfc-editor.org/info/rfc4575>>.
- [RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, DOI 10.17487/RFC4579, August 2006, <<https://www.rfc-editor.org/info/rfc4579>>.
- [RFC5194] van Wijk, A., Ed. and G. Gybels, Ed., "Framework for Real-Time Text over IP Using the Session Initiation Protocol (SIP)", RFC 5194, DOI 10.17487/RFC5194, June 2008, <<https://www.rfc-editor.org/info/rfc5194>>.

## Author's Address

Gunnar Hellstrom  
Gunnar Hellstrom Accessible Communication  
Esplanaden 30  
SE-13670 Vendelso  
Sweden

Email: [gunnar.hellstrom@ghaccess.se](mailto:gunnar.hellstrom@ghaccess.se)

AVTCore  
Internet-Draft  
Updates: 4103 (if approved)  
Intended status: Standards Track  
Expires: 27 November 2021

G. Hellstrom  
Gunnar Hellstrom Accessible Communication  
26 May 2021

RTP-mixer formatting of multiparty Real-time text  
draft-ietf-avtc core-multi-party-rtt-mix-20

Abstract

This document provides enhancements for RFC 4103 real-time text mixing suitable for a centralized conference model that enables source identification and rapidly interleaved transmission of text from different sources. The intended use is for real-time text mixers and participant endpoints capable of providing an efficient presentation or other treatment of a multiparty real-time text session. The specified mechanism builds on the standard use of the Contributing Source (CSRC) list in the Realtime Protocol (RTP) packet for source identification. The method makes use of the same "text/t140" and "text/red" formats as for two-party sessions.

Solutions using multiple RTP streams in the same RTP session are briefly mentioned, as they could have some benefits over the RTP-mixer model. The possibility to implement the solution in a wide range of existing RTP implementations made the RTP-mixer model be selected to be fully specified in this document.

A capability exchange is specified so that it can be verified that a mixer and a participant can handle the multiparty-coded real-time text stream using the RTP-mixer method. The capability is indicated by use of an RFC 8866 Session Description Protocol (SDP) media attribute "rtt-mixer".

The document updates RFC 4103 "RTP Payload for Text Conversation".

A specification of how a mixer can format text for the case when the endpoint is not multiparty-aware is also provided.

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 <https://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 27 November 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

- 1. Introduction . . . . . 4
  - 1.1. Terminology . . . . . 6
  - 1.2. Selected solution and considered alternatives . . . . . 7
  - 1.3. Intended application . . . . . 9
- 2. Overview of the two specified solutions and selection of method . . . . . 10
  - 2.1. The RTP-mixer-based solution for multiparty-aware endpoints . . . . . 10
  - 2.2. Mixing for multiparty-unaware endpoints . . . . . 11
  - 2.3. Offer/answer considerations . . . . . 11
  - 2.4. Actions depending on capability negotiation result . . . 13
- 3. Details for the RTP-mixer-based mixing method for multiparty-aware endpoints . . . . . 13
  - 3.1. Use of fields in the RTP packets . . . . . 13
  - 3.2. Initial transmission of a BOM character . . . . . 14
  - 3.3. Keep-alive . . . . . 14
  - 3.4. Transmission interval . . . . . 14
  - 3.5. Only one source per packet . . . . . 15
  - 3.6. Do not send received text to the originating source . . . 15
  - 3.7. Clean incoming text . . . . . 16

3.8.	Redundant transmission principles . . . . .	16
3.9.	Text placement in packets . . . . .	16
3.10.	Empty T140blocks . . . . .	17
3.11.	Creation of the redundancy . . . . .	17
3.12.	Timer offset fields . . . . .	18
3.13.	Other RTP header fields . . . . .	18
3.14.	Pause in transmission . . . . .	18
3.15.	RTCP considerations . . . . .	19
3.16.	Reception of multiparty contents . . . . .	19
3.17.	Performance considerations . . . . .	21
3.18.	Security for session control and media . . . . .	21
3.19.	SDP offer/answer examples . . . . .	22
3.20.	Packet sequence example from interleaved transmission . . . . .	23
3.21.	Maximum character rate "cps" . . . . .	26
4.	Presentation level considerations . . . . .	26
4.1.	Presentation by multiparty-aware endpoints . . . . .	27
4.2.	Multiparty mixing for multiparty-unaware endpoints . . . . .	29
5.	Relation to Conference Control . . . . .	35
5.1.	Use with SIP centralized conferencing framework . . . . .	36
5.2.	Conference control . . . . .	36
6.	Gateway Considerations . . . . .	36
6.1.	Gateway considerations with Textphones . . . . .	36
6.2.	Gateway considerations with WebRTC . . . . .	36
7.	Updates to RFC 4103 . . . . .	37
8.	Congestion considerations . . . . .	38
9.	IANA Considerations . . . . .	38
9.1.	Registration of the "rtt-mixer" SDP media attribute . . . . .	38
10.	Security Considerations . . . . .	39
11.	Change history . . . . .	40
11.1.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-20 . . . . .	40
11.2.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-19 . . . . .	40
11.3.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-18 . . . . .	40
11.4.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-17 . . . . .	40
11.5.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-16 . . . . .	40
11.6.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-15 . . . . .	41
11.7.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-14 . . . . .	41
11.8.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-13 . . . . .	41
11.9.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-12 . . . . .	42

11.10. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-11 . . . . .	42
11.11. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-10 . . . . .	42
11.12. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-09 . . . . .	42
11.13. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-08 . . . . .	43
11.14. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-07 . . . . .	43
11.15. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-06 . . . . .	43
11.16. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-05 . . . . .	43
11.17. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-04 . . . . .	43
11.18. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-03 . . . . .	44
11.19. Changes included in	
draft-ietf-avtcore-multi-party-rtt-mix-02 . . . . .	45
11.20. Changes to draft-ietf-avtcore-multi-party-rtt-mix-01 . .	45
11.21. Changes from	
draft-hellstrom-avtcore-multi-party-rtt-source-03 to	
draft-ietf-avtcore-multi-party-rtt-mix-00 . . . . .	45
11.22. Changes from	
draft-hellstrom-avtcore-multi-party-rtt-source-02 to	
-03 . . . . .	45
11.23. Changes from	
draft-hellstrom-avtcore-multi-party-rtt-source-01 to	
-02 . . . . .	46
11.24. Changes from	
draft-hellstrom-avtcore-multi-party-rtt-source-00 to	
-01 . . . . .	47
12. References . . . . .	47
12.1. Normative References . . . . .	47
12.2. Informative References . . . . .	48
Acknowledgements . . . . .	49
Author's Address . . . . .	49

## 1. Introduction

"RTP Payload for Text Conversation" [RFC4103] specifies use of the Real-Time Transport Protocol (RTP) [RFC3550] for transmission of real-time text (RTT) and the "text/t140" format. It also specifies a redundancy format "text/red" for increased robustness. The "text/red" format is registered in [RFC4102].

Real-time text is usually provided together with audio and sometimes with video in conversational sessions.

A requirement related to multiparty sessions from the presentation level standard T.140 [T140] for real-time text is: "The display of text from the members of the conversation should be arranged so that the text from each participant is clearly readable, and its source and the relative timing of entered text is visualized in the display."

Another requirement is that the mixing procedure must not introduce delays in the text streams that are experienced to be disturbing the real-time experience of the receiving users.

Use of RTT is increasing, and specifically, use in emergency calls is increasing. Emergency call use requires multiparty mixing because it is common that one agent needs to transfer the call to another specialized agent but is obliged to stay on the call at least to verify that the transfer was successful. Mixer implementations for RFC 4103 "RTP Payload for Text Conversation" can use traditional RFC 3550 RTP functions for mixing and source identification, but the performance of the mixer when giving turns for the different sources to transmit is limited when using the default transmission characteristics with redundancy.

The redundancy scheme of [RFC4103] enables efficient transmission of earlier transmitted redundant text in packets together with new text. However, the redundancy header format has no source indicators for the redundant transmissions. The redundant parts in a packet must therefore be from the same source as the new text. The recommended transmission is one new and two redundant generations of text (T140blocks) in each packet and the recommended transmission interval for two-party use is 300 ms.

Real-time text mixers for multiparty sessions need to include the source with each transmitted group of text from a conference participant so that the text can be transmitted interleaved with text groups from different sources at the rate they are created. This enables the text groups to be presented by endpoints in suitable grouping with other text from the same source.

The presentation can then be arranged so that text from different sources can be presented in real-time and easily read. At the same time it is possible for a reading user to perceive approximately when the text was created in real time by the different parties. The transmission and mixing is intended to be done in a general way, so that presentation can be arranged in a layout decided by the endpoint.

There are existing implementations of RFC 4103 in endpoints without the updates from this document. These will not be able to receive and present real-time text mixed for multiparty-aware endpoints.

A negotiation mechanism is therefore needed for verification if the parties are able to handle a common method for multiparty transmission and agreeing on using that method.

A fallback mixing procedure is also needed for cases when the negotiation result indicates that a receiving endpoint is not capable of handling the mixed format. Multiparty-unaware endpoints would possibly otherwise present all received multiparty mixed text as if it came from the same source regardless of any accompanying source indication coded in fields in the packet. Or they may have other undesirable ways of acting on the multiparty content. The fallback method is called the mixing procedure for multiparty-unaware endpoints. The fallback method is naturally not expected to meet all performance requirements placed on the mixing procedure for multiparty-aware endpoints.

The document updates [RFC4103] by introducing an attribute for declaring support of the RTP-mixer-based multiparty mixing case and rules for source indications and interleaving of text from different sources.

### 1.1. Terminology

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 above.

The terms Source Description (SDES), Canonical name (CNAME), Name (NAME), Synchronization Source (SSRC), Contributing Source (CSRC), CSRC list, CSRC count [CC], Real-Time control protocol (RTCP), RTP-mixer, RTP-translator are defined in [RFC3550].

The term "T140block" is defined in [RFC4103] to contain one or more T.140 code elements.

"TTY" stands for a textphone type used in North America.

Web based real-time communication (WebRTC) is specified by the World Wide Web Consortium (W3C) and IETF. See [RFC8825].

"DTLS-SRTP" is a Datagram Transport Layer Security (DTLS) extension for use with Secure Real-Time Transport Protocol/Secure Real-Time Control Protocol (SRTP/SRTCP) specified in [RFC5764].

"multiparty-aware" describes an endpoint receiving real-time text from multiple sources through a common conference mixer being able to present the text in real-time, separated by source, and presented so that a user can get an impression of the approximate relative timing of text from different parties.

"multiparty-unaware" describes an endpoint not itself being able to separate text from different sources when received through a common conference mixer.

## 1.2. Selected solution and considered alternatives

A number of alternatives were considered when searching an efficient and easily implemented multiparty method for real-time text. This section explains a few of them briefly.

### Multiple RTP streams, one per participant

One RTP stream per source would be sent in the same RTP session with the "text/red" format. From some points of view, use of multiple RTP streams, one for each source, sent in the same RTP session would be efficient, and would use exactly the same packet format as [RFC4103] and the same payload type. A couple of relevant scenarios using multiple RTP-streams are specified in "RTP Topologies" [RFC7667]. One possibility of special interest is the Selective Forwarding Middlebox (SFM) topology specified in RFC 7667 section 3.7 that could enable end-to-end encryption. In contrast to audio and video, real-time text is only transmitted when the users actually transmit information. Thus, an SFM solution would not need to exclude any party from transmission under normal conditions. In order to allow the mixer to convey the packets with the payload preserved and encrypted, an SFM solution would need to act on some specific characteristics of the "text/red" format. The redundancy headers are part of the payload, so the receiver would need to just assume that the payload type number in the redundancy header is for "text/t140". The characters per second parameter (cps) would need to act per stream. The relation between the SSRC and the source would need to be conveyed in some specified way, e.g., in the CSRC. Recovery and loss detection would preferably be based on sequence number gap detection. Thus, sequence number gaps in the incoming stream to the mixer would need to be reflected in the stream to the participant, with no new gaps created by the mixer. However, the RTP implementation in both mixers and endpoints need to support multiple streams in the same RTP session in order to use this

mechanism. For best deployment opportunity, it should be possible to upgrade existing endpoint solutions to be multiparty-aware with a reasonable effort. There is currently a lack of support for multi-stream RTP in certain implementations. This fact led to this solution being only briefly mentioned in this document as an option for further study.

#### RTP-mixer-based method for multiparty-aware endpoints

The "text/red" format in RFC 4103 is sent with a shorter transmission interval with the RTP-mixer method and indicating the source in the CSRC field. The "text/red" format with a "text/t140" payload in a single RTP stream can be sent when text is available from the call participants instead of at the regular 300 ms. Transmission of packets with text from different sources can then be done smoothly while simultaneous transmission occurs as long as it is not limited by the maximum character rate "cps". With ten participants sending text simultaneously, the switching and transmission performance is good. With more simultaneously sending participants, and with receivers having the default capacity there will be a noticeable jerkiness and delay in text presentation. The jerkiness will be more expressed the more participants who send text simultaneously. Two seconds jerkiness will be noticeable and slightly unpleasant, but it corresponds in time to what typing humans often cause by hesitation or changing position while typing. A benefit of this method is that no new packet format needs to be introduced and implemented. Since simultaneous typing by more than two parties is expected to be very rare as described in Section 1.3, this method can be used successfully with good performance. Recovery of text in case of packet loss is based on analysis of timestamps of received redundancy versus earlier received text. Negotiation is based on a new SDP media attribute "rtt-mixer". This method is selected to be the main one specified in this document.

#### Multiple sources per packet

A new "text" media subtype would be specified with up to 15 sources in each packet. The mechanism would make use of the RTP mixer model specified in RTP [RFC3550]. The sources are indicated in strict order in the CSRC list of the RTP packets. The CSRC list can have up to 15 members. Therefore, text from up to 15 sources can be included in each packet. Packets are normally sent with 300 ms intervals. The mean delay will be 150 ms. A new redundancy packet format is specified. This method would result in good performance, but would require standardization and implementation of new releases in the target technologies that would take more time than desirable to complete. It was therefore not selected to be included in this document.

#### Mixing for multiparty-unaware endpoints

Presentation of text from multiple parties is prepared by the mixer in one single stream. It is desirable to have a method that does not require any modifications in existing user devices implementing RFC 4103 for RTT without explicit support of multiparty sessions. This is possible by having the mixer insert a new line and a text formatted source label before each switch of text source in the stream. Switch of source can only be done in places in the text where it does not disturb the perception of the contents. Text from only one source can be presented in real time at a time. The delay will therefore vary. The method also has other limitations, but is included in this document as a fallback method. In calls where parties take turns properly by ending their entries with a new line, the limitations will have limited influence on the user experience. When only two parties send text, these two will see the text in real time with no delay. This method is specified as a fallback method in this document.

#### RTT transport in WebRTC

Transport of real-time text in the WebRTC technology is specified to use the WebRTC data channel in [RFC8865]. That specification contains a section briefly describing its use in multiparty sessions. The focus of this document is RTP transport. Therefore, even if the WebRTC transport provides good multiparty performance, it is just mentioned in this document in relation to providing gateways with multiparty capabilities between RTP and WebRTC technologies.

### 1.3. Intended application

The method for multiparty real-time text specified in this document is primarily intended for use in transmission between mixers and endpoints in centralized mixing configurations. It is also applicable between mixers. An often mentioned application is for emergency service calls with real-time text and voice, where a call taker wants to make an attended handover of a call to another agent, and stay to observe the session. Multimedia conference sessions with support for participants to contribute in text is another application. Conferences with central support for speech-to-text conversion is yet another mentioned application.

In all these applications, normally only one participant at a time will send long text utterances. In some cases, one other participant will occasionally contribute with a longer comment simultaneously. That may also happen in some rare cases when text is interpreted to text in another language in a conference. Apart from these cases, other participants are only expected to contribute with very brief utterings while others are sending text.

Users expect that the text they send is presented in real-time in a readable way to the other participants even if they send simultaneously with other users and even when they make brief edit operations of their text by backspacing and correcting their text.

Text is supposed to be human generated, by some text input means, such as typing on a keyboard or using speech-to-text technology. Occasional small cut-and-paste operations may appear even if that is not the initial purpose of real-time text.

The real-time characteristics of real-time text is essential for the participants to be able to contribute to a conversation. If the text is too much delayed from typing a letter to its presentation, then, in some conference situations, the opportunity to comment will be gone and someone else will grab the turn. A delay of more than one second in such situations is an obstacle for good conversation.

## 2. Overview of the two specified solutions and selection of method

This section contains a brief introduction of the two methods specified in this document.

### 2.1. The RTP-mixer-based solution for multiparty-aware endpoints

This method specifies negotiated use of the RFC 4103 format for multiparty transmission in a single RTP stream. The main purpose of this document is to specify a method for true multiparty real-time text mixing for multiparty-aware endpoints that can be widely deployed. The RTP-mixer-based method makes use of the current format for real-time text in [RFC4103]. It is an update of RFC 4103 by a clarification on one way to use it in the multiparty situation. That is done by completing a negotiation for this kind of multiparty capability and by interleaving packets from different sources. The source is indicated in the CSRC element in the RTP packets. Specific considerations are made to be able to recover text after packet loss.

The detailed procedures for the RTP-mixer-based multiparty-aware case are specified in Section 3.

Please use [RFC4103] as reference when reading the specification.

## 2.2. Mixing for multiparty-unaware endpoints

A method is also specified in this document for cases when the endpoint participating in a multiparty call does not itself implement any solution, or not the same, as the mixer. The method requires the mixer to insert text dividers and readable labels and only send text from one source at a time until a suitable point appears for source change. This solution is a fallback method with functional limitations. It acts on the presentation level.

A mixer SHOULD by default format and transmit text to a call participant to be suitable to present on a multiparty-unaware endpoint which has not negotiated any method for true multiparty RTT handling, but negotiated a "text/red" or "text/t140" format in a session. This SHOULD be done if nothing else is specified for the application in order to maintain interoperability. Section 4.2 specifies how this mixing is done.

## 2.3. Offer/answer considerations

RTP Payload for Text Conversation [RFC4103] specifies use of RTP [RFC3550], and a redundancy format "text/red" for increased robustness of real-time text transmission. This document updates [RFC4103] by introducing a capability negotiation for handling multiparty real-time text, a way to indicate the source of transmitted text, and rules for efficient timing of the transmissions interleaved from different sources.

The capability negotiation for the "RTP-mixer-based multiparty method" is based on use of the SDP media attribute "rtt-mixer".

The syntax is as follows:  
"a=rtt-mixer"

If any other method for RTP-based multiparty real-time text gets specified by additional work, it is assumed that it will be recognized by some specific SDP feature exchange.

### 2.3.1. Initial offer

A party intending to set up a session and being willing to use the RTP-mixer-based method of this specification for sending or receiving or both sending and receiving real-time text SHALL include the "rtt-mixer" SDP attribute in the corresponding "text" media section in the initial offer.

The party MAY indicate capability for both the RTP-mixer-based method of this specification and other methods.

When the offeror has sent the offer including the "rtt-mixer" attribute, it MUST be prepared to receive and handle real-time text formatted according to both the method for multiparty-aware parties specified in Section 3 in this specification and two-party formatted real-time text.

#### 2.3.2. Answering the offer

A party receiving an offer containing the "rtt-mixer" SDP attribute and being willing to use the RTP-mixer-based method of this specification for sending or receiving or both sending and receiving SHALL include the "rtt-mixer" SDP attribute in the corresponding "text" media section in the answer.

If the offer did not contain the "rtt-mixer" attribute, the answer MUST NOT contain the "rtt-mixer" attribute.

Even when the "rtt-mixer" attribute is successfully negotiated, the parties MAY send and receive two-party coded real-time text.

An answer MUST NOT include acceptance of more than one method for multiparty real-time text in the same RTP session.

When the answer including acceptance is transmitted, the answerer MUST be prepared to act on received text in the negotiated session according to the method for multiparty-aware parties specified in Section 3 of this specification. Reception of text for a two-party session SHALL also be supported.

#### 2.3.3. Offeror processing the answer

When the answer is processed by the offeror, it MUST act as specified in Section 2.4

#### 2.3.4. Modifying a session

A session MAY be modified at any time by any party offering a modified SDP with or without the "rtt-mixer" SDP attribute expressing a desired change in the support of multiparty real-time text.

If the modified offer adds indication of support for multiparty real-time text by including the "rtt-mixer" SDP attribute, the procedures specified in the previous subsections SHALL be applied.

If the modified offer deletes indication of support for multiparty real-time text by excluding the "rtt-mixer" SDP attribute, the answer MUST NOT contain the "rtt-mixer" attribute. After processing this SDP exchange, the parties MUST NOT send real-time text formatted for multiparty-aware parties according to this specification.

#### 2.4. Actions depending on capability negotiation result

A transmitting party SHALL send text according to the RTP-mixer-based multiparty method only when the negotiation for that method was successful and when it conveys text for another source. In all other cases, the packets SHALL be populated and interpreted as for a two-party session.

A party which has negotiated the "rtt-mixer" SDP media attribute MUST populate the CSRC-list, and format the packets according to Section 3 if it acts as an rtp-mixer and sends multiparty text.

A party which has negotiated the "rtt-mixer" SDP media attribute MUST interpret the contents of the "CC" field, the CSRC-list and the packets according to Section 3 in received RTP packets in the corresponding RTP stream.

A party which has not successfully completed the negotiation of the "rtt-mixer" SDP media attribute MUST NOT transmit packets interleaved from different sources in the same RTP stream as specified in Section 3. If the party is a mixer and did declare the "rtt-mixer" SDP media attribute, it SHOULD perform the procedure for multiparty-unaware endpoints. If the party is not a mixer, it SHOULD transmit as in a two-party session according to [RFC4103].

### 3. Details for the RTP-mixer-based mixing method for multiparty-aware endpoints

#### 3.1. Use of fields in the RTP packets

The CC field SHALL show the number of members in the CSRC list, which SHALL be one (1) in transmissions from a mixer when conveying text from other sources in a multiparty session, and otherwise 0.

When text is conveyed by a mixer during a multiparty session, a CSRC list SHALL be included in the packet. The single member in the CSRC-list SHALL contain the SSRC of the source of the T140blocks in the packet.

When redundancy is used, the RECOMMENDED level of redundancy is to use one primary and two redundant generations of T140blocks. In some cases, a primary or redundant T140block is empty, but is still represented by a member in the redundancy header.

In other regards, the contents of the RTP packets are equal to what is specified in [RFC4103].

### 3.2. Initial transmission of a BOM character

As soon as a participant is known to participate in a session with another entity and is available for text reception, a Unicode Byte-Order Mark (BOM) character SHALL be sent to it by the other entity according to the procedures in this section. This is useful in many configurations to open ports and firewalls and setting up the connection between the application and the network. If the transmitter is a mixer, then the source of this character SHALL be indicated to be the mixer itself.

Note that the BOM character SHALL be transmitted with the same redundancy procedures as any other text.

### 3.3. Keep-alive

After that, the transmitter SHALL send keep-alive traffic to the receiver(s) at regular intervals when no other traffic has occurred during that interval, if that is decided for the actual connection. It is RECOMMENDED to use the keep-alive solution from [RFC6263]. The consent check of [RFC7675] is a possible alternative if it is used anyway for other reasons.

### 3.4. Transmission interval

A "text/red" or "text/t140" transmitter in a mixer SHALL send packets distributed in time as long as there is something (new or redundant T140blocks) to transmit. The maximum transmission interval between text transmissions from the same source SHALL then be 330 ms, when no other limitations cause a longer interval to be temporarily used. It is RECOMMENDED to send the next packet to a receiver as soon as new text to that receiver is available, as long as the mean character rate of new text to the receiver calculated over the last 10 one-second intervals does not exceed the "cps" value of the receiver. The intention is to keep the latency low and network load limited while keeping good protection against text loss in bursty packet loss conditions. The main purpose of the 330 ms interval is for timing of redundant transmission, when no new text from the same source is available.

The reason for the value 330 ms is that many sources of text will transmit new text with 300 ms intervals during periods of continuous user typing, and then reception in the mixer of such new text will cause a combined transmission of the new text and the unsent redundancy from the previous transmission. Only when the user stops typing, the 330 ms interval will be applied to send the redundancy.

If the Characters Per Second (cps) value is reached, a longer transmission interval SHALL be applied for text from all sources as specified in [RFC4103] and only as much of the text queued for transmission SHALL be sent at the end of each transmission interval as can be allowed without exceeding the "cps" value. Division of text for partial transmission MUST then be made at T140block borders. When the transmission rate falls under the "cps" value again, the transmission intervals SHALL be returned to 330 ms and transmission of new text SHALL return to be made as soon as new text is available.

NOTE: that extending the transmission intervals during high load periods does not change the number of characters to be conveyed. It just evens out the load in time and reduces the number of packets per second. With human created conversational text, the sending user will eventually take a pause letting transmission catch up.

See also Section 8.

For a transmitter not acting as a mixer, the transmission interval principles from [RFC4103] apply, and the normal transmission interval SHALL be 300 ms.

### 3.5. Only one source per packet

New text and redundant copies of earlier text from one source SHALL be transmitted in the same packet if available for transmission at the same time. Text from different sources MUST NOT be transmitted in the same packet.

### 3.6. Do not send received text to the originating source

Text received by a mixer from a participant SHOULD NOT be included in transmission from the mixer to that participant, because the normal behavior of the endpoint is to present locally-produced text locally.

### 3.7. Clean incoming text

A mixer SHALL handle reception, recovery from packet loss, deletion of superfluous redundancy, marking of possible text loss and deletion of 'BOM' characters from each participant before queueing received text for transmission to receiving participants as specified in [RFC4103] for single-party sources and Section 3.16 for multiparty sources (chained mixers).

### 3.8. Redundant transmission principles

A transmitting party using redundancy SHALL send redundant repetitions of T140blocks already transmitted in earlier packets.

The number of redundant generations of T140blocks to include in transmitted packets SHALL be deduced from the SDP negotiation. It SHALL be set to the minimum of the number declared by the two parties negotiating a connection. It is RECOMMENDED to declare and transmit one original and two redundant generations of the T140blocks because that provides good protection against text loss in case of packet loss, and low overhead.

### 3.9. Text placement in packets

The mixer SHALL compose and transmit an RTP packet to a receiver when one or more of the following conditions have occurred:

- \* The transmission interval is the normal 330 ms and there is newly received unsent text available for transmission to that receiver.
- \* The current transmission interval has passed and is longer than the normal 330 ms and there is newly received unsent text available for transmission to that receiver.
- \* The current transmission interval ( normally 330 ms) has passed since already transmitted text was queued for transmission as redundant text.

The principles from [RFC4103] apply for populating the header, the redundancy header and the data in the packet with specifics specified here and in the following sections.

At the time of transmission, the mixer SHALL populate the RTP packet with all T140blocks queued for transmission originating from the source in turn for transmission as long as this is not in conflict with the allowed number of characters per second ("cps") or the maximum packet size. In this way, the latency of the latest received text is kept low even in moments of simultaneous transmission from many sources.

Redundant text SHALL also be included, and the assessment of how much new text can be included within the maximum packet size MUST take into account that the redundancy has priority to be transmitted in its entirety. See Section 3.4

The SSRC of the source SHALL be placed as the only member in the CSRC-list.

Note: The CSRC-list in an RTP packet only includes the participant whose text is included in text blocks. It is not the same as the total list of participants in a conference. With audio and video media, the CSRC-list would often contain all participants who are not muted whereas text participants that don't type are completely silent and thus are not represented in RTP packet CSRC-lists.

### 3.10. Empty T140blocks

If no unsent T140blocks were available for a source at the time of populating a packet, but T140blocks are available which have not yet been sent the full intended number of redundant transmissions, then the primary T140block for that source is composed of an empty T140block, and populated (without taking up any length) in a packet for transmission. The corresponding SSRC SHALL be placed as usual in its place in the CSRC-list.

The first packet in the session, the first after a source switch, and the first after a pause SHALL be populated with the available T140blocks for the source in turn to be sent as primary, and empty T140blocks for the agreed number of redundancy generations.

### 3.11. Creation of the redundancy

The primary T140block from a source in the latest transmitted packet is saved for populating the first redundant T140block for that source in the next transmission of text from that source. The first redundant T140block for that source from the latest transmission is saved for populating the second redundant T140block in the next transmission of text from that source.

Usually this is the level of redundancy used. If a higher level of redundancy is negotiated, then the procedure SHALL be maintained until all available redundant levels of T140blocks are placed in the packet. If a receiver has negotiated a lower number of "text/red" generations, then that level SHALL be the maximum used by the transmitter.

The T140blocks saved for transmission as redundant data are assigned a planned transmission time 330 ms after the current time, but SHOULD be transmitted earlier if new text for the same source gets in turn for transmission before that time.

### 3.12. Timer offset fields

The timestamp offset values SHALL be inserted in the redundancy header, with the time offset from the RTP timestamp in the packet when the corresponding T140block was sent as primary.

The timestamp offsets are expressed in the same clock tick units as the RTP timestamp.

The timestamp offset values for empty T140blocks have no relevance but SHOULD be assigned realistic values.

### 3.13. Other RTP header fields

The number of members in the CSRC list (0 or 1) SHALL be placed in the "CC" header field. Only mixers place value 1 in the "CC" field. A value of "0" indicates that the source is the transmitting device itself and that the source is indicated by the SSRC field. This value is used by endpoints, and by mixers sending self-sourced data.

The current time SHALL be inserted in the timestamp.

The SSRC header field SHALL contain the SSRC of the RTP session where the packet will be transmitted.

The M-bit SHALL be handled as specified in [RFC4103].

### 3.14. Pause in transmission

When there is no new T140block to transmit, and no redundant T140block that has not been retransmitted the intended number of times from any source, the transmission process SHALL be stopped until either new T140blocks arrive, or a keep-alive method calls for transmission of keep-alive packets.

### 3.15. RTCP considerations

A mixer SHALL send RTCP reports with SDES, CNAME, and NAME information about the sources in the multiparty call. This makes it possible for participants to compose a suitable label for text from each source.

Privacy considerations SHALL be taken when composing these fields. They contain name and address information that may be sensitive to transmit in its entirety, e.g., to unauthenticated participants.

### 3.16. Reception of multiparty contents

The "text/red" receiver included in an endpoint with presentation functions will receive RTP packets in the single stream from the mixer, and SHALL distribute the T140blocks for presentation in presentation areas for each source. Other receiver roles, such as gateways or chained mixers, are also feasible. They require considerations if the stream shall just be forwarded, or distributed based on the different sources.

#### 3.16.1. Acting on the source of the packet contents

If the "CC" field value of a received packet is 1, it indicates that the text is conveyed from a source indicated in the single member in the CSRC-list, and the receiver MUST act on the source according to its role. If the CC value is 0, the source is indicated in the SSRC field.

#### 3.16.2. Detection and indication of possible text loss

The receiver SHALL monitor the RTP sequence numbers of the received packets for gaps and packets out of order. If a sequence number gap appears and still exists after some defined short time for jitter and reordering resolution, the packets in the gap SHALL be regarded as lost.

If it is known that only one source is active in the RTP session, then it is likely that a gap equal to or larger than the agreed number of redundancy generations (including the primary) causes text loss. In that case, the receiver SHALL create a t140block with a marker for possible text loss [T140ad1] and associate it with the source and insert it in the reception buffer for that source.

If it is known that more than one source is active in the RTP session, then it is not possible in general to evaluate if text was lost when packets were lost. With two active sources and the recommended number of redundancy generations (3), it can take a gap

of five consecutive lost packets until any text may be lost, but text loss can also appear if three non-consecutive packets are lost when they contained consecutive data from the same source. A simple method to decide when there is risk for resulting text loss is to evaluate if three or more packets were lost within one second. If this simple method is used, then a t140block SHOULD be created with a marker for possible text loss [T140ad1] and associated with the SSRC of the RTP session as a general input from the mixer.

Implementations MAY apply more refined methods for more reliable detection of whether text was lost or not. Any refined method SHOULD prefer marking possible loss rather than not marking when it is uncertain if there was loss.

### 3.16.3. Extracting text and handling recovery

When applying the following procedures, the effects MUST be considered of possible timestamp wrap around and the RTP session possibly changing SSRC.

When a packet is received in an RTP session using the packetization for multiparty-aware endpoints, its T140blocks SHALL be extracted in the following way.

The source SHALL be extracted from the CSRC-list if available, otherwise from the SSRC.

If the received packet is the first packet received from the source, then all T140blocks in the packet SHALL be retrieved and assigned to a receive buffer for the source beginning with the oldest available redundant generation, continuing with the younger redundant generations in age order and finally the primary.

Note: The normal case is that in the first packet, only the primary data has contents. The redundant data has contents in the first received packet from a source only after initial packet loss.

If the packet is not the first packet from a source, then if redundant data is available, the process SHALL start with the oldest generation. The timestamp of that redundant data SHALL be created by subtracting its timestamp offset from the RTP timestamp. If the resulting timestamp is later than the latest retrieved data from the same source, then the redundant data SHALL be retrieved and appended to the receive buffer. The process SHALL be continued in the same way for all younger generations of redundant data. After that, the timestamp of the packet SHALL be compared with the timestamp of the latest retrieved data from the same source and if it is later, then the primary data SHALL be retrieved from the packet and appended to the receive buffer for the source.

#### 3.16.4. Delete 'BOM'

Unicode character 'BOM' is used as a start indication and sometimes used as a filler or keep alive by transmission implementations. These SHALL be deleted after extraction from received packets.

#### 3.17. Performance considerations

This solution has good performance with low text delays, as long as the mean number of characters per second sent during any 10-second interval from a number of simultaneously sending participants to a receiving participant, does not reach the "cps" value. At higher numbers of sent characters per second, a jerkiness is visible in the presentation of text. The solution is therefore suitable for emergency service use, relay service use, and small or well-managed larger multimedia conferences. Only in large unmanaged conferences with a high number of participants there may on very rare occasions appear situations when many participants happen to send text simultaneously. In such circumstances, the result may be unpleasantly jerky presentation of text from each sending participant. It should be noted that it is only the number of users sending text within the same moment that causes jerkiness, not the total number of users with RTT capability.

#### 3.18. Security for session control and media

Security mechanisms to provide confidentiality and integrity protection and peer authentication SHOULD be applied when possible regarding the capabilities of the participating devices by use of SIP over TLS by default according to [RFC5630] section 3.1.3 on the session control level and by default using DTLS-SRTP [RFC5764] on the media level. In applications where legacy endpoints without security are allowed, a negotiation SHOULD be performed to decide if encryption on the media level will be applied. If no other security solution is mandated for the application, then OSRTP [RFC8643] is a

suitable method to be applied to negotiate SRTP media security with DTLS. Most SDP examples below are for simplicity expressed without the security additions. The principles (but not all details) for applying DTLS-SRTP [RFC5764] security are shown in a couple of the following examples.

Further general security considerations are covered in Section 10.

End-to-end encryption would require further work and could be based on WebRTC as specified in Section 1.2 or on double encryption as specified in [RFC8723].

### 3.19. SDP offer/answer examples

This section shows some examples of SDP for session negotiation of the real-time text media in SIP sessions. Audio is usually provided in the same session, and sometimes also video. The examples only show the part of importance for the real-time text media. The examples relate to the single RTP stream mixing for multiparty-aware endpoints and for multiparty-unaware endpoints.

Note: Multiparty RTT MAY also be provided through other methods, e.g., by a Selective Forwarding Middlebox (SFM). In that case, the SDP of the offer will include something specific for that method, e.g., an SDP attribute or another media format. An answer selecting the use of that method would accept it by a corresponding acknowledgement included in the SDP. The offer may contain also the "rtt-mixer" SDP media attribute for the main RTT media when the offeror has capability for both multiparty methods, while an answer, selecting to use SFM will not include the "rtt-mixer" SDP media attribute.

Offer example for "text/red" format and multiparty support:

```
m=text 11000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

Answer example from a multiparty-aware device

```
m=text 14000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

Offer example for "text/red" format including multiparty and security:

```
a=fingerprint: (fingerprint1)
m=text 11000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

The "fingerprint" is sufficient to offer DTLS-SRTP, with the media line still indicating RTP/AVP.

Note: For brevity, the entire value of the SDP fingerprint attribute is not shown in this and the following example.

Answer example from a multiparty-aware device with security

```
a=fingerprint: (fingerprint2)
m=text 16000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

With the "fingerprint" the device acknowledges use of SRTP/DTLS.

Answer example from a multiparty-unaware device that also does not support security:

```
m=text 12000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
```

### 3.20. Packet sequence example from interleaved transmission

This example shows a symbolic flow of packets from a mixer including loss and recovery. The sequence includes interleaved transmission of text from two RTT sources A and B. P indicates primary data. R1 is first redundant generation data and R2 is the second redundant generation data. A1, B1, A2 etc. are text chunks (T140blocks) received from the respective sources and sent on to the receiver by the mixer. X indicates a dropped packet between the mixer and a receiver. The session is assumed to use original and two redundant generations of RTT.

```

-----
Seq no 101, Time=20400
CC=1
CSRC list A
R2: A1, Offset=600
R1: A2, Offset=300
P: A3
-----

```

Assuming that earlier packets (with text A1 and A2) were received in sequence, text A3 is received from packet 101 and assigned to reception buffer A. The mixer is now assumed to have received initial text from source B 100 ms after packet 101 and will send that text. Transmission of A2 and A3 as redundancy is planned for 330 ms after packet 101 if no new text from A is ready to be sent before that.

```

-----
Seq no 102, Time=20500
CC=1
CSRC list B
R2 Empty, Offset=600
R1: Empty, Offset=300
P: B1
-----

```

Packet 102 is received.  
B1 is retrieved from this packet. Redundant transmission of B1 is planned 330 ms after packet 102.

```

X-----
X Seq no 103, Timer=20730
X CC=1
X CSRC list A
X R2: A2, Offset=630
X R1: A3, Offset=330
X P: Empty
X-----

```

Packet 103 is assumed to be lost due to network problems. It contains redundancy for A. Sending A3 as second level redundancy is planned for 330 ms after packet 103.

```
X-----|
X Seq no 104, Timer=20800|
X CC=1|
X CSRC list B|
X R2: Empty, Offset=600|
X R1: B1, Offset=300|
X P: B2|
X-----|
```

Packet 104 contains text from B, including new B2 and redundant B1. It is assumed dropped due to network problems.

The mixer has A3 redundancy to send, but no new text appears from A and therefore the redundancy is sent 330 ms after the previous packet with text from A.

```
-----|
Seq no 105, Timer=21060|
CC=1|
CSRC list A|
R2: A3, Offset=660|
R1: Empty, Offset=330|
P: Empty|
-----|
```

Packet 105 is received.

A gap for lost packets 103 and 104 is detected.

Assume that no other loss was detected during the last second.

Then it can be concluded that nothing was totally lost.

R2 is checked. Its original time was  $21060 - 660 = 20400$ .

A packet with text from A was received with that timestamp, so nothing needs to be recovered.

B1 and B2 still need to be transmitted as redundancy.

This is planned 330 ms after packet 104. That would be at 21130.

```
-----|
Seq no 106, Timer=21130|
CC=1|
CSRC list B|
R2: B1, Offset=630|
R1: B2, Offset=330|
P: Empty|
-----|
```

Packet 106 is received.

The second level redundancy in packet 106 is B1 and has timestamp offset 630 ms. The timestamp of packet 106 minus 630 is 20500 which is the timestamp of packet 102 that was received. So B1 does not need to be retrieved. The first level redundancy in packet 106 has offset 330. The timestamp of packet 106 minus 330 is 20800. That is later than the latest received packet with source B. Therefore B2 is retrieved and assigned to the input buffer for source B. No primary is available in packet 106.

After this sequence, A3 and B1 and B2 have been received. In this case no text was lost.

### 3.21. Maximum character rate "cps"

The default maximum rate of reception of "text/t140" real-time text is in [RFC4103] specified to be 30 characters per second. The actual rate is calculated without regard to any redundant text transmission and is in the multiparty case evaluated for all sources contributing to transmission to a receiver. The value MAY be modified in the "cps" parameter of the FMTP attribute in the media section for the "text/t140" media. A mixer combining real-time text from a number of sources may occasionally have a higher combined flow of text coming from the sources. Endpoints SHOULD therefore specify a suitable higher value for the "cps" parameter, corresponding to its real reception capability. A value for "cps" of 90 SHALL be the default for the "text/t140" stream in the "text/red" format when multiparty real-time text is negotiated. See [RFC4103] for the format and use of the "cps" parameter. The same rules apply for the multiparty case except for the default value.

## 4. Presentation level considerations

"Protocol for multimedia application text conversation" [T140] provides the presentation level requirements for the [RFC4103] transport. Functions for erasure and other formatting functions are specified in [T140] which has the following general statement for the presentation:

"The display of text from the members of the conversation should be arranged so that the text from each participant is clearly readable, and its source and the relative timing of entered text is visualized in the display. Mechanisms for looking back in the contents from the current session should be provided. The text should be displayed as soon as it is received."

Strict application of [T140] is of essence for the interoperability of real-time text implementations and to fulfill the intention that the session participants have the same information conveyed in the text contents of the conversation without necessarily having the exact same layout of the conversation.

[T140] specifies a set of presentation control codes to include in the stream. Some of them are optional. Implementations MUST ignore optional control codes that they do not support.

There is no strict "message" concept in real-time text. The Unicode Line Separator character SHALL be used as a separator allowing a part of received text to be grouped in presentation. The characters "CRLF" may be used by other implementations as a replacement for Line Separator. The "CRLF" combination SHALL be erased by just one erasing action, the same as the Line Separator. Presentation functions are allowed to group text for presentation in smaller groups than the line separators imply and present such groups with source indication together with text groups from other sources (see the following presentation examples). Erasure has no specific limit by any delimiter in the text stream.

#### 4.1. Presentation by multiparty-aware endpoints

A multiparty-aware receiving party, presenting real-time text MUST separate text from different sources and present them in separate presentation fields. The receiving party MAY separate presentation of parts of text from a source in readable groups based on other criteria than line separator and merge these groups in the presentation area when it benefits the user to most easily find and read text from the different participants. The criteria MAY e.g., be a received comma, full stop, or other phrase delimiters, or a long pause.

When text is received from multiple original sources, the presentation SHALL provide a view where text is added in multiple presentation fields.

If the presentation presents text from different sources in one common area, the presenting endpoint SHOULD insert text from the local user ended at suitable points merged with received text to indicate the relative timing for when the text groups were completed. In this presentation mode, the receiving endpoint SHALL present the source of the different groups of text. This presentation style is called the "chat" style here and provides a possibility to follow text arriving from multiple parties and the approximate relative time that text is received related to text from the local user.

A view of a three-party RTT call in chat style is shown in this example .

[Alice] Hi, Alice here.	^
[Bob] Bob as well.	-
[Eve] Hi, this is Eve, calling from Paris. I thought you should be here.	
[Alice] I am coming on Thursday, my performance is not until Friday morning.	
[Bob] And I on Wednesday evening.	
[Alice] Can we meet on Thursday evening?	
[Eve] Yes, definitely. How about 7pm. at the entrance of the restaurant Le Lion Blanc?	
[Eve] we can have dinner and then take a walk	-
<Eve-typing> But I need to be back to the hotel by 11 because I need	v ^
<Bob-typing> I wou	-
of course, I underst	v

Figure 3: Example of a three-party RTT call presented in chat style seen at participant 'Alice's endpoint.

Other presentation styles than the chat style MAY be arranged.

This figure shows how a coordinated column view MAY be presented.

Bob	Eve	Alice
My flight is to Orly		I will arrive by TGV. Convenient to the main station.
Eve, will you do your presentation on Friday?	Hi all, can we plan for the seminar?	
Fine, wo	Yes, Friday at 10.	We need to meet befo

Figure 4: An example of a coordinated column-view of a three-party session with entries ordered vertically in approximate time-order.

#### 4.2. Multiparty mixing for multiparty-unaware endpoints

When the mixer has indicated RTT multiparty capability in an SDP negotiation, but the multiparty capability negotiation fails with an endpoint, then the agreed "text/red" or "text/t140" format SHALL be used and the mixer SHOULD compose a best-effort presentation of multiparty real-time text in one stream intended to be presented by an endpoint with no multiparty awareness, when that is desired in the actual implementation. The following specifies a procedure which MAY be applied in that situation.

This presentation format has functional limitations and SHOULD be used only to enable participation in multiparty calls by legacy deployed endpoints implementing only RFC 4103 without any multiparty extensions specified in this document.

The principles and procedures below do not specify any new protocol elements. They are instead composed of information from [T140] and an ambition to provide a best-effort presentation on an endpoint which has functions originally intended only for two-party calls.

The mixer mixing for multiparty-unaware endpoints SHALL compose a simulated, limited multiparty RTT view suitable for presentation in one presentation area. The mixer SHALL group text in suitable groups and prepare for presentation of them by inserting a line separator between them if the transmitted text did not already end with a new line (line separator or CRLF). A presentable label SHALL be composed and sent for the source initially in the session and after each source switch. With this procedure the time for switching from transmission of text from one source to transmission of text from another source depends on the actions of the users. In order to expedite source switching, a user can, for example, end its turn with a new line.

#### 4.2.1. Actions by the mixer at reception from the call participants

When text is received by the mixer from the different participants, the mixer SHALL recover text from redundancy if any packets are lost. The mark for lost text [T140ad1] SHALL be inserted in the stream if unrecoverable loss appears. Any Unicode "BOM" characters, possibly used for keep-alive, SHALL be deleted. The time of creation of text (retrieved from the RTP timestamp) SHALL be stored together with the received text from each source in queues for transmission to the recipients in order to be able to evaluate text loss.

#### 4.2.2. Actions by the mixer for transmission to the recipients

The following procedure SHALL be applied for each multiparty-unaware recipient of multiparty text from the mixer.

The text for transmission SHALL be formatted by the mixer for each receiving user for presentation in one single presentation area. Text received from a participant SHOULD NOT be included in transmission to that participant because it is usually presented locally at transmission time. When there is text available for transmission from the mixer to a receiving party from more than one participant, the mixer SHALL switch between transmission of text from the different sources at suitable points in the transmitted stream.

When switching source, the mixer SHALL insert a line separator if the already transmitted text did not end with a new line (line separator or CRLF). A label SHALL be composed of information in the CNAME and NAME fields in RTCP reports from the participant to have its text transmitted, or from other session information for that user. The label SHALL be delimited by suitable characters (e.g., '[ ]') and transmitted. The CSRC SHALL indicate the selected source. Then text from that selected participant SHALL be transmitted until a new suitable point for switching source is reached.

Information available to the mixer for composing the label may contain sensitive personal information that SHOULD NOT be revealed in sessions not securely authenticated and confidentiality protected. Privacy considerations regarding how much personal information is included in the label SHOULD therefore be taken when composing the label.

Seeking a suitable point for switching source SHALL be done when there is older text waiting for transmission from any party than the age of the last transmitted text. Suitable points for switching are:

- \* A completed phrase ended by comma
- \* A completed sentence
- \* A new line (line separator or CRLF)
- \* A long pause (e.g., > 10 seconds) in received text from the currently transmitted source
- \* If text from one participant has been transmitted with text from other sources waiting for transmission for a long time (e.g., > 1 minute) and none of the other suitable points for switching has occurred, a source switch MAY be forced by the mixer at the next word delimiter, and also even if a word delimiter does not occur within a time (e.g., 15 seconds) after the scan for a word delimiter started.

When switching source, the source which has the oldest text in queue SHALL be selected to be transmitted. A character display count SHALL be maintained for the currently transmitted source, starting at zero after the label is transmitted for the currently transmitted source.

The status SHALL be maintained for the latest control code for Select Graphic Rendition (SGR) from each source. If there is an SGR code stored as the status for the current source before the source switch is done, a reset of SGR SHALL be sent by the sequence SGR 0 [009B 0000 006D] after the new line and before the new label during a source switch. See SGR below for an explanation. This transmission does not influence the display count.

If there is an SGR code stored for the new source after the source switch, that SGR code SHALL be transmitted to the recipient before the label. This transmission does not influence the display count.

#### 4.2.3. Actions on transmission of text

Text from a source sent to the recipient SHALL increase the display count by one per transmitted character.

#### 4.2.4. Actions on transmission of control codes

The following control codes specified by T.140 require specific actions. They SHALL cause specific considerations in the mixer. Note that the codes presented here are expressed in UCS-16, while transmission is made in the UTF-8 encoding of these codes.

BEL 0007 Bell Alert in session. Provides for alerting during an active session. The display count SHALL NOT be altered.

NEW LINE 2028 Line separator. Check and perform a source switch if appropriate. Increase the display count by 1.

CR LF 000D 000A A supported but not preferred way of requesting a new line. Check and perform a source switch if appropriate. Increase the display count by 1.

INT ESC 0061 Interrupt (used to initiate the mode negotiation procedure). The display count SHALL NOT be altered.

SGR 009B Ps 006D Select graphic rendition. Ps is the rendition parameters specified in ISO 6429. The display count SHALL NOT be altered. The SGR code SHOULD be stored for the current source.

SOS 0098 Start of string, used as a general protocol element introducer, followed by a maximum 256-byte string and the ST. The display count SHALL NOT be altered.

ST 009C String terminator, end of SOS string. The display count SHALL NOT be altered.

ESC 001B Escape - used in control strings. The display count SHALL NOT be altered for the complete escape code.

Byte order mark "BOM" (U+FEFF) "Zero width, no break space", used for synchronization and keep-alive. It SHALL be deleted from incoming streams. It SHALL also be sent first after session establishment to the recipient. The display count SHALL NOT be altered.

Missing text mark (U+FFFD) "Replacement character", represented as a

question mark in a rhombus, or if that is not feasible, replaced by an apostrophe '. It marks the place in the stream of possible text loss. This mark SHALL be inserted by the reception procedure in case of unrecoverable loss of packets. The display count SHALL be increased by one when sent as for any other character.

SGR If a control code for selecting graphic rendition (SGR) other than reset of the graphic rendition (SGR 0) is sent to a recipient, that control code SHALL also be stored as the status for the source in the storage for SGR status. If a reset graphic rendition (SGR 0) originating from a source is sent, then the SGR status storage for that source SHALL be cleared. The display count SHALL NOT be increased.

BS (U+0008) Back Space, intended to erase the last entered character by a source. Erasure by backspace cannot always be performed as the erasing party intended. If an erasing action erases all text up to the end of the leading label after a source switch, then the mixer MUST NOT transmit more backspaces. Instead, it is RECOMMENDED that a letter "X" is inserted in the text stream for each backspace as an indication of the intent to erase more. A new line is usually coded by a Line Separator, but the character combination "CRLF" MAY be used instead. Erasure of a new line is in both cases done by just one erasing action (Backspace). If the display count has a positive value it SHALL be decreased by one when the BS is sent. If the display count is at zero, it SHALL NOT be altered.

#### 4.2.5. Packet transmission

A mixer transmitting to a multiparty-unaware terminal SHALL send primary data only from one source per packet. The SSRC SHALL be the SSRC of the mixer. The CSRC list SHALL contain one member and be the SSRC of the source of the primary data.

#### 4.2.6. Functional limitations

When a multiparty-unaware endpoint presents a conversation in one display area in a chat style, it inserts source indications for remote text and local user text as they are merged in completed text groups. When an endpoint using this layout receives and presents text mixed for multiparty-unaware endpoints, there will be two levels of source indicators for the received text; one generated by the mixer and inserted in a label after each source switch, and another generated by the receiving endpoint and inserted after each switch between local and remote source in the presentation area. This will waste display space and look inconsistent to the reader.

New text can be presented only from one source at a time. Switch of source to be presented takes place at suitable places in the text, such as end of phrase, end of sentence, line separator and inactivity. Therefore, the time to switch to present waiting text from other sources may become long and will vary and depend on the actions of the currently presented source.

Erasure can only be done up to the latest source switch. If a user tries to erase more text, the erasing actions will be presented as letter X after the label.

Text loss because of network errors may hit the label between entries from different parties, causing risk for misunderstanding from which source a piece of text is.

These facts make it strongly RECOMMENDED implementing multiparty awareness in RTT endpoints. The use of the mixing method for multiparty-unaware endpoints should be left for use with endpoints which are impossible to upgrade to become multiparty-aware.

#### 4.2.7. Example views of presentation on multiparty-unaware endpoints

The following pictures are examples of the view on a participant's display for the multiparty-unaware case.

Conference	Alice
[Bob]:My flight is to Orly. [Eve]:Hi all, can we plan for the seminar.	I will arrive by TGV. Convenient to the main station.
[Bob]:Eve, will you do your presentation on Friday? [Eve]:Yes, Friday at 10.	
[Bob]: Fine, wo	We need to meet befo

Figure 5: Alice who has a conference-unaware client is receiving the multiparty real-time text in a single-stream.

This figure shows how a coordinated column view MAY be presented on Alice's device in a view with two-columns. The mixer inserts labels to show how the sources alternate in the column with received text.

The mixer alternates between the sources at suitable points in the text exchange so that text entries from each party can be conveniently read.

(Alice) Hi, Alice here.	^
(mix) [Bob] Bob as well.	-
[Eve] Hi, this is Eve, calling from Paris I thought you should be here.	
(Alice) I am coming on Thursday, my performance is not until Friday morning.	
(mix) [Bob] And I on Wednesday evening.	
[Eve] we can have dinner and then walk	
[Eve] But I need to be back to the hotel by 11 because I need	
of course, I underst	- v

Figure 6: An example of a view of the multiparty-unaware presentation in chat style. Alice is the local user.

In this view, there is a tradition in receiving applications to include a label showing the source of the text, here shown with parenthesis "()". The mixer also inserts source labels for the multiparty call participants, here shown with brackets "[ ]".

## 5. Relation to Conference Control

### 5.1. Use with SIP centralized conferencing framework

The Session Initiation Protocol (SIP) conferencing framework, mainly specified in [RFC4353], [RFC4579] and [RFC4575] is suitable for coordinating sessions including multiparty RTT. The RTT stream between the mixer and a participant is one and the same during the conference. Participants get announced by notifications when participants are joining or leaving, and further user information may be provided. The SSRC of the text to expect from joined users MAY be included in a notification. The notifications MAY be used both for security purposes and for translation to a label for presentation to other users.

### 5.2. Conference control

In managed conferences, control of the real-time text media SHOULD be provided in the same way as other for media, e.g., for muting and unmuting by the direction attributes in SDP [RFC8866].

Note that floor control functions may be of value for RTT users as well as for users of other media in a conference.

## 6. Gateway Considerations

### 6.1. Gateway considerations with Textphones

multiparty RTT sessions may involve gateways of different kinds. Gateways involved in setting up sessions SHALL correctly reflect the multiparty capability or unawareness of the combination of the gateway and the remote endpoint beyond the gateway.

One case that may occur is a gateway to Public Switched Telephone Network (PSTN) for communication with textphones (e.g., TTYs). Textphones are limited devices with no multiparty awareness, and it SHOULD therefore be suitable for the gateway to not indicate multiparty awareness for that case. Another solution is that the gateway indicates multiparty capability towards the mixer, and includes the multiparty mixer function for multiparty-unaware endpoints itself. This solution makes it possible to adapt to the functional limitations of the textphone.

More information on gateways to textphones is found in [RFC5194]

### 6.2. Gateway considerations with WebRTC

Gateway operation to real-time text in WebRTC may also be required. In WebRTC, RTT is specified in [RFC8865].

A multiparty bridge may have functionality for communicating by RTT both in RTP streams with RTT and WebRTC T.140 data channels. Other configurations may consist of a multiparty bridge with either technology for RTT transport and a separate gateway for conversion of the text communication streams between RTP and T.140 data channel.

In WebRTC, it is assumed that for a multiparty session, one T.140 data channel is established for each source from a gateway or bridge to each participant. Each participant also has a data channel with a two-way connection with the gateway or bridge.

The T.140 data channel used both ways is for text from the WebRTC user and from the bridge or gateway itself to the WebRTC user. The label parameter of this T.140 data channel is used as the NAME field in RTCP to participants on the RTP side. The other T.140 data channels are only for text from other participants to the WebRTC user.

When a new participant has entered the session with RTP transport of RTT, a new T.140 channel SHOULD be established to WebRTC users with the label parameter composed of information from the NAME field in RTCP on the RTP side.

When a new participant has entered the multiparty session with RTT transport in a WebRTC T.140 data channel, the new participant SHOULD be announced by a notification to RTP users. The label parameter from the WebRTC side SHOULD be used as the NAME RTCP field on the RTP side, or other available session information.

When a participant on the RTP side is disconnected from the multiparty session, the corresponding T.140 data channel(s) SHOULD be closed.

When a WebRTC user of T.140 data channels disconnects from the mixer, the corresponding RTP streams or sources in an RTP-mixed stream SHOULD be closed.

T.140 data channels MAY be opened and closed by negotiation or renegotiation of the session or by any other valid means as specified in section 1 of [RFC8865].

## 7. Updates to RFC 4103

This document updates [RFC4103] by introducing an SDP media attribute "rtt-mixer" for negotiation of multiparty-mixing capability with the [RFC4103] format, and by specifying the rules for packets when multiparty capability is negotiated and in use.

## 8. Congestion considerations

The congestion considerations and recommended actions from [RFC4103] are also valid in multiparty situations.

The time values SHALL then be applied per source of text sent to a receiver.

If the very unlikely situation appears that many participants in a conference send text simultaneously for a long period, a delay may build up for presentation of text at the receivers if the limitation in characters per second ("cps") to be transmitted to the participants is exceeded. More delay than 7 seconds can cause confusion in the session. It is therefore RECOMMENDED that an RTP-mixer-based mixer discards such text causing excessive delays and inserts a general indication of possible text loss [T140ad1] in the session. If the main text contributor is indicated in any way, the mixer MAY avoid deleting text from that participant. It should however be noted that human creation of text normally contains pauses, when the transmission can catch up, so that the transmission overload situations are expected to be very rare.

## 9. IANA Considerations

### 9.1. Registration of the "rtt-mixer" SDP media attribute

[RFC EDITOR NOTE: Please replace all instances of RFCXXXX with the RFC number of this document.]

IANA is asked to register the new SDP attribute "rtt-mixer".

Contact name: IESG

Contact email: iesg@ietf.org

Attribute name: rtt-mixer

Attribute semantics: See RFCXXXX Section 2.3

Attribute value: none

Usage level: media

Purpose: Indicate support by mixer and endpoint of multiparty mixing for real-time text transmission, using a common RTP-stream for transmission of text from a number of sources mixed with one source at a time and the source indicated in a single CSRC-list member.

Charset Dependent: no

O/A procedure: See RFCXXXX Section 2.3

Mux Category: normal

Reference: RFCXXXX

## 10. Security Considerations

The RTP-mixer model requires the mixer to be allowed to decrypt, pack, and encrypt secured text from the conference participants. Therefore, the mixer needs to be trusted to maintain confidentiality and integrity of the RTT data. This situation is similar to the situation for handling audio and video media in centralized mixers.

The requirement to transfer information about the user in RTCP reports in SDES, CNAME, and NAME fields, and in conference notifications, may have privacy concerns as already stated in RFC 3550 [RFC3550], and may be restricted for privacy reasons. When used for creation of readable labels in the presentation, the receiving user will then get a more symbolic label for the source.

The services available through the RTT mixer may have special interest for deaf and hard-of-hearing persons. Some users may want to refrain from revealing such characteristics broadly in conferences. The design of the conference systems where the mixer is included MAY need to be made with confidentiality of such characteristics in mind.

Participants with malicious intentions may appear and e.g., disturb the multiparty session by emitting a continuous flow of text. They may also send text that appears to originate from other participants. Counteractions should be to require secure signaling, media and authentication, and to provide higher-layer conference functions e.g., for blocking, muting, and expelling participants.

Participants with malicious intentions may also try to disturb the presentation by sending incomplete or malformed control codes. Handling of text from the different sources by the receivers MUST therefore be well separated so that the effects of such actions only affect text from the source causing the action.

Care should be taken that if use of the mixer is allowed for users both with and without security procedures, opens for possible attacks by both unauthenticated call participants and even eavesdropping and manipulating of content non-participants.

As already stated in Section 3.18, security in media SHOULD be applied by using DTLS-SRTP [RFC5764] on the media level.

Further security considerations specific for this application are specified in Section 3.18.

## 11. Change history

[RFC Editor: Please remove this section prior to publication.]

### 11.1. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-20

Inclusion of edits as response to a comment by Benjamin Kaduk in section 3.16.3 to make the recovery procedure generic.

Added persons to the acknowledgements and moved acknowledgements to last in the document.

### 11.2. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-19

Edits because of comments in a review by Francesca Palombini.

Edits because of comments from Benjamin Kaduk.

Proposed to not change anything because of Robert Wilton's comments.

Two added sentences in the security section to meet comments by Roman Danyliw.

### 11.3. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-18

Edits of nits as proposed in a review by Lars Eggert.

Edits as response to review by Martin Duke.

### 11.4. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-17

Actions on Gen-ART review comments.

Actions on SecDir review comments.

### 11.5. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-16

Improvements in the offer/answer considerations section by adding subsections for each phase in the negotiation as requested by IANA expert review.

## 11.6. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-15

Actions on review comments from Jurgen Schonwalder:

A bit more about congestion situations and that they are expected to be very rare.

Explanation of differences in security between the conference-aware and the conference-unaware case added in security section.

Presentation examples with source labels made less confusing, and explained.

Reference to T.140 inserted at first mentioning of T.140.

Reference to RFC 8825 inserted to explain WebRTC

Nit in wording in terminology section adjusted.

## 11.7. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-14

Changes from comments by Murray Kucherawy during AD review.

Many SHOULD in section 4.2 on multiparty-unaware mixing changed to SHALL, and the whole section instead specified to be optional depending on the application.

Some SHOULD in section 3 either explained or changed to SHALL.

In order to have explainable conditions behind SHOULDs, the transmission interval in 3.4 is changed to as soon as text is available as a main principle. The call participants send with 300 ms interval so that will create realistic load conditions anyway.

## 11.8. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-13

Changed year to 2021.

Changed reference to draft on RTT in WebRTC to recently published RFC 8865.

Changed label brackets in example from "[]" to "()" to avoid nits comment.

Changed reference "RFC 4566" to recently published "RFC 8866"

#### 11.9. Changes included in draft-ietf-avtc core-multi-party-rtt-mix-12

Changes according to responses on comments from Brian Rosen in Avtc core list on 2020-12-05 and -06.

Changes according to responses to comments by Bernard Aboba in avtc core list 2020-12-06.

Introduction of an optiona RTP multi-stream mixing method for further study as proposed by Bernard Aboba.

Changes clarifying how to open and close T.140 data channels included in 6.2 after comments by Lorenzo Miniero.

Changes to satisfy nits check. Some "not" changed to "NOT" in normative wording combinations. Some lower case normative words changed to upper case. A normative reference deleted from the abstract. Two informative documents moved from normative references to informative references.

#### 11.10. Changes included in draft-ietf-avtc core-multi-party-rtt-mix-11

Timestamps and timestamp offsets added to the packet examples in section 3.23, and the description corrected.

A number of minor corrections added in sections 3.10 - 3.23.

#### 11.11. Changes included in draft-ietf-avtc core-multi-party-rtt-mix-10

The packet composition was modified for interleaving packets from different sources.

The packet reception was modified for the new interleaving method.

The packet sequence examples was adjusted for the new interleaving method.

Modifications according to responses to Brian Rosen of 2020-11-03

#### 11.12. Changes included in draft-ietf-avtc core-multi-party-rtt-mix-09

Changed name on the SDP media attribute to "rtt-mixer"

Restructure of section 2 for balance between aware and unaware cases.

Moved conference control to own section.

Improved clarification of recovery and loss in the packet sequence example.

A number of editorial corrections and improvements.

11.13. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-08

Deleted the method requiring a new packet format "text/rex" because of the longer standardization and implementation period it needs.

Focus on use of RFC 4103 text/red format with shorter transmission interval, and source indicated in CSRC.

11.14. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-07

Added a method based on the "text/red" format and single source per packet, negotiated by the "rtt-mixer" SDP attribute.

Added reasoning and recommendation about indication of loss.

The highest number of sources in one packet is 15, not 16. Changed.

Added in information on update to RFC 4103 that RFC 4103 explicitly allows addition of FEC method. The redundancy is a kind of forward error correction.

11.15. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-06

Improved definitions list format.

The format of the media subtype parameters is made to match the requirements.

The mapping of media subtype parameters to SDP is included.

The "cps" parameter belongs to the t140 subtype and does not need to be registered here.

11.16. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-05

nomenclature and editorial improvements

"this document" used consistently to refer to this document.

11.17. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-04

'Redundancy header' renamed to 'data header'.

More clarifications added.

Language and figure number corrections.

#### 11.18. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-03

Mention possible need to mute and raise hands as for other media.  
---done ----

Make sure that use in two-party calls is also possible and explained.  
- may need more wording -

Clarify the RTT is often used together with other media. --done--

Tell that text mixing is N-1. A users own text is not received in the mix. -done-

In 3. correct the interval to: A "text/rex" transmitter SHOULD send packets distributed in time as long as there is something (new or redundant T140blocks) to transmit. The maximum transmission interval SHOULD then be 300 ms. It is RECOMMENDED to send a packet to a receiver as soon as new text to that receiver is available, as long as the time after the latest sent packet to the same receiver is more than 150 ms, and also the maximum character rate to the receiver is not exceeded. The intention is to keep the latency low while keeping a good protection against text loss in bursty packet loss conditions.  
-done-

In 1.3 say that the format is used both ways. -done-

In 13.1 change presentation area to presentation field so that reader does not think it shall be totally separated. -done-

In Performance and intro, tell the performance in number of simultaneous sending users and introduced delay 16, 150 vs requirements 5 vs 500. -done --

Clarify redundancy level per connection. -done-

Timestamp also for the last data header. To make it possible for all text to have time offset as for transmission from the source. Make that header equal to the others. -done-

Mixer always use the CSRC list, even for its own BOM. -done-

Combine all talk about transmission interval (300 ms vs when text has arrived) in section 3 in one paragraph or close to each other. -done-

Documents the goal of good performance with low delay for 5 simultaneous typers in the introduction. -done-

Describe better that only primary text shall be sent on to receivers. Redundancy and loss must be resolved by the mixer. -done-

11.19. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-02

SDP and better description and visibility of security by OSRTP RFC 8634 needed.

The description of gatewaying to WebRTC extended.

The description of the data header in the packet is improved.

11.20. Changes to draft-ietf-avtcore-multi-party-rtt-mix-01

2,5,6 More efficient format "text/rex" introduced and attribute a=rtt-mix deleted.

3. Brief about use of OSRTP for security included- More needed.

4. Brief motivation for the solution and why not rtp-translator is used added to intro.

7. More limitations for the multiparty-unaware mixing method inserted.

8. Updates to RFC 4102 and 4103 more clearly expressed.

9. Gateway to WebRTC started. More needed.

11.21. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-03 to draft-ietf-avtcore-multi-party-rtt-mix-00

Changed file name to draft-ietf-avtcore-multi-party-rtt-mix-00

Replaced CDATA in IANA registration table with better coding.

Converted to xml2rfc version 3.

11.22. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-02 to -03

Changed company and e-mail of the author.

Changed title to "RTP-mixer formatting of multi-party Real-time text" to better match contents.

Check and modification where needed of use of RFC 2119 words SHALL etc.

More about the CC value in sections on transmitters and receivers so that 1-to-1 sessions do not use the mixer format.

Enhanced section on presentation for multiparty-unaware endpoints

A paragraph recommending cps=150 inserted in the performance section.

11.23. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-01 to -02

In Abstract and 1. Introduction: Introduced wording about regulatory requirements.

In section 5: The transmission interval is decreased to 100 ms when there is text from more than one source to transmit.

In section 11 about SDP negotiation, a SHOULD-requirement is introduced that the mixer should make a mix for multiparty-unaware endpoints if the negotiation is not successful. And a reference to a later chapter about it.

The presentation considerations chapter 14 is extended with more information about presentation on multiparty-aware endpoints, and a new section on the multiparty-unaware mixing with low functionality but SHOULD be implemented in mixers. Presentation examples are added.

A short chapter 15 on gateway considerations is introduced.

Clarification about the text/t140 format included in chapter 10.

This sentence added to the chapter 10 about use without redundancy. "The text/red format SHOULD be used unless some other protection against packet loss is utilized, for example a reliable network or transport."

Note about deviation from RFC 2198 added in chapter 4.

In chapter 9. "Use with SIP centralized conferencing framework" the following note is inserted: Note: The CSRC-list in an RTP packet only includes participants whose text is included in one or more text blocks. It is not the same as the list of participants in a conference. With audio and video media, the CSRC-list would often contain all participants who are not muted whereas text participants that don't type are completely silent and so don't show up in RTP packet CSRC-lists.

#### 11.24. Changes from draft-hellstrom-avtcore-multi-party-rtt-source-00 to -01

Editorial cleanup.

Changed capability indication from fntp-parameter to SDP attribute "rtt-mix".

Swapped order of redundancy elements in the example to match reality.

Increased the SDP negotiation section

## 12. References

### 12.1. 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>>.
- [RFC4102] Jones, P., "Registration of the text/red MIME Sub-Type", RFC 4102, DOI 10.17487/RFC4102, June 2005, <<https://www.rfc-editor.org/info/rfc4102>>.
- [RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <<https://www.rfc-editor.org/info/rfc4103>>.
- [RFC5630] Audet, F., "The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)", RFC 5630, DOI 10.17487/RFC5630, October 2009, <<https://www.rfc-editor.org/info/rfc5630>>.

- [RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <<https://www.rfc-editor.org/info/rfc5764>>.
- [RFC6263] Marjou, X. and A. Sollaud, "Application Mechanism for Keeping Alive the NAT Mappings Associated with RTP / RTP Control Protocol (RTCP) Flows", RFC 6263, DOI 10.17487/RFC6263, June 2011, <<https://www.rfc-editor.org/info/rfc6263>>.
- [RFC7675] Perumal, M., Wing, D., Ravindranath, R., Reddy, T., and M. Thomson, "Session Traversal Utilities for NAT (STUN) Usage for Consent Freshness", RFC 7675, DOI 10.17487/RFC7675, October 2015, <<https://www.rfc-editor.org/info/rfc7675>>.
- [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>>.
- [RFC8865] Holmberg, C. and G. Hellström, "T.140 Real-Time Text Conversation over WebRTC Data Channels", RFC 8865, DOI 10.17487/RFC8865, January 2021, <<https://www.rfc-editor.org/info/rfc8865>>.
- [RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <<https://www.rfc-editor.org/info/rfc8866>>.
- [T140] ITU-T, "Recommendation ITU-T T.140 (02/1998), Protocol for multimedia application text conversation", February 1998, <<https://www.itu.int/rec/T-REC-T.140-199802-I/en>>.
- [T140ad1] ITU-T, "Recommendation ITU-T.140 Addendum 1 - (02/2000), Protocol for multimedia application text conversation", February 2000, <<https://www.itu.int/rec/T-REC-T.140-200002-I!Add1/en>>.

## 12.2. Informative References

- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, DOI 10.17487/RFC4353, February 2006, <<https://www.rfc-editor.org/info/rfc4353>>.

- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, Ed., "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, DOI 10.17487/RFC4575, August 2006, <<https://www.rfc-editor.org/info/rfc4575>>.
- [RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, DOI 10.17487/RFC4579, August 2006, <<https://www.rfc-editor.org/info/rfc4579>>.
- [RFC5194] van Wijk, A., Ed. and G. Gybels, Ed., "Framework for Real-Time Text over IP Using the Session Initiation Protocol (SIP)", RFC 5194, DOI 10.17487/RFC5194, June 2008, <<https://www.rfc-editor.org/info/rfc5194>>.
- [RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <<https://www.rfc-editor.org/info/rfc7667>>.
- [RFC8643] Johnston, A., Aboba, B., Hutton, A., Jesske, R., and T. Stach, "An Opportunistic Approach for Secure Real-time Transport Protocol (OSRTP)", RFC 8643, DOI 10.17487/RFC8643, August 2019, <<https://www.rfc-editor.org/info/rfc8643>>.
- [RFC8723] Jennings, C., Jones, P., Barnes, R., and A.B. Roach, "Double Encryption Procedures for the Secure Real-Time Transport Protocol (SRTP)", RFC 8723, DOI 10.17487/RFC8723, April 2020, <<https://www.rfc-editor.org/info/rfc8723>>.
- [RFC8825] Alvestrand, H., "Overview: Real-Time Protocols for Browser-Based Applications", RFC 8825, DOI 10.17487/RFC8825, January 2021, <<https://www.rfc-editor.org/info/rfc8825>>.

#### Acknowledgements

The author want to thank the following persons for support, reviews and valuable comments: Bernard Aboba, Amanda Baber, Roman Danyliw, Spencer Dawkins, Martin Duke, Lars Eggert, James Hamlin, Benjamin Kaduk, Murray Kucherawy, Paul Kyziwat, Jonathan Lennox, Lorenzo Miniero, Dan Mongrain, Francesca Palombini, Colin Perkins, Brian Rosen, Juergen Schoenwaelder, Rich Salz, Robert Wilton, Dale Worley, Peter Yee and Yong Xin.

#### Author's Address

Gunnar Hellstrom  
Gunnar Hellstrom Accessible Communication  
SE-13670 Vendelso  
Sweden

Email: [gunnar.hellstrom@ghaccess.se](mailto:gunnar.hellstrom@ghaccess.se)

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: December 18, 2020

M. Westerlund  
B. Burman  
Ericsson  
C. Perkins  
University of Glasgow  
H. Alvestrand  
Google  
R. Even  
June 16, 2020

Guidelines for using the Multiplexing Features of RTP to Support  
Multiple Media Streams  
draft-ietf-avtc core-multiplex-guidelines-12

Abstract

The Real-time Transport Protocol (RTP) is a flexible protocol that can be used in a wide range of applications, networks, and system topologies. That flexibility makes for wide applicability, but can complicate the application design process. One particular design question that has received much attention is how to support multiple media streams in RTP. This memo discusses the available options and design trade-offs, and provides guidelines on how to use the multiplexing features of RTP to support multiple media streams.

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 <https://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 December 18, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1.	Introduction . . . . .	3
2.	Definitions . . . . .	4
2.1.	Terminology . . . . .	4
2.2.	Subjects Out of Scope . . . . .	5
3.	RTP Multiplexing Overview . . . . .	5
3.1.	Reasons for Multiplexing and Grouping RTP Streams . . . . .	5
3.2.	RTP Multiplexing Points . . . . .	6
3.2.1.	RTP Session . . . . .	7
3.2.2.	Synchronisation Source (SSRC) . . . . .	8
3.2.3.	Contributing Source (CSRC) . . . . .	10
3.2.4.	RTP Payload Type . . . . .	11
3.3.	Issues Related to RTP Topologies . . . . .	12
3.4.	Issues Related to RTP and RTCP Protocol . . . . .	13
3.4.1.	The RTP Specification . . . . .	13
3.4.2.	Multiple SSRCs in a Session . . . . .	15
3.4.3.	Binding Related Sources . . . . .	15
3.4.4.	Forward Error Correction . . . . .	17
4.	Considerations for RTP Multiplexing . . . . .	17
4.1.	Interworking Considerations . . . . .	17
4.1.1.	Application Interworking . . . . .	17
4.1.2.	RTP Translator Interworking . . . . .	18
4.1.3.	Gateway Interworking . . . . .	19
4.1.4.	Multiple SSRC Legacy Considerations . . . . .	20
4.2.	Network Considerations . . . . .	20
4.2.1.	Quality of Service . . . . .	20
4.2.2.	NAT and Firewall Traversal . . . . .	21
4.2.3.	Multicast . . . . .	23
4.3.	Security and Key Management Considerations . . . . .	24
4.3.1.	Security Context Scope . . . . .	24
4.3.2.	Key Management for Multi-party Sessions . . . . .	25
4.3.3.	Complexity Implications . . . . .	26
5.	RTP Multiplexing Design Choices . . . . .	26
5.1.	Multiple Media Types in One Session . . . . .	26
5.2.	Multiple SSRCs of the Same Media Type . . . . .	28
5.3.	Multiple Sessions for One Media Type . . . . .	29
5.4.	Single SSRC per Endpoint . . . . .	30

5.5. Summary . . . . .	32
6. Guidelines . . . . .	32
7. IANA Considerations . . . . .	33
8. Security Considerations . . . . .	33
9. Contributors . . . . .	34
10. Acknowledgments . . . . .	34
11. References . . . . .	34
11.1. Normative References . . . . .	34
11.2. Informative References . . . . .	35
Appendix A. Dismissing Payload Type Multiplexing . . . . .	39
Appendix B. Signalling Considerations . . . . .	40
B.1. Session Oriented Properties . . . . .	41
B.2. SDP Prevents Multiple Media Types . . . . .	42
B.3. Signalling RTP Stream Usage . . . . .	42
Authors' Addresses . . . . .	43

## 1. Introduction

The Real-time Transport Protocol (RTP) [RFC3550] is a commonly used protocol for real-time media transport. It is a protocol that provides great flexibility and can support a large set of different applications. RTP was from the beginning designed for multiple participants in a communication session. It supports many topology paradigms and usages, as defined in [RFC7667]. RTP has several multiplexing points designed for different purposes. These enable support of multiple RTP streams and switching between different encoding or packetization of the media. By using multiple RTP sessions, sets of RTP streams can be structured for efficient processing or identification. Thus, an RTP application designer needs to understand how to best use the RTP session, the RTP stream identifier (SSRC), and the RTP payload type to meet the application's needs.

There have been increased interest in more advanced usage of RTP. For example, multiple RTP streams can be used when a single endpoint has multiple media sources (like multiple cameras or microphones) that need to be sent simultaneously. Consequently, questions are raised regarding the most appropriate RTP usage. The limitations in some implementations, RTP/RTCP extensions, and signalling have also been exposed. This document aims to clarify the usefulness of some functionalities in RTP which will hopefully result in more complete implementations in the future.

The purpose of this document is to provide clear information about the possibilities of RTP when it comes to multiplexing. The RTP application designer needs to understand the implications arising from a particular usage of the RTP multiplexing points. The document

will provide some guidelines and recommend against some usages as being unsuitable, in general or for particular purposes.

The document starts with some definitions and then goes into the existing RTP functionalities around multiplexing. Both the desired behaviour and the implications of a particular behaviour depend on which topologies are used, which requires some consideration. This is followed by a discussion of some choices in multiplexing behaviour and their impacts. Some designs of RTP usage are discussed. Finally, some guidelines and examples are provided.

## 2. Definitions

### 2.1. Terminology

The definitions in Section 3 of [RFC3550] are referenced normatively.

The taxonomy defined in [RFC7656] is referenced normatively.

The following terms and abbreviations are used in this document:

**Multiparty:** A communication situation including multiple endpoints. In this document, it will be used to refer to situations where more than two endpoints communicate.

**Multiplexing:** The operation of taking multiple entities as input, aggregating them onto some common resource while keeping the individual entities addressable such that they can later be fully and unambiguously separated (de-multiplexed) again.

**RTP Receiver:** An Endpoint or Middlebox receiving RTP streams and RTCP messages. It uses at least one SSRC to send RTCP messages. An RTP Receiver may also be an RTP Sender.

**RTP Sender:** An Endpoint sending one or more RTP streams, but also sending RTCP messages.

**RTP Session Group:** One or more RTP sessions that are used together to perform some function. Examples are multiple RTP sessions used to carry different layers of a layered encoding. In an RTP Session Group, CNAMEs are assumed to be valid across all RTP sessions, and designate synchronisation contexts that can cross RTP sessions; i.e. SSRCs that map to a common CNAME can be assumed to have RTCP Sender Report (SR) timing information derived from a common clock such that they can be synchronised for playback.

**Signalling:** The process of configuring endpoints to participate in one or more RTP sessions.

Note: The above definitions of RTP Receiver and RTP Sender are consistent with the usage in [RFC3550].

## 2.2. Subjects Out of Scope

This document is focused on issues that affect RTP. Thus, issues that involve signalling protocols, such as whether SIP [RFC3261], Jingle [JINGLE] or some other protocol is in use for session configuration, the particular syntaxes used to define RTP session properties, or the constraints imposed by particular choices in the signalling protocols, are mentioned only as examples in order to describe the RTP issues more precisely.

This document assumes the applications will use RTCP. While there are applications that don't send RTCP, they do not conform to the RTP specification, and thus can be regarded as reusing the RTP packet format but not implementing the RTP protocol.

## 3. RTP Multiplexing Overview

### 3.1. Reasons for Multiplexing and Grouping RTP Streams

There are several reasons why an endpoint might choose to send multiple media streams. In the below discussion, please keep in mind that the reasons for having multiple RTP streams vary and include but are not limited to the following:

- o Multiple media sources
- o Multiple RTP streams might be needed to represent one media source for instance:
  - \* To carry different layers of an scalable encoding of a media source
  - \* Alternative encodings during simulcast, for instance using different codecs for the same audio stream
  - \* Alternative formats during simulcast, for instance multiple resolutions of the same video stream
- o A retransmission stream might repeat some parts of the content of another RTP stream
- o A Forward Error Correction (FEC) stream might provide material that can be used to repair another RTP stream

For each of these reasons, it is necessary to decide if each additional RTP stream is sent within the same RTP session as the other RTP streams, or if it is necessary to use additional RTP sessions to group the RTP streams. The choice suitable for one situation, might not be the choice suitable in another situation or combination of reasons. The clearest understanding is associated with multiplexing multiple media sources of the same media type. However, all reasons warrant discussion and clarification on how to deal with them. As the discussion below will show, in reality we cannot choose a single one of SSRC or RTP session multiplexing solutions for all purposes. To utilise RTP well and as efficiently as possible, both are needed. The real issue is finding the right guidance on when to create additional RTP sessions and when additional RTP streams in the same RTP session is the right choice.

### 3.2. RTP Multiplexing Points

This section describes the multiplexing points present in the RTP protocol that can be used to distinguish RTP streams and groups of RTP streams. Figure 1 outlines the process of demultiplexing incoming RTP streams starting already at the socket representing reception of one or more transport flows, e.g. based on the UDP destination port. It also demultiplexes RTP/RTCP from any other protocols, such as STUN [RFC5389] and DTLS-SRTP [RFC5764] on the same transport as described in [RFC7983]. The Processing and Buffering (PB) step of Figure 1 terminates the RTP/RTCP protocol and prepares the RTP payload for input to the decoder.

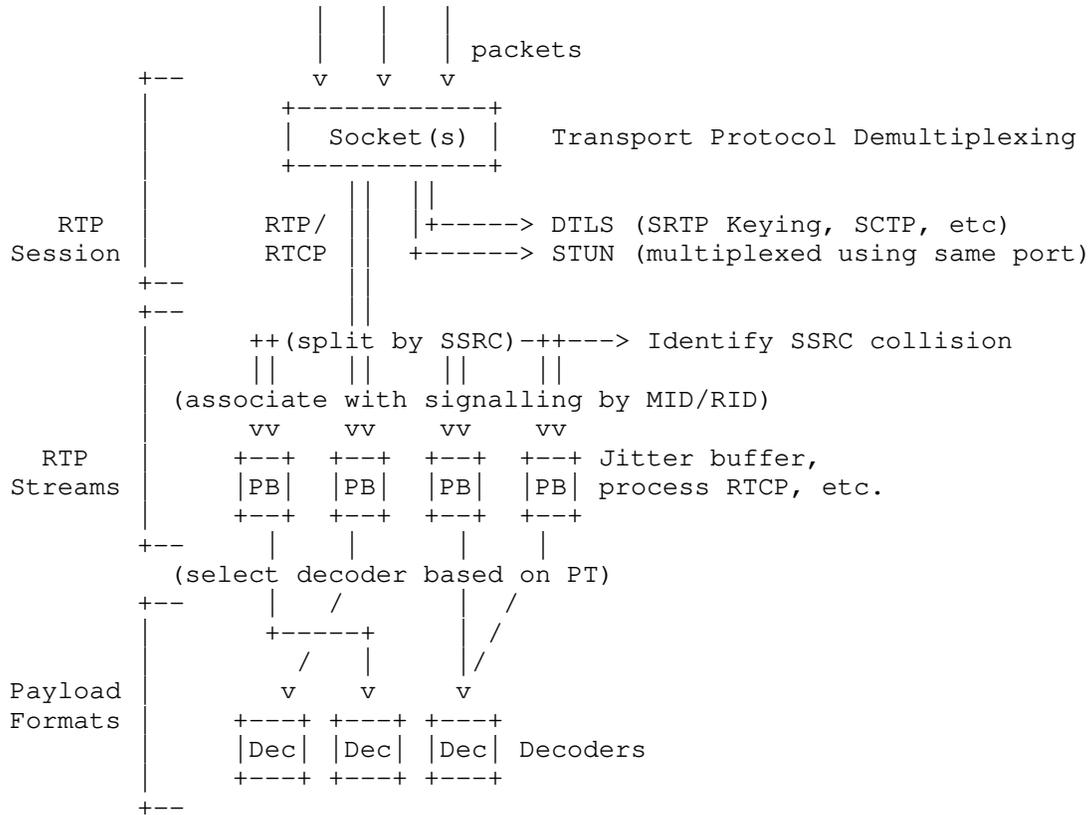


Figure 1: RTP Demultiplexing Process

### 3.2.1. RTP Session

An RTP session is the highest semantic layer in the RTP protocol, and represents an association between a group of communicating endpoints. RTP does not contain a session identifier, yet different RTP sessions must be possible to identify both across a set of different endpoints and from the perspective of a single endpoint.

For RTP session separation across endpoints, the set of participants that form an RTP session is defined as those that share a single synchronisation source space [RFC3550]. That is, if a group of participants are each aware of the synchronisation source identifiers belonging to the other participants, then those participants are in a single RTP session. A participant can become aware of a synchronisation source identifier by receiving an RTP packet containing it in the SSRC field or CSRC list, by receiving an RTCP

packet mentioning it in an SSRC field, or through signalling (e.g., the Session Description Protocol (SDP) [RFC4566] "a=ssrc:" attribute [RFC5576]). Thus, the scope of an RTP session is determined by the participants' network interconnection topology, in combination with RTP and RTCP forwarding strategies deployed by the endpoints and any middleboxes, and by the signalling.

For RTP session separation within a single endpoint RTP relies on the underlying transport layer, and on the signalling to identify RTP sessions in a manner that is meaningful to the application. A single endpoint can have one or more transport flows for the same RTP session, and a single RTP session can span multiple transport layer flows even if all endpoints use a single transport layer flow per endpoint for that RTP session. The signalling layer might give RTP sessions an explicit identifier, or the identification might be implicit based on the addresses and ports used. Accordingly, a single RTP session can have multiple associated identifiers, explicit and implicit, belonging to different contexts. For example, when running RTP on top of UDP/IP, an endpoint can identify and delimit an RTP session from other RTP sessions by their UDP source and destination IP addresses and UDP port numbers. A single RTP session can be using multiple IP/UDP flows for receiving and/or sending RTP packets to other endpoints or middleboxes, even if the endpoint does not have multiple IP addresses. Using multiple IP addresses only makes it more likely to require multiple IP/UDP flows. Another example is SDP media descriptions (the "m=" line and the following associated lines) that signal the transport flow and RTP session configuration for the endpoint's part of the RTP session. The SDP grouping framework [RFC5888] allows labeling of the media descriptions to be used so that RTP Session Groups can be created. Through use of Negotiating Media Multiplexing Using the Session Description Protocol (SDP) [I-D.ietf-mmusic-sdp-bundle-negotiation], multiple media descriptions become part of a common RTP session where each media description represents the RTP streams sent or received for a media source.

The RTP protocol makes no normative statements about the relationship between different RTP sessions, however the applications that use more than one RTP session will have some higher layer understanding of the relationship between the sessions they create.

### 3.2.2. Synchronisation Source (SSRC)

A synchronisation source (SSRC) identifies a source of an RTP stream, or an RTP receiver when sending RTCP. Every endpoint has at least one SSRC identifier, even if it does not send RTP packets. RTP endpoints that are only RTP receivers still send RTCP and use their SSRC identifiers in the RTCP packets they send. An endpoint can have

multiple SSRC identifiers if it sends multiple RTP streams. Endpoints that are both RTP sender and RTP receiver use the same SSRC(s) in both roles.

The SSRC is a 32-bit identifier. It is present in every RTP and RTCP packet header, and in the payload of some RTCP packet types. It can also be present in SDP signalling. Unless pre-signalled, e.g. using the SDP "a=ssrc:" attribute [RFC5576], the SSRC is chosen at random. It is not dependent on the network address of the endpoint, and is intended to be unique within an RTP session. SSRC collisions can occur, and are handled as specified in [RFC3550] and [RFC5576], resulting in the SSRC of the colliding RTP streams or receivers changing. An endpoint that changes its network transport address during a session has to choose a new SSRC identifier to avoid being interpreted as looped source, unless a mechanism providing a virtual transport (such as ICE [RFC8445]) abstracts the changes.

SSRC identifiers that belong to the same synchronisation context (i.e., that represent RTP streams that can be synchronised using information in RTCP SR packets) use identical CNAME chunks in corresponding RTCP SDES packets. SDP signalling can also be used to provide explicit SSRC grouping [RFC5576].

In some cases, the same SSRC identifier value is used to relate streams in two different RTP sessions, such as in RTP retransmission [RFC4588]. This is to be avoided since there is no guarantee that SSRC values are unique across RTP sessions. For the RTP retransmission [RFC4588] case it is recommended to use explicit binding of the source RTP stream and the redundancy stream, e.g. using the RepairedRtpStreamId RTCP SDES item [I-D.ietf-avtext-rid]. The RepairedRtpStreamId is a rather recent mechanism, so one cannot expect older applications to follow this recommendation.

Note that RTP sequence number and RTP timestamp are scoped by the SSRC and thus specific per RTP stream.

Different types of entities use an SSRC to identify themselves, as follows:

A real media source: Uses the SSRC to identify a "physical" media source.

A conceptual media source: Uses the SSRC to identify the result of applying some filtering function in a network node, for example a filtering function in an RTP mixer that provides the most active speaker based on some criteria, or a mix representing a set of other sources.

An RTP receiver: Uses the SSRC to identify itself as the source of its RTCP reports.

An endpoint that generates more than one media type, e.g. a conference participant sending both audio and video, need not (and, indeed, should not) use the same SSRC value across RTP sessions. RTCP compound packets containing the CNAME SDES item is the designated method to bind an SSRC to a CNAME, effectively cross-correlating SSRCs within and between RTP Sessions as coming from the same endpoint. The main property attributed to SSRCs associated with the same CNAME is that they are from a particular synchronisation context and can be synchronised at playback.

An RTP receiver receiving a previously unseen SSRC value will interpret it as a new source. It might in fact be a previously existing source that had to change SSRC number due to an SSRC conflict. Use of the MID extension [I-D.ietf-mmusic-sdp-bundle-negotiation] helps to identify which media source the new SSRC represents and use of the RID extension [I-D.ietf-mmusic-rid] helps to identify what encoding or redundancy stream it represents, even though the SSRC changed. However, the originator of the previous SSRC ought to have ended the conflicting source by sending an RTCP BYE for it prior to starting to send with the new SSRC, making the new SSRC a new source.

### 3.2.3. Contributing Source (CSRC)

The Contributing Source (CSRC) is not a separate identifier. Rather an SSRC identifier is listed as a CSRC in the RTP header of a packet generated by an RTP mixer or video MCU/switch, if the corresponding SSRC was in the header of one of the packets that contributed to the output.

It is not possible, in general, to extract media represented by an individual CSRC since it is typically the result of a media merge (e.g. mix) operation on the individual media streams corresponding to the CSRC identifiers. The exception is the case when only a single CSRC is indicated as this represent forwarding of an RTP stream, possibly modified. The RTP header extension for Mixer-to-Client Audio Level Indication [RFC6465] expands on the receiver's information about a packet with a CSRC list. Due to these restrictions, CSRC will not be considered a fully qualified multiplexing point and will be disregarded in the rest of this document.

### 3.2.4. RTP Payload Type

Each RTP stream utilises one or more RTP payload formats. An RTP payload format describes how the output of a particular media codec is framed and encoded into RTP packets. The payload format is identified by the payload type (PT) field in the RTP packet header. The combination of SSRC and PT therefore identifies a specific RTP stream in a specific encoding format. The format definition can be taken from [RFC3551] for statically allocated payload types, but ought to be explicitly defined in signalling, such as SDP, both for static and dynamic payload types. The term "format" here includes those aspects described by out-of-band signalling means; in SDP, the term "format" includes media type, RTP timestamp sampling rate, codec, codec configuration, payload format configurations, and various robustness mechanisms such as redundant encodings [RFC2198].

The RTP payload type is scoped by the sending endpoint within an RTP session. PT has the same meaning across all RTP streams in an RTP session. All SSRCs sent from a single endpoint share the same payload type definitions. The RTP payload type is designed such that only a single payload type is valid at any time instant in the RTP stream's timestamp time line, effectively time-multiplexing different payload types if any change occurs. The payload type can change on a per-packet basis for an SSRC, for example a speech codec making use of generic comfort noise [RFC3389]. If there is a true need to send multiple payload types for the same SSRC that are valid for the same instant, then redundant encodings [RFC2198] can be used. Several additional constraints than the ones mentioned above need to be met to enable this use, one of which is that the combined payload sizes of the different payload types ought not exceed the transport MTU.

Other aspects of RTP payload format use are described in How to Write an RTP Payload Format [RFC8088].

The payload type is not a multiplexing point at the RTP layer (see Appendix A for a detailed discussion of why using the payload type as an RTP multiplexing point does not work). The RTP payload type is, however, used to determine how to consume and decode an RTP stream. The RTP payload type number is sometimes used to associate an RTP stream with the signalling, which in general requires that unique RTP payload type numbers are used in each context. Use of MID, e.g. when bundling "m=" sections [I-D.ietf-mmusic-sdp-bundle-negotiation], can replace the payload type as signalling association and unique RTP payload types are then no longer required for that purpose.

### 3.3. Issues Related to RTP Topologies

The impact of how RTP multiplexing is performed will in general vary with how the RTP session participants are interconnected, described by RTP Topology [RFC7667].

Even the most basic use case, denoted Topo-Point-to-Point in [RFC7667], raises a number of considerations that are discussed in detail in following sections. They range over such aspects as:

- o Does my communication peer support RTP as defined with multiple SSRCs per RTP session?
- o Do I need network differentiation in form of QoS (Section 4.2.1)?
- o Can the application more easily process and handle the media streams if they are in different RTP sessions?
- o Do I need to use additional RTP streams for RTP retransmission or FEC?

For some point to multi-point topologies (e.g. Topo-ASM and Topo-SSM in [RFC7667]), multicast is used to interconnect the session participants. Special considerations (documented in Section 4.2.3) are then needed as multicast is a one-to-many distribution system.

Sometimes an RTP communication can end up in a situation when the communicating peers are not compatible for various reasons:

- o No common media codec for a media type thus requiring transcoding.
- o Different support for multiple RTP streams and RTP sessions.
- o Usage of different media transport protocols, i.e., RTP or other.
- o Usage of different transport protocols, e.g., UDP, DCCP, or TCP.
- o Different security solutions, e.g., IPsec, TLS, DTLS, or SRTP with different keying mechanisms.

In many situations this is resolved by the inclusion of a translator between the two peers, as described by Topo-PtP-Translator in [RFC7667]. The translator's main purpose is to make the peers look compatible to each other. There can also be other reasons than compatibility to insert a translator in the form of a middlebox or gateway, for example a need to monitor the RTP streams. Beware that changing the stream transport characteristics in the translator can

require thorough understanding of aspects from congestion control and media adaptation to application-layer semantics.

Within the uses enabled by the RTP standard, the point to point topology can contain one or more RTP sessions with one or more media sources per session, each having one or more RTP streams per media source.

### 3.4. Issues Related to RTP and RTCP Protocol

Using multiple RTP streams is a well-supported feature of RTP. However, for most implementers or people writing RTP/RTCP applications or extensions attempting to apply multiple streams, it can be unclear when it is most appropriate to add an additional RTP stream in an existing RTP session and when it is better to use multiple RTP sessions. This section discusses the various considerations needed.

#### 3.4.1. The RTP Specification

RFC 3550 contains some recommendations and a bullet list with 5 arguments for different aspects of RTP multiplexing. Please review Section 5.2 of [RFC3550]. Five important aspects are quoted below.

1. If, say, two audio streams shared the same RTP session and the same SSRC value, and one were to change encodings and thus acquire a different RTP payload type, there would be no general way of identifying which stream had changed encodings.

The first argument is to use different SSRC for each individual RTP stream, which is fundamental to RTP operation.

2. An SSRC is defined to identify a single timing and sequence number space. Interleaving multiple payload types would require different timing spaces if the media clock rates differ and would require different sequence number spaces to tell which payload type suffered packet loss.

The second argument is advocating against demultiplexing RTP streams within a session based only on their RTP payload type numbers, which still stands as can be seen by the extensive list of issues found in Appendix A.

3. The RTCP sender and receiver reports (see Section 6.4) can only describe one timing and sequence number space per SSRC and do not carry a payload type field.

The third argument is yet another argument against payload type multiplexing.

4. An RTP mixer would not be able to combine interleaved streams of incompatible media into one stream.

The fourth argument is against multiplexing RTP packets that require different handling into the same session. In most cases the RTP mixer must embed application logic to handle streams; the separation of streams according to stream type is just another piece of application logic, which might or might not be appropriate for a particular application. One type of application that can mix different media sources blindly is the audio-only telephone bridge, although the ability to do that comes from the well-defined scenario that is aided by use of a single media type, even though individual streams may use incompatible codec types; most other types of applications need application-specific logic to perform the mix correctly.

5. Carrying multiple media in one RTP session precludes: the use of different network paths or network resource allocations if appropriate; reception of a subset of the media if desired, for example just audio if video would exceed the available bandwidth; and receiver implementations that use separate processes for the different media, whereas using separate RTP sessions permits either single- or multiple-process implementations.

The fifth argument discusses network aspects that are described in Section 4.2. It also goes into aspects of implementation, like Split Component Terminal (see Section 3.10 of [RFC7667]) endpoints where different processes or inter-connected devices handle different aspects of the whole multi-media session.

A summary of RFC 3550's view on multiplexing is to use unique SSRCs for anything that is its own media/packet stream, and to use different RTP sessions for media streams that don't share a media type. This document supports the first point; it is very valid. The latter needs further discussion, as imposing a single solution on all usages of RTP is inappropriate. "Multiple Media Types in an RTP Session specification" [I-D.ietf-avtcore-multi-media-rtp-session] updates RFC 3550 to allow multiple media types in a RTP session. It also provides a detailed analysis of the potential benefits and issues in having multiple media types in the same RTP session. Thus, that document provides a wider scope for an RTP session and considers multiple media types in one RTP session as a possible choice for the RTP application designer.

### 3.4.2. Multiple SSRCs in a Session

Using multiple SSRCs at one endpoint in an RTP session requires resolving some unclear aspects of the RTP specification. These could potentially lead to some interoperability issues as well as some potential significant inefficiencies, as further discussed in "RTP Considerations for Endpoints Sending Multiple Media Streams" [RFC8108]. An RTP application designer should consider these issues and the possible application impact from lack of appropriate RTP handling or optimization in the peer endpoints.

Using multiple RTP sessions can potentially mitigate application issues caused by multiple SSRCs in an RTP session.

### 3.4.3. Binding Related Sources

A common problem in a number of various RTP extensions has been how to bind related RTP streams together. This issue is common to both using additional SSRCs and multiple RTP sessions.

The solutions can be divided into a few groups:

- o RTP/RTCP based
- o Signalling based, e.g. SDP
- o Grouping related RTP sessions
- o Grouping SSRCs within an RTP session

Most solutions are explicit, but some implicit methods have also been applied to the problem.

The SDP-based signalling solutions are:

**SDP Media Description Grouping:** The SDP Grouping Framework [RFC5888] uses various semantics to group any number of media descriptions. This has primarily been grouping RTP sessions, but in combination with [I-D.ietf-mmusic-sdp-bundle-negotiation] it can also group multiple media descriptions within a single RTP session.

**SDP Media Multiplexing:** Negotiating Media Multiplexing Using the Session Description Protocol (SDP) [I-D.ietf-mmusic-sdp-bundle-negotiation] uses both SDP and RTCP information to associate RTP streams to SDP media descriptions. This allows both to group RTP streams belonging to an SDP media description, and to group multiple SDP media descriptions into a single RTP session.

SDP SSRC grouping: Source-Specific Media Attributes in SDP [RFC5576] includes a solution for grouping SSRCs the same way as the Grouping framework groups Media Descriptions.

The above grouping constructs support many use cases. Those solutions have shortcomings in cases where the session's dynamic properties are such that it is difficult or a drain on resources to keep the list of related SSRCs up to date.

An RTP/RTCP-based grouping solution is to use the RTCP SDES CNAME to bind related RTP streams to an endpoint or to a synchronization context. For applications with a single RTP stream per type (media, source or redundancy stream), CNAME is sufficient for that purpose independently of whether one or more RTP sessions are used. However, some applications choose not to use CNAME because of perceived complexity or a desire not to implement RTCP and instead use the same SSRC value to bind related RTP streams across multiple RTP sessions. RTP Retransmission [RFC4588] in multiple RTP session mode and Generic FEC [RFC5109] both use the CNAME method to relate the RTP streams, which may work but might have some downsides in RTP sessions with many participating SSRCs. It is not recommended to use identical SSRC values across RTP sessions to relate RTP streams; When an SSRC collision occurs, this will force change of that SSRC in all RTP sessions and thus resynchronize all of them instead of only the single media stream having the collision.

Another method to implicitly bind SSRCs is used by RTP Retransmission [RFC4588] when using the same RTP session as the source RTP stream for retransmissions. The receiver missing a packet issues an RTP retransmission request, and then awaits a new SSRC carrying the RTP retransmission payload and where that SSRC is from the same CNAME. This limits a requester to having only one outstanding retransmission request on any new source SSRCs per endpoint.

RTP Payload Format Restrictions [I-D.ietf-mmusic-rid] provides an RTP/RTCP based mechanism to unambiguously identify the RTP streams within an RTP session and restrict the streams' payload format parameters in a codec-agnostic way beyond what is provided with the regular payload types. The mapping is done by specifying an "a=rid" value in the SDP offer/answer signalling and having the corresponding RtpStreamId value as an SDES item and an RTP header extension. The RID solution also includes a solution for binding redundancy RTP streams to their original source RTP streams, given that those use RID identifiers.

Experience has found that an explicit binding between the RTP streams, agnostic of SSRC values, behaves well. That way, solutions

using multiple RTP streams in a single RTP session and in multiple RTP sessions will use the same type of binding.

#### 3.4.4. Forward Error Correction

There exist a number of Forward Error Correction (FEC) based schemes for how to mitigate packet loss in the original streams. Most of the FEC schemes protect a single source flow. The protection is achieved by transmitting a certain amount of redundant information that is encoded such that it can repair one or more packet losses over the set of packets the redundant information protects. This sequence of redundant information needs to be transmitted as its own media stream, or in some cases, instead of the original media stream. Thus, many of these schemes create a need for binding related flows as discussed above. Looking at the history of these schemes, there are schemes using multiple SSRCs and schemes using multiple RTP sessions, and some schemes that support both modes of operation.

Using multiple RTP sessions supports the case where some set of receivers might not be able to utilise the FEC information. By placing it in a separate RTP session and if separating RTP sessions on transport level, FEC can easily be ignored already on the transport level, without considering any RTP layer information.

In usages involving multicast, having the FEC information on its own multicast group allows for similar flexibility. This is especially useful when receivers see heterogeneous packet loss rates. A receiver can decide, based on measurement of experienced packet loss rates, whether to join a multicast group with the suitable FEC data repair capabilities.

### 4. Considerations for RTP Multiplexing

#### 4.1. Interworking Considerations

There are several different kinds of interworking, and this section discusses two; interworking directly between different applications, and interworking of applications through an RTP Translator. The discussion includes the implications of potentially different RTP multiplexing point choices and limitations that have to be considered when working with some legacy applications.

##### 4.1.1. Application Interworking

It is not uncommon that applications or services of similar but not identical usage, especially the ones intended for interactive communication, encounter a situation where one want to interconnect two or more of these applications.

In these cases, one ends up in a situation where one might use a gateway to interconnect applications. This gateway must then either change the multiplexing structure or adhere to the respective limitations in each application.

There are two fundamental approaches to building a gateway: using RTP Translator interworking (RTP bridging), where the gateway acts as an RTP Translator with the two interconnected applications being members of the same RTP session; or using Gateway Interworking with RTP termination, where there are independent RTP sessions between each interconnected application and the gateway.

For interworking to be feasible, any security solution in use needs to be compatible and capable of exchanging keys with either the peer or the gateway under the used trust model. Secondly, the applications need to use media streams in a way that makes sense in both applications.

#### 4.1.2. RTP Translator Interworking

From an RTP perspective, the RTP Translator approach could work if all the applications are using the same codecs with the same payload types, have made the same multiplexing choices, and have the same capabilities in number of simultaneous RTP streams combined with the same set of RTP/RTCP extensions being supported. Unfortunately, this might not always be true.

When a gateway is implemented via an RTP Translator, an important consideration is if the two applications being interconnected need to use the same approach to multiplexing. If one side is using RTP session multiplexing and the other is using SSRC multiplexing with BUNDLE [I-D.ietf-mmusic-sdp-bundle-negotiation], it may be possible for the RTP translator to map the RTP streams between both sides using some method, e.g. based on the number and order of SDP "m=" lines from each side. There are also challenges with SSRC collision handling since, unless SSRC translation is applied on the RTP translator, there may be a collision on the SSRC multiplexing side that the RTP session multiplexing side will not be aware of. Furthermore, if one of the applications is capable of working in several modes (such as being able to use additional RTP streams in one RTP session or multiple RTP sessions at will), and the other one is not, successful interconnection depends on locking the more flexible application into the operating mode where interconnection can be successful, even if none of the participants are using the less flexible application when the RTP sessions are being created.

#### 4.1.3. Gateway Interworking

When one terminates RTP sessions at the gateway, there are certain tasks that the gateway has to carry out:

- o Generating appropriate RTCP reports for all RTP streams (possibly based on incoming RTCP reports), originating from SSRCs controlled by the gateway.
- o Handling SSRC collision resolution in each application's RTP sessions.
- o Signalling, choosing, and policing appropriate bit-rates for each session.

For applications that use any security mechanism, e.g., in the form of SRTP, the gateway needs to be able to decrypt and verify source integrity of the incoming packets, and re-encrypt, integrity protect, and sign the packets as the peer in the other application's security context. This is necessary even if all that's needed is a simple remapping of SSRC numbers. If this is done, the gateway also needs to be a member of the security contexts of both sides, and thus a trusted entity.

The gateway might also need to apply transcoding (for incompatible codec types), media-level adaptations that cannot be solved through media negotiation (such as rescaling for incompatible video size requirements), suppression of content that is known not to be handled in the destination application, or the addition or removal of redundancy coding or scalability layers to fit the needs of the destination domain.

From the above, we can see that the gateway needs to have an intimate knowledge of the application requirements; a gateway is by its nature application specific, not a commodity product.

These gateways might therefore potentially block application evolution by blocking RTP and RTCP extensions that the applications have been extended with but that are unknown to the gateway.

If one uses security mechanism, like SRTP, the gateway and the necessary trust in it by the peers is an additional risk to the communication security. The gateway also incur additional complexities in form of the decrypt-encrypt cycles needed for each forwarded packet. SRTP, due to its keying structure, also requires that each RTP session needs different master keys, as use of the same key in two RTP sessions can for some ciphers result in a reuse of a

one-time pad that completely breaks the confidentiality of the packets.

#### 4.1.4. Multiple SSRC Legacy Considerations

Historically, the most common RTP use cases have been point-to-point Voice over IP (VoIP) or streaming applications, commonly with no more than one media source per endpoint and media type (typically audio or video). Even in conferencing applications, especially voice-only, the conference focus or bridge has provided a single stream to each participant containing a mix of the other participants. It is also common to have individual RTP sessions between each endpoint and the RTP mixer, meaning that the mixer functions as an RTP-terminating gateway.

Applications and systems that aren't updated to handle multiple streams following these recommendations can have issues with participating in RTP sessions containing multiple SSRCs within a single session, such as:

1. Need to handle more than one stream simultaneously rather than replacing an already existing stream with a new one.
2. Be capable of decoding multiple streams simultaneously.
3. Be capable of rendering multiple streams simultaneously.

This indicates that gateways attempting to interconnect to this class of devices have to make sure that only one RTP stream of each media type gets delivered to the endpoint if it's expecting only one, and that the multiplexing format is what the device expects. It is highly unlikely that RTP translator-based interworking can be made to function successfully in such a context.

#### 4.2. Network Considerations

The RTP implementer needs to consider that the RTP multiplexing choice also impacts network level mechanisms.

##### 4.2.1. Quality of Service

Quality of Service mechanisms are either flow based or packet marking based. RSVP [RFC2205] is an example of a flow based mechanism, while Diff-Serv [RFC2474] is an example of a packet marking based one.

For a flow based scheme, additional SSRC will receive the same QoS as all other RTP streams being part of the same 5-tuple (protocol,

source address, destination address, source port, destination port), which is the most common selector for flow based QoS.

For a packet marking based scheme, the method of multiplexing will not affect the possibility to use QoS. Different Differentiated Services Code Points (DSCP) can be assigned to different packets within a transport flow (5-Tuple) as well as within an RTP stream, assuming usage of UDP or other transport protocol that do not have issues with packet reordering within the transport flow (5-tuple). To avoid packet reordering issues, packets belonging to the same RTP flow should limit its use of DSCP to those whose corresponding Per Hop Behavior (PHB) that do not enable reordering. If the transport protocol used assumes in order delivery of packet, such as TCP and SCTP, then a single DSCP should be used. For more discussion of this see [RFC7657].

The method for assigning marking to packets can impact what number of RTP sessions to choose. If this marking is done using a network ingress function, it can have issues discriminating the different RTP streams. The network API on the endpoint also needs to be capable of setting the marking on a per-packet basis to reach the full functionality.

#### 4.2.2. NAT and Firewall Traversal

In today's networks there exist a large number of middleboxes. The ones that normally have most impact on RTP are Network Address Translators (NAT) and Firewalls (FW).

Below we analyse and comment on the impact of requiring more underlying transport flows in the presence of NATs and Firewalls:

**End-Point Port Consumption:** A given IP address only has 65536 available local ports per transport protocol for all consumers of ports that exist on the machine. This is normally never an issue for an end-user machine. It can become an issue for servers that handle large number of simultaneous streams. However, if the application uses ICE to authenticate STUN requests, a server can serve multiple endpoints from the same local port, and use the whole 5-tuple (source and destination address, source and destination port, protocol) as identifier of flows after having securely bound them to the remote endpoint address using the STUN request. In theory, the minimum number of media server ports needed are the maximum number of simultaneous RTP sessions a single endpoint can use. In practice, implementation will probably benefit from using more server ports to simplify implementation or avoid performance bottlenecks.

**NAT State:** If an endpoint sits behind a NAT, each flow it generates to an external address will result in a state that has to be kept in the NAT. That state is a limited resource. In home or Small Office/Home Office (SOHO) NATs, memory or processing are usually the most limited resources. For large scale NATs serving many internal endpoints, available external ports are likely the scarce resource. Port limitations is primarily a problem for larger centralised NATs where endpoint independent mapping requires each flow to use one port for the external IP address. This affects the maximum number of internal users per external IP address. However, as a comparison, a real-time video conference session with audio and video likely uses less than 10 UDP flows, compared to certain web applications that can use 100+ TCP flows to various servers from a single browser instance.

**NAT Traversal Extra Delay:** Performing the NAT/FW traversal takes a certain amount of time for each flow. It also takes time in a phase of communication between accepting to communicate and the media path being established, which is fairly critical. The best case scenario for additional NAT/FW traversal time after finding the first valid candidate pair following the specified ICE procedures is  $1.5*RTT + Ta*(Additional\_Flows-1)$ , where  $Ta$  is the pacing timer. That assumes a message in one direction, immediately followed by a check back. The reason it isn't more, is that ICE first finds one candidate pair that works prior to attempting to establish multiple flows. Thus, there is no extra time until one has found a working candidate pair. Based on that working pair, the extra time is needed to in parallel establish the, in most cases 2-3, additional flows. However, packet loss causes extra delays, at least 500 ms, which is the minimal retransmission timer for ICE.

**NAT Traversal Failure Rate:** Due to the need to establish more than a single flow through the NAT, there is some risk that establishing the first flow succeeds but that one or more of the additional flows fail. The risk that this happens is hard to quantify, but ought to be fairly low as one flow from the same interfaces has just been successfully established. Thus only rare events such as NAT resource overload, or selecting particular port numbers that are filtered etc., ought to be reasons for failure.

**Deep Packet Inspection and Multiple Streams:** Firewalls differ in how deeply they inspect packets. Due to all previous issues with firewall and Session Boarder Gateways (SBG) with RTP transport media e.g. in Voice over IP (VoIP) systems, there exists a significant risk that deeply inspecting firewalls will have similar legacy issues with multiple SSRCS as some RTP stack implementations.

Using additional RTP streams in the same RTP session and transport flow does not introduce any additional NAT traversal complexities per RTP stream. This can be compared with normally one or two additional transport flows per RTP session when using multiple RTP sessions. Additional lower layer transport flows will be needed, unless an explicit de-multiplexing layer is added between RTP and the transport protocol. At time of writing no such mechanism was defined.

#### 4.2.3. Multicast

Multicast groups provides a powerful tool for a number of real-time applications, especially the ones that desire broadcast-like behaviours with one endpoint transmitting to a large number of receivers, like in IPTV. There is also the RTP/RTCP extension to better support Source Specific Multicast (SSM) [RFC5760]. Many-to-many communication, which RTP [RFC3550] was originally built to support, has several limitations in common with multicast.

One limitation is that, for any group, sender side adaptation with the intent to suit all receivers would have to adapt to the most limited receiver experiencing the worst conditions among the group participants, which imposes degradation for all participants. For broadcast-type applications with a large number of receivers, this is not acceptable. Instead, various receiver-based solutions are employed to ensure that the receivers achieve best possible performance. By using scalable encoding and placing each scalability layer in a different multicast group, the receiver can control the amount of traffic it receives. To have each scalability layer on a different multicast group, one RTP session per multicast group is used.

In addition, the transport flow considerations in multicast are a bit different from unicast; NATs with port translation are not useful in the multicast environment, meaning that the entire port range of each multicast address is available for distinguishing between RTP sessions.

Thus, when using broadcast applications it appears easiest and most straightforward to use multiple RTP sessions for sending different media flows used for adapting to network conditions. It is also common that streams improving transport robustness are sent in their own multicast group to allow for interworking with legacy or to support different levels of protection.

Many-to-many applications have different needs and the most appropriate multiplexing choice will depend on how the actual application is realized. Multicast applications that are capable of using sender side congestion control can avoid the use of multiple

multicast sessions and RTP sessions that result from use of receiver side congestion control.

The properties of a broadcast application using RTP multicast:

1. Uses a group of RTP sessions, not just one. Each endpoint will need to be a member of a number of RTP sessions in order to perform well.
2. Within each RTP session, the number of RTP receivers is likely to be much larger than the number of RTP senders.
3. The applications need signalling functions to identify the relationships between RTP sessions.
4. The applications need signalling or RTP/RTCP functions to identify the relationships between SSRCs in different RTP sessions when needs beyond CNAME exist.

Both broadcast and many-to-many multicast applications share a signalling requirement; all of the participants need the same RTP and payload type configuration. Otherwise, A could for example be using payload type 97 as the video codec H.264 while B thinks it is MPEG-2. SDP offer/answer [RFC3264] is not appropriate for ensuring this property in broadcast/multicast context. The signalling aspects of broadcast/multicast are not explored further in this memo.

Security solutions for this type of group communication are also challenging. First, the key-management and the security protocol need to support group communication. Second, source authentication requires special solutions. For more discussion on this please review Options for Securing RTP Sessions [RFC7201].

#### 4.3. Security and Key Management Considerations

When dealing with point-to-point, 2-member RTP sessions only, there are few security issues that are relevant to the choice of having one RTP session or multiple RTP sessions. However, there are a few aspects of multiparty sessions that might warrant consideration. For general information of possible methods of securing RTP, please review RTP Security Options [RFC7201].

##### 4.3.1. Security Context Scope

When using SRTP [RFC3711], the security context scope is important and can be a necessary differentiation in some applications. As SRTP's crypto suites are (so far) built around symmetric keys, the receiver will need to have the same key as the sender. This results

in that no one in a multi-party session can be certain that a received packet really was sent by the claimed sender and not by another party having access to the key. The single SRTP algorithm not having this property is the TESLA source authentication [RFC4383]. However, TESLA adds delay to achieve source authentication. In most cases, symmetric ciphers provide sufficient security properties but create issues in a few cases.

The first case is when someone leaves a multi-party session and one wants to ensure that the party that left can no longer access the RTP streams. This requires that everyone re-keys without disclosing the new keys to the excluded party.

A second case is when using security as an enforcing mechanism for stream access differentiation between different receivers. Take for example a scalable layer or a high quality simulcast version that only users paying a premium are allowed to access. The mechanism preventing a receiver from getting the high quality stream can be based on the stream being encrypted with a key that user can't access without paying premium, using the key-management to limit access to the key.

SRTP [RFC3711] as specified uses per SSRC unique keys, however the original assumption was a single session master key from which SSRC specific RTP and RTCP keys were derived. However, that assumption was proven incorrect, as the application usage and the developed key-management mechanisms have chosen many different methods for ensuring SSRC unique keys. The key-management functions have different capabilities to establish different sets of keys, normally on a per-endpoint basis. For example, DTLS-SRTP [RFC5764] and Security Descriptions [RFC4568] establish different keys for outgoing and incoming traffic from an endpoint. This key usage has to be written into the cryptographic context, possibly associated with different SSRCs. Thus, limitations do exist depending on chosen key-management method and due to integration of particular implementations of the key-management and SRTP.

#### 4.3.2. Key Management for Multi-party Sessions

The capabilities of the key-management combined with the RTP multiplexing choices affects the resulting security properties, control over the secured media, and who have access to it.

Multi-party sessions contain at least one RTP stream from each active participant. Depending on the multi-party topology [RFC7667], each participant can both send and receive multiple RTP streams. Transport translator-based sessions (Topo-Trn-Translator) and multicast sessions (Topo-ASM), can neither use Security Description

[RFC4568] nor DTLS-SRTP [RFC5764] without an extension as each endpoint provides its set of keys. In centralised conferences, the signalling counterpart is a conference server, and the transport translator is the media plane unicast counterpart (to which DTLS messages would be sent). Thus, an extension like Encrypted Key Transport [I-D.ietf-perc-srtp-ekt-diet] or a MIKEY [RFC3830] based solution that allows for keying all session participants with the same master key is needed.

Privacy Enhanced RTP Conferencing (PERC) also enables a different trust model with semi-trusted media switching RTP middleboxes [I-D.ietf-perc-private-media-framework].

#### 4.3.3. Complexity Implications

The usage of security functions can surface complexity implications from the choice of multiplexing and topology. This becomes especially evident in RTP topologies having any type of middlebox that processes or modifies RTP/RTCP packets. While there is very small overhead for an RTP translator or mixer to rewrite an SSRC value in the RTP packet of an unencrypted session, the cost is higher when using cryptographic security functions. For example, if using SRTP [RFC3711], the actual security context and exact crypto key are determined by the SSRC field value. If one changes SSRC, the encryption and authentication must use another key. Thus, changing the SSRC value implies a decryption using the old SSRC and its security context, followed by an encryption using the new one.

### 5. RTP Multiplexing Design Choices

This section discusses how some RTP multiplexing design choices can be used in applications to achieve certain goals, and a summary of the implications of such choices. For each design there is discussion of benefits and downsides.

#### 5.1. Multiple Media Types in One Session

This design uses a single RTP session for multiple different media types, like audio and video, and possibly also transport robustness mechanisms like FEC or retransmission. An endpoint can send zero, one or more media sources per media type, resulting in a number of RTP streams of various media types for both source and redundancy streams.

The Advantages:

1. Only a single RTP session is used, which implies:

- \* Minimal need to keep NAT/FW state.
  - \* Minimal NAT/FW-traversal cost.
  - \* Fate-sharing for all media flows.
  - \* Minimal overhead for security association establishment.
2. Dynamic allocation of RTP streams can be handled almost entirely at RTP level. How localized this can be kept to RTP level depends on the application's needs for explicit indication of the stream usage and how timely that can be signalled.

The Disadvantages:

- a. It is less suitable for interworking with other applications that use individual RTP sessions per media type or multiple sessions for a single media type, due to the risk of SSRC collision and thus potential need for SSRC translation.
- b. Negotiation of individual bandwidths for the different media types is currently only possible in SDP when using RID [I-D.ietf-mmusic-rid].
- c. It is not suitable for Split Component Terminal (see Section 3.10 of [RFC7667]).
- d. Flow-based QoS cannot be used to provide separate treatment of RTP streams compared to others in the single RTP session.
- e. If there is significant asymmetry between the RTP streams' RTCP reporting needs, there are some challenges in configuration and usage to avoid wasting RTCP reporting on the RTP stream that does not need that frequent reporting.
- f. It is not suitable for applications where some receivers like to receive only a subset of the RTP streams, especially if multicast or transport translator is being used.
- g. There is some additional concern with legacy implementations that do not support the RTP specification fully when it comes to handling multiple SSRC per endpoint, as multiple simultaneous media types are sent as separate SSRC in the same RTP session.
- h. If the applications need finer control over which session participants are included in different sets of security associations, most key-management mechanisms will have difficulties establishing such a session.

## 5.2. Multiple SSRCs of the Same Media Type

In this design, each RTP session serves only a single media type. The RTP session can contain multiple RTP streams, either from a single endpoint or from multiple endpoints. This commonly creates a low number of RTP sessions, typically only one for audio and one for video, with a corresponding need for two listening ports when using RTP/RTCP multiplexing [RFC5761].

### The Advantages

1. It works well with Split Component Terminal (see Section 3.10 of [RFC7667]) where the split is per media type.
2. It enables flow-based QoS with different prioritisation between media types.
3. For applications with dynamic usage of RTP streams, i.e. frequently added and removed, having much of the state associated with the RTP session rather than per individual SSRC can avoid the need for in-session signalling of meta-information about each SSRC. In the simple cases this allows for unsignalled RTP streams where session level information and RTCP SDES item (e.g. CNAME) are sufficient. In the more complex cases where more source-specific metadata needs to be signalled the SSRC can be associated with an intermediate identifier, e.g. the MID conveyed as an SDES item as defined in Section 15 of [I-D.ietf-mmusic-sdp-bundle-negotiation].
4. There is low overhead for security association establishment.

### The Disadvantages

- a. There are a slightly higher number of RTP sessions needed compared to Multiple Media Types in one Session Section 5.1. This implies:
  - \* More NAT/FW state is needed.
  - \* There is increased NAT/FW-traversal cost in both processing and delay.
- b. There is some potential for concern with legacy implementations that don't support the RTP specification fully when it comes to handling multiple SSRC per endpoint.
- c. It is not possible to control security association for sets of RTP streams within the same media type with today's key-

management mechanisms, unless these are split into different RTP sessions (Section 5.3).

For RTP applications where all RTP streams of the same media type share same usage, this structure provides efficiency gains in amount of network state used and provides more fate sharing with other media flows of the same type. At the same time, it is still maintaining almost all functionalities for the negotiation signaling of properties per individual media type, and also enables flow based QoS prioritisation between media types. It handles multi-party sessions well, independently of multicast or centralised transport distribution, as additional sources can dynamically enter and leave the session.

### 5.3. Multiple Sessions for One Media Type

This design goes one step further than above (Section 5.2) by using multiple RTP sessions also for a single media type. The main reason for going in this direction is that the RTP application needs separation of the RTP streams due to their usage, such as e.g. scalability over multicast, simulcast, need for extended QoS prioritisation, or the need for fine-grained signalling using RTP session-focused signalling tools.

The Advantages:

1. This is more suitable for multicast usage where receivers can individually select which RTP sessions they want to participate in, assuming each RTP session has its own multicast group.
2. The application can indicate its usage of the RTP streams on RTP session level, when multiple different usages exist.
3. There is less need for SSRC-specific explicit signalling for each media stream and thus reduced need for explicit and timely signalling when RTP streams are added or removed.
4. It enables detailed QoS prioritisation for flow-based mechanisms.
5. It works well with Split Component Terminal (see Section 3.10 of [RFC7667]).
6. The scope for who is included in a security association can be structured around the different RTP sessions, thus enabling such functionality with existing key-management.

The Disadvantages:

- a. There is an increased amount of session configuration state compared to Multiple SSRCs of the Same Media Type, due to the increased amount of RTP sessions.
- b. For RTP streams that are part of scalability, simulcast or transport robustness, a method to bind sources across multiple RTP sessions is needed.
- c. There is some potential for concern with legacy implementations that don't support the RTP specification fully when it comes to handling multiple SSRC per endpoint.
- d. There is higher overhead for security association establishment, due to the increased number of RTP sessions.
- e. If the applications need more fine-grained control than per RTP session over which participants that are included in different sets of security associations, most of today's key-management will have difficulties establishing such a session.

For more complex RTP applications that have several different usages for RTP streams of the same media type, or uses scalability or simulcast, this solution can enable those functions at the cost of increased overhead associated with the additional sessions. This type of structure is suitable for more advanced applications as well as multicast-based applications requiring differentiation to different participants.

#### 5.4. Single SSRC per Endpoint

In this design each endpoint in a point-to-point session has only a single SSRC, thus the RTP session contains only two SSRCs, one local and one remote. This session can be used both unidirectional, i.e. only a single RTP stream, or bi-directional, i.e. both endpoints have one RTP stream each. If the application needs additional media flows between the endpoints, it will have to establish additional RTP sessions.

The Advantages:

1. This design has great legacy interoperability potential as it will not tax any RTP stack implementations.
2. The signalling has good possibilities to negotiate and describe the exact formats and bitrates for each RTP stream, especially using today's tools in SDP.

3. It is possible to control security association per RTP stream with current key-management, since each RTP stream is directly related to an RTP session, and the most used keying mechanisms operates on a per-session basis.

The Disadvantages:

- a. There is a linear growth of the amount of NAT/FW state with number of RTP streams.
- b. There is increased delay and resource consumption from NAT/FW-traversal.
- c. There are likely larger signalling message and signalling processing requirements due to the increased amount of session-related information.
- d. There is higher potential for a single RTP stream to fail during transport between the endpoints, due to the need for separate NAT/FW- traversal for every RTP stream since there is only one stream per session.
- e. The amount of explicit state for relating RTP streams grows, depending on how the application relates RTP streams.
- f. The port consumption might become a problem for centralised services, where the central node's port or 5-tuple filter consumption grows rapidly with the number of sessions.
- g. For applications where the RTP stream usage is highly dynamic, i.e. entering and leaving, the amount of signalling can become high. Issues can also arise from the need for timely establishment of additional RTP sessions.
- h. If, against the recommendation, the same SSRC value is reused in multiple RTP sessions rather than being randomly chosen, interworking with applications that use a different multiplexing structure will require SSRC translation.

RTP applications with a strong need to interwork with legacy RTP applications can potentially benefit from this structure. However, a large number of media descriptions in SDP can also run into issues with existing implementations. For any application needing a larger number of media flows, the overhead can become very significant. This structure is also not suitable for non-mixed multi-party sessions, as any given RTP stream from each participant, although having same usage in the application, needs its own RTP session. In addition, the dynamic behaviour that can arise in multi-party

applications can tax the signalling system and make timely media establishment more difficult.

### 5.5. Summary

Both the "Single SSRC per Endpoint" and the "Multiple Media Types in One Session" are cases that require full explicit signalling of the media stream relations. However, they operate on two different levels where the first primarily enables session level binding, and the second needs SSRC level binding. From another perspective, the two solutions are the two extreme points when it comes to number of RTP sessions needed.

The two other designs, "Multiple SSRCs of the Same Media Type" and "Multiple Sessions for One Media Type", are two examples that primarily allows for some implicit mapping of the role or usage of the RTP streams based on which RTP session they appear in. It thus potentially allows for less signalling and in particular reduces the need for real-time signalling in sessions with dynamically changing number of RTP streams. They also represent points in-between the first two designs when it comes to amount of RTP sessions established, i.e. representing an attempt to balance the amount of RTP sessions with the functionality the communication session provides both on network level and on signalling level.

## 6. Guidelines

This section contains a number of multi-stream guidelines for implementers, system designers, or specification writers.

Do not require use of the same SSRC value across RTP sessions:

As discussed in Section 3.4.3 there exist drawbacks in using the same SSRC in multiple RTP sessions as a mechanism to bind related RTP streams together. It is instead recommended to use a mechanism to explicitly signal the relation, either in RTP/RTCP or in the signalling mechanism used to establish the RTP session(s).

Use additional RTP streams for additional media sources: In the cases where an RTP endpoint needs to transmit additional RTP streams of the same media type in the application, with the same processing requirements at the network and RTP layers, it is suggested to send them in the same RTP session. For example a telepresence room where there are three cameras, and each camera captures 2 persons sitting at the table, sending each camera as its own RTP stream within a single RTP session is suggested.

Use additional RTP sessions for streams with different requirements:

When RTP streams have different processing requirements from the network or the RTP layer at the endpoints, it is suggested that the different types of streams are put in different RTP sessions. This includes the case where different participants want different subsets of the set of RTP streams.

When using multiple RTP sessions, use grouping: When using multiple RTP session solutions, it is suggested to explicitly group the involved RTP sessions when needed using a signalling mechanism, for example The Session Description Protocol (SDP) Grouping Framework [RFC5888], using some appropriate grouping semantics.

RTP/RTCP Extensions Support Multiple RTP Streams as Well as Multiple RTP Sessions:

When defining an RTP or RTCP extension, the creator needs to consider if this extension is applicable to use with additional SSRCs and multiple RTP sessions. Any extension intended to be generic must support both. Extensions that are not as generally applicable will have to consider if interoperability is better served by defining a single solution or providing both options.

Extensions for Transport Support: When defining new RTP/RTCP extensions intended for transport support, like the retransmission or FEC mechanisms, they must include support for both multiple RTP streams in the same RTP session and multiple RTP sessions, such that application developers can choose freely from the set of mechanisms without concerning themselves with which of the multiplexing choices a particular solution supports.

## 7. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section can be removed on publication as an RFC.

## 8. Security Considerations

The security considerations of the RTP specification [RFC3550], any applicable RTP profile [RFC3551],[RFC4585],[RFC3711], and the extensions for sending multiple media types in a single RTP session [I-D.ietf-avtcore-multi-media-rtp-session], RID [I-D.ietf-mmusic-rid], BUNDLE [I-D.ietf-mmusic-sdp-bundle-negotiation], [RFC5760], [RFC5761], apply if selected and thus need to be considered in the evaluation.

There is discussion of the security implications of choosing multiple SSRC vs multiple RTP sessions in Section 4.3.

## 9. Contributors

Hui Zheng (Marvin) contributed to WG draft versions -04 and -05 of the document.

## 10. Acknowledgments

The Authors like to acknowledge and thank Cullen Jennings, Dale R Worley, Huang Yihong (Rachel), Benjamin Kaduk, Mirja Kuehlewind, and Vijay Gurbani for review and comments.

## 11. References

### 11.1. Normative References

- [I-D.ietf-avtcore-multi-media-rtp-session]  
Westerlund, M., Perkins, C., and J. Lennox, "Sending Multiple Types of Media in a Single RTP Session", draft-ietf-avtcore-multi-media-rtp-session-13 (work in progress), December 2015.
- [I-D.ietf-mmusic-rid]  
Roach, A., "RTP Payload Format Restrictions", draft-ietf-mmusic-rid-15 (work in progress), May 2018.
- [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-54 (work in progress), December 2018.
- [I-D.ietf-perc-srtp-ekt-diet]  
Jennings, C., Mattsson, J., McGrew, D., Wing, D., and F. Andreassen, "Encrypted Key Transport for DTLS and Secure RTP", draft-ietf-perc-srtp-ekt-diet-11 (work in progress), January 2020.
- [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>>.

- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [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>>.
- [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>>.
- [RFC5760] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback", RFC 5760, DOI 10.17487/RFC5760, February 2010, <<https://www.rfc-editor.org/info/rfc5760>>.
- [RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, DOI 10.17487/RFC5761, April 2010, <<https://www.rfc-editor.org/info/rfc5761>>.
- [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>>.
- [RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <<https://www.rfc-editor.org/info/rfc7667>>.

## 11.2. Informative References

- [I-D.ietf-avtext-rid]  
Roach, A., Nandakumar, S., and P. Thatcher, "RTP Stream Identifier Source Description (SDS)", draft-ietf-avtext-rid-09 (work in progress), October 2016.

- [I-D.ietf-perc-private-media-framework]  
Jones, P., Benham, D., and C. Groves, "A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing (PERC)", draft-ietf-perc-private-media-framework-12 (work in progress), June 2019.
- [JINGLE] Ludwig, S., Beda, J., Saint-Andre, P., McQueen, R., Egan, S., and J. Hildebrand, "XEP-0166: Jingle", XMPP.org <https://xmpp.org/extensions/xep-0166.html>, September 2018.
- [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, DOI 10.17487/RFC2198, September 1997, <<https://www.rfc-editor.org/info/rfc2198>>.
- [RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <<https://www.rfc-editor.org/info/rfc2205>>.
- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <<https://www.rfc-editor.org/info/rfc2474>>.
- [RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, DOI 10.17487/RFC2974, October 2000, <<https://www.rfc-editor.org/info/rfc2974>>.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <<https://www.rfc-editor.org/info/rfc3261>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.
- [RFC3389] Zopf, R., "Real-time Transport Protocol (RTP) Payload for Comfort Noise (CN)", RFC 3389, DOI 10.17487/RFC3389, September 2002, <<https://www.rfc-editor.org/info/rfc3389>>.

- [RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, DOI 10.17487/RFC3830, August 2004, <<https://www.rfc-editor.org/info/rfc3830>>.
- [RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <<https://www.rfc-editor.org/info/rfc4103>>.
- [RFC4383] Baugher, M. and E. Carrara, "The Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Secure Real-time Transport Protocol (SRTP)", RFC 4383, DOI 10.17487/RFC4383, February 2006, <<https://www.rfc-editor.org/info/rfc4383>>.
- [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>>.
- [RFC4568] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, DOI 10.17487/RFC4568, July 2006, <<https://www.rfc-editor.org/info/rfc4568>>.
- [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>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.
- [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>>.
- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <<https://www.rfc-editor.org/info/rfc5389>>.
- [RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <<https://www.rfc-editor.org/info/rfc5764>>.

- [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>>.
- [RFC6465] Ivov, E., Ed., Marocco, E., Ed., and J. Lennox, "A Real-time Transport Protocol (RTP) Header Extension for Mixer-to-Client Audio Level Indication", RFC 6465, DOI 10.17487/RFC6465, December 2011, <<https://www.rfc-editor.org/info/rfc6465>>.
- [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>>.
- [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>>.
- [RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <<https://www.rfc-editor.org/info/rfc7826>>.
- [RFC7983] Petit-Huguenin, M. and G. Salgueiro, "Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)", RFC 7983, DOI 10.17487/RFC7983, September 2016, <<https://www.rfc-editor.org/info/rfc7983>>.
- [RFC8088] Westerlund, M., "How to Write an RTP Payload Format", RFC 8088, DOI 10.17487/RFC8088, May 2017, <<https://www.rfc-editor.org/info/rfc8088>>.
- [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>>.
- [RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018, <<https://www.rfc-editor.org/info/rfc8445>>.

## Appendix A. Dismissing Payload Type Multiplexing

This section documents a number of reasons why using the payload type as a multiplexing point is unsuitable for most issues related to multiple RTP streams. Attempting to use Payload type multiplexing beyond its defined usage has well known negative effects on RTP discussed below. To use payload type as the single discriminator for multiple streams implies that all the different RTP streams are being sent with the same SSRC, thus using the same timestamp and sequence number space. This has many effects:

1. Putting constraints on RTP timestamp rate for the multiplexed media. For example, RTP streams that use different RTP timestamp rates cannot be combined, as the timestamp values need to be consistent across all multiplexed media frames. Thus streams are forced to use the same RTP timestamp rate. When this is not possible, payload type multiplexing cannot be used.
2. Many RTP payload formats can fragment a media object over multiple RTP packets, like parts of a video frame. These payload formats need to determine the order of the fragments to correctly decode them. Thus, it is important to ensure that all fragments related to a frame or a similar media object are transmitted in sequence and without interruptions within the object. This can relatively simple be solved on the sender side by ensuring that the fragments of each RTP stream are sent in sequence.
3. Some media formats require uninterrupted sequence number space between media parts. These are media formats where any missing RTP sequence number will result in decoding failure or invoking a repair mechanism within a single media context. The text/T140 payload format [RFC4103] is an example of such a format. These formats will need a sequence numbering abstraction function between RTP and the individual RTP stream before being used with payload type multiplexing.
4. Sending multiple media streams in the same sequence number space makes it impossible to determine which media stream lost a packet. This as the payload type that is used for demultiplex the media stream is not received. Thus, causing the receiver difficulties in determining which stream to apply packet loss concealment or other stream-specific loss mitigation mechanisms.
5. If RTP Retransmission [RFC4588] is used and there is a loss, it is possible to ask for the missing packet(s) by SSRC and sequence number, not by payload type. If only some of the payload type multiplexed streams are of interest, there is no

way of telling which missing packet(s) belong to the interesting stream(s) and all lost packets need be requested, wasting bandwidth.

6. The current RTCP feedback mechanisms are built around providing feedback on RTP streams based on stream ID (SSRC), packet (sequence numbers) and time interval (RTP timestamps). There is almost never a field to indicate which payload type is reported, so sending feedback for a specific RTP payload type is difficult without extending existing RTCP reporting.
7. The current RTCP media control messages [RFC5104] specification is oriented around controlling particular media flows, i.e. requests are done addressing a particular SSRC. Such mechanisms would need to be redefined to support payload type multiplexing.
8. The number of payload types are inherently limited. Accordingly, using payload type multiplexing limits the number of streams that can be multiplexed and does not scale. This limitation is exacerbated if one uses solutions like RTP and RTCP multiplexing [RFC5761] where a number of payload types are blocked due to the overlap between RTP and RTCP.
9. At times, there is a need to group multiplexed streams and this is currently possible for RTP sessions and for SSRC, but there is no defined way to group payload types.
10. It is currently not possible to signal bandwidth requirements per RTP stream when using payload type multiplexing.
11. Most existing SDP media level attributes cannot be applied on a per payload type level and would require re-definition in that context.
12. A legacy endpoint that does not understand the indication that different RTP payload types are different RTP streams might be slightly confused by the large amount of possibly overlapping or identically defined RTP payload types.

#### Appendix B. Signalling Considerations

Signalling is not an architectural consideration for RTP itself, so this discussion has been moved to an appendix. However, it is extremely important for anyone building complete applications, so it is deserving of discussion.

We document salient issues here that need to be addressed by the WGs that use some form of signaling to establish RTP sessions. These

issues cannot simply be addressed by tweaking, extending, or profiling RTP, but require a dedicated and indepth look at the signaling primitives that set up the RTP sessions.

There exist various signalling solutions for establishing RTP sessions. Many are SDP [RFC4566] based, however SDP functionality is also dependent on the signalling protocols carrying the SDP. RTSP [RFC7826] and SAP [RFC2974] both use SDP in a declarative fashion, while SIP [RFC3261] uses SDP with the additional definition of Offer/Answer [RFC3264]. The impact on signalling and especially SDP needs to be considered as it can greatly affect how to deploy a certain multiplexing point choice.

### B.1. Session Oriented Properties

One aspect of the existing signalling is that it is focused on RTP sessions, or in the case of SDP, the media description concept. There are a number of things that are signalled on media description level but those are not necessarily strictly bound to an RTP session and could be of interest to signal specifically for a particular RTP stream (SSRC) within the session. The following properties have been identified as being potentially useful to signal not only on RTP session level:

- o Bitrate/Bandwidth exist today only at aggregate or as a common "any RTP stream" limit, unless either codec-specific bandwidth limiting or RTCP signalling using TMMBR [RFC5104] is used.
- o Which SSRC that will use which RTP payload type (this will be visible from the first media packet, but is sometimes useful to know before packet arrival).

Some of these issues are clearly SDP's problem rather than RTP limitations. However, if the aim is to deploy an solution using additional SSRCs that contains several sets of RTP streams with different properties (encoding/packetization parameter, bit-rate, etc.), putting each set in a different RTP session would directly enable negotiation of the parameters for each set. If insisting on additional SSRC only, a number of signalling extensions are needed to clarify that there are multiple sets of RTP streams with different properties and that they need in fact be kept different, since a single set will not satisfy the application's requirements.

For some parameters, such as RTP payload type, resolution and framerate, a SSRC-linked mechanism has been proposed in [I-D.ietf-mmusic-rid]

## B.2. SDP Prevents Multiple Media Types

SDP chose to use the m= line both to delineate an RTP session and to specify the top level of the MIME media type; audio, video, text, image, application. This media type is used as the top-level media type for identifying the actual payload format and is bound to a particular payload type using the rtpmap attribute. This binding has to be loosened in order to use SDP to describe RTP sessions containing multiple MIME top level types.

[I-D.ietf-mmusic-sdp-bundle-negotiation] describes how to let multiple SDP media descriptions use a single underlying transport in SDP, which allows to define one RTP session with media types having different MIME top level types.

## B.3. Signalling RTP Stream Usage

RTP streams being transported in RTP have some particular usage in an RTP application. This usage of the RTP stream is in many applications so far implicitly signalled. For example, an application might choose to take all incoming audio RTP streams, mix them and play them out. However, in more advanced applications that use multiple RTP streams there will be more than a single usage or purpose among the set of RTP streams being sent or received. RTP applications will need to signal this usage somehow. The signalling used will have to identify the RTP streams affected by their RTP-level identifiers, which means that they have to be identified either by their session or by their SSRC + session.

In some applications, the receiver cannot utilise the RTP stream at all before it has received the signalling message describing the RTP stream and its usage. In other applications, there exists a default handling that is appropriate.

If all RTP streams in an RTP session are to be treated in the same way, identifying the session is enough. If SSRCs in a session are to be treated differently, signalling needs to identify both the session and the SSRC.

If this signalling affects how any RTP central node, like an RTP mixer or translator that selects, mixes or processes streams, treats the streams, the node will also need to receive the same signalling to know how to treat RTP streams with different usage in the right fashion.

## Authors' Addresses

Magnus Westerlund  
Ericsson  
Torshamnsgatan 23  
SE-164 80 Kista  
Sweden

Phone: +46 10 714 82 87  
Email: [magnus.westerlund@ericsson.com](mailto:magnus.westerlund@ericsson.com)

Bo Burman  
Ericsson  
Gronlandsgatan 31  
SE-164 60 Kista  
Sweden

Email: [bo.burman@ericsson.com](mailto:bo.burman@ericsson.com)

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

Email: [csp@csp Perkins.org](mailto:csp@csp Perkins.org)

Harald Tveit Alvestrand  
Google  
Kungsbron 2  
Stockholm 11122  
Sweden

Email: [harald@alvestrand.no](mailto:harald@alvestrand.no)

Roni Even

Email: [ron.even.tlv@gmail.com](mailto:ron.even.tlv@gmail.com)

AVTCORE WG  
Internet-Draft  
Intended status: Standards Track  
Expires: September 3, 2016

J. Lennox  
Vidyo  
M. Westerlund  
Ericsson  
Q. Wu  
Huawei  
C. Perkins  
University of Glasgow  
March 2, 2016

Sending Multiple RTP Streams in a Single RTP Session: Grouping RTCP  
Reception Statistics and Other Feedback  
draft-ietf-avtcORE-rtp-multi-stream-optimisation-12

Abstract

RTP allows multiple RTP streams to be sent in a single session, but requires each Synchronisation Source (SSRC) to send RTCP reception quality reports for every other SSRC visible in the session. This causes the number of RTCP reception reports to grow with the number of SSRCs, rather than the number of endpoints. In many cases most of these RTCP reception reports are unnecessary, since all SSRCs of an endpoint are normally co-located and see the same reception quality. This memo defines a Reporting Group extension to RTCP to reduce the reporting overhead in such scenarios.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 3, 2016.

## Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
3. RTCP Reporting Groups . . . . .	3
3.1. Semantics and Behaviour of RTCP Reporting Groups . . . . .	4
3.2. Identifying Members of an RTCP Reporting Group . . . . .	5
3.2.1. Definition and Use of the RTCP RGRP SDES Item . . . . .	5
3.2.2. Definition and Use of the RTCP RGRS Packet . . . . .	6
3.3. Interactions with the RTP/AVPF Feedback Profile . . . . .	8
3.4. Interactions with RTCP Extended Report (XR) Packets . . . . .	9
3.5. Middlebox Considerations . . . . .	9
3.6. SDP Signalling for Reporting Groups . . . . .	10
4. Properties of RTCP Reporting Groups . . . . .	12
4.1. Bandwidth Benefits of RTCP Reporting Groups . . . . .	12
4.2. Compatibility of RTCP Reporting Groups . . . . .	13
5. Security Considerations . . . . .	13
6. IANA Considerations . . . . .	15
7. References . . . . .	16
7.1. Normative References . . . . .	16
7.2. Informative References . . . . .	16
Authors' Addresses . . . . .	18

## 1. Introduction

The Real-time Transport Protocol (RTP) [RFC3550] is a protocol for group communication, supporting multiparty multimedia sessions. A single RTP session can support multiple participants sending at once, and can also support participants sending multiple simultaneous RTP streams. Examples of the latter might include a participant with multiple cameras who chooses to send multiple views of a scene, or a participant that sends audio and video flows multiplexed in a single RTP session. Rules for handling RTP sessions containing multiple RTP

streams are described in [RFC3550] with some clarifications in [I-D.ietf-avtcore-rtp-multi-stream].

An RTP endpoint will have one or more synchronisation sources (SSRCs). It will have at least one RTP Stream, and thus SSRC, for each media source it sends, and might use multiple SSRCs per media source when using media scalability features [RFC6190], forward error correction, RTP retransmission [RFC4588], or similar mechanisms. An endpoint that is not sending any RTP stream, will have at least one SSRC to use for reporting and any feedback messages. Each SSRC has to send RTCP sender reports corresponding to the RTP packets it sends, and receiver reports for traffic it receives. That is, every SSRC will send RTCP packets to report on every other SSRC. This rule is simple, but can be quite inefficient for endpoints that send large numbers of RTP streams in a single RTP session. Consider a session comprising ten participants, each sending three media sources, each with their own RTP stream. There will be 30 SSRCs in such an RTP session, and each of those 30 SSRCs will send an RTCP Sender Report/Receiver Report packet (containing several report blocks) per reporting interval as each SSRC reports on all the others. However, the three SSRCs comprising each participant are commonly co-located such that they see identical reception quality. If there was a way to indicate that several SSRCs are co-located, and see the same reception quality, then two-thirds of those RTCP reports could be suppressed. This would allow the remaining RTCP reports to be sent more often, while keeping within the same RTCP bandwidth fraction.

This memo defines such an RTCP extension, RTCP Reporting Groups. This extension is used to indicate the SSRCs that originate from the same endpoint, and therefore have identical reception quality, hence allowing the endpoints to suppress unnecessary RTCP reception quality reports.

## 2. Terminology

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

## 3. RTCP Reporting Groups

An RTCP Reporting Group is a set of synchronization sources (SSRCs) that are co-located at a single endpoint (which could be an end host or a middlebox) in an RTP session. Since they are co-located, every SSRC in the RTCP reporting group will have an identical view of the network conditions, and see the same lost packets, jitter, etc. This allows a single representative to send RTCP reception quality reports

on behalf of the rest of the reporting group, reducing the number of RTCP packets that need to be sent without loss of information.

### 3.1. Semantics and Behaviour of RTCP Reporting Groups

A group of co-located SSRCs that see identical network conditions can form an RTCP reporting group. If reporting groups are in use, an RTP endpoint with multiple SSRCs MAY put those SSRCs into a reporting group if their view of the network is identical; i.e., if they report on traffic received at the same interface of an RTP endpoint. SSRCs with different views of the network MUST NOT be put into the same reporting group.

An endpoint that has combined its SSRCs into an RTCP reporting group will choose one (or a subset) of those SSRCs to act as "reporting source(s)" for that RTCP reporting group. A reporting source will send RTCP SR/RR reception quality reports on behalf of the other members of the RTCP reporting group. A reporting source MUST suppress the RTCP SR/RR reports that relate to other members of the reporting group, and only report on remote SSRCs. The other members (non reporting sources) of the RTCP reporting group will suppress their RTCP reception quality reports, and instead send an RTCP RGRS packet (see Section 3.2.2) to indicate that they are part of an RTCP reporting group and give the SSRCs of the reporting sources.

If there are large numbers of remote SSRCs in the RTP session, then the reception quality reports generated by the reporting source might grow too large to fit into a single compound RTCP packet, forcing the reporting source to use a round-robin policy to determine what remote SSRCs it includes in each compound RTCP packet, and so reducing the frequency of reports on each SSRC. To avoid this, in sessions with large numbers of remote SSRCs, an RTCP reporting group MAY use more than one reporting source. If several SSRCs are acting as reporting sources for an RTCP reporting group, then each reporting source MUST have non-overlapping sets of remote SSRCs it reports on.

An endpoint MUST NOT create an RTCP reporting group that comprises only a single local SSRC (i.e., an RTCP reporting group where the reporting source is the only member of the group), unless it is anticipated that the group might have additional SSRCs added to it in the future.

If a reporting source leaves the RTP session (i.e., if it sends a RTCP BYE packet, or leaves the session without sending BYE under the rules of [RFC3550] section 6.3.7), the remaining members of the RTCP reporting group MUST either (a) have another reporting source, if one exists, report on the remote SSRCs the leaving SSRC reported on, (b) choose a new reporting source, or (c) disband the RTCP reporting

group and begin sending reception quality reports following [RFC3550] and [I-D.ietf-avtcore-rtp-multi-stream].

The RTCP timing rules assign different bandwidth fractions to senders and receivers. This lets senders transmit RTCP reception quality reports more often than receivers. If a reporting source in an RTCP reporting group is a receiver, but one or more non-reporting SSRCs in the RTCP reporting group are senders, then the endpoint MAY treat the reporting source as a sender for the purpose of RTCP bandwidth allocation, increasing its RTCP bandwidth allocation, provided it also treats one of the senders as if it were a receiver and makes the corresponding reduction in RTCP bandwidth for that SSRC. However, the application needs to consider the impact on the frequency of transmitting of the synchronization information included in RTCP Sender Reports.

### 3.2. Identifying Members of an RTCP Reporting Group

When RTCP Reporting Groups are in use, the other SSRCs in the RTP session need to be able to identify which SSRCs are members of an RTCP reporting group. Two RTCP extensions are defined to support this: the RTCP RGRP SDES item is used by the reporting source(s) to identify an RTCP reporting group, and the RTCP RGRS packet is used by other members of an RTCP reporting group to identify the reporting source(s).

#### 3.2.1. Definition and Use of the RTCP RGRP SDES Item

This document defines a new RTCP SDES item to identify an RTCP reporting group. The motivation for giving a reporting group an identify is to ensure that the RTCP reporting group and its member SSRCs can be correctly associated when there are multiple reporting sources, and to ensure that a reporting SSRC can be associated with the correct reporting group if an SSRC collision occurs.

This document defines the RTCP Source Description (SDES) RGRP item. The RTCP SDES RGRP item MUST be sent by the reporting sources in a reporting group, and MUST NOT be sent by other members of the reporting group or by SSRCs that are not members of any RTCP reporting group. Specifically, every reporting source in an RTCP reporting group MUST include an RTCP SDES packet containing an RGRP item in every compound RTCP packet in which it sends an RR or SR packet (i.e., in every RTCP packet it sends, unless Reduced-Size RTCP [RFC5506] is in use).

Syntactically, the format of the RTCP SDES RGRP item is identical to that of the RTCP SDES CNAME item [RFC7022], except that the SDES item type field MUST have value RGRP=(TBA) instead of CNAME=1. The value

of the RTCP SDES RGRP item MUST be chosen with the same concerns about global uniqueness and the same privacy considerations as the RTCP SDES CNAME. The value of the RTCP SDES RGRP item MUST be stable throughout the lifetime of the reporting group, even if some or all of the reporting sources change their SSRC due to collisions, or if the set of reporting sources changes.

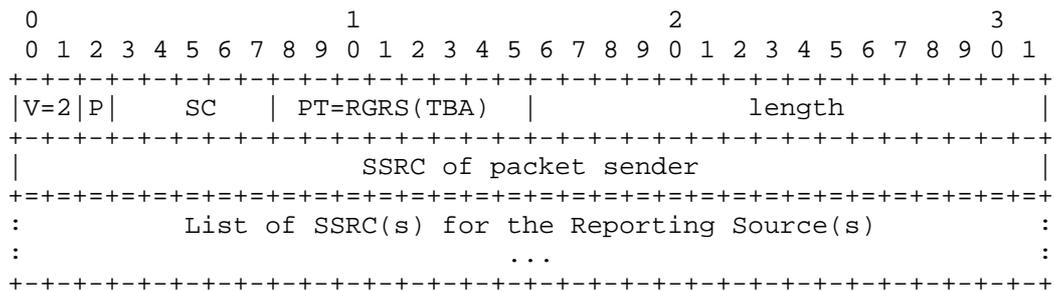
Note to RFC Editor: please replace (TBA) in the above paragraph with the RTCP SDES item type number assigned to the RGRP item, then delete this note.

An RTP mixer or translator that forwards RTCP SR or RR packets from members of a reporting group MUST forward the corresponding RTCP SDES RGRP items as well, even if it otherwise strips SDES items other than the CNAME item.

3.2.2. Definition and Use of the RTCP RGRS Packet

A new RTCP packet type is defined to allow the members of an RTCP reporting group to identify the reporting sources for that group. This allows participants in an RTP session to distinguish an SSRC that is sending empty RTCP reception reports because it is a member of an RTCP reporting group, from an SSRC that is sending empty RTCP reception reports because it is not receiving any traffic. It also explicitly identifies the reporting sources, allowing other members of the RTP session to know which SSRCs are acting as the reporting sources for an RTCP reporting group, and allowing them to detect if RTCP packets from any of the reporting sources are being lost.

The format of the RTCP RGRS packet is defined below. It comprises the fixed RTCP header that indicates the packet type and length, the SSRC of the packet sender, and a list of reporting sources for the RTCP reporting group of which the packet sender is a member.



The fields in the RTCP RGRS packet have the following definition:

version (V): 2 bits unsigned integer. This field identifies the RTP version. The current RTP version is 2.

padding (P): 1 bit. If set, the padding bit indicates that the RTCP packet contains additional padding octets at the end that are not part of the control information but are included in the length field. See [RFC3550].

Source Count (SC): 5 bits unsigned integer. Indicates the number of reporting source SSRCs that are included in this RTCP packet. As the RTCP RGRS packet MUST NOT be sent by reporting sources, all the SSRCs in the list of reporting sources will be different from the SSRC of the packet sender. Every RTCP RGRS packet MUST contain at least one reporting source SSRC.

Payload type (PT): 8 bits unsigned integer. The RTCP packet type number that identifies the packet as being an RTCP RGRS packet. The RGRS RTCP packet has the value [TBA].

Note to RFC Editor: please replace [TBA] here, and in the packet format diagram above, with the RTCP packet type that IANA assigns to the RTCP RGRS packet.

Length: 16 bits unsigned integer. The length of this packet in 32-bit words minus one, including the header and any padding. This is in line with the definition of the length field used in RTCP sender and receiver reports [RFC3550]. Since all RTCP RGRS packets include at least one reporting source SSRC, the length will always be 2 or greater.

SSRC of packet sender: 32 bits. The SSRC of the sender of this packet.

List of SSRCs for the Reporting Source(s): A variable length size (as indicated by SC header field) of the 32 bit SSRC values of the reporting sources for the RTCP Reporting Group of which the packet sender is a member.

Every source that belongs to an RTCP reporting group but is not a reporting source MUST include an RTCP RGRS packet in every compound RTCP packet in which it sends an RR or SR packet (i.e., in every RTCP packet it sends, unless Reduced-Size RTCP [RFC5506] is in use). Each RTCP RGRS packet MUST contain the SSRC identifier of at least one reporting source. If there are more reporting sources in an RTCP reporting group than can fit into an RTCP RGRS packet, the members of that reporting group MUST send the SSRCs of the reporting sources in a round-robin fashion in consecutive RTCP RGRS packets, such that all

the SSRCs of the reporting sources are included over the course of several RTCP reporting intervals.

An RTP mixer or translator that forwards RTCP SR or RR packets from members of a reporting group **MUST** also forward the corresponding RGRS RTCP packets. If the RTP mixer or translator rewrites SSRC values of the packets it forwards, it **MUST** make the corresponding changes to the RTCP RGRS packets.

### 3.3. Interactions with the RTP/AVPF Feedback Profile

Use of the RTP/AVPF Feedback Profile [RFC4585] allows SSRCs to send rapid RTCP feedback requests and codec control messages. If use of the RTP/AVPF profile has been negotiated in an RTP session, members of an RTCP reporting group can send rapid RTCP feedback and codec control messages following [RFC4585] and [RFC5104], as updated by Section 5.4 of [I-D.ietf-avtcore-rtp-multi-stream], and by the following considerations.

The members of an RTCP reporting group will all see identical network conditions. Accordingly, one might therefore think that it doesn't matter which SSRC in the reporting group sends the RTP/AVPF feedback or codec control messages. There might be, however, cases where the sender of the feedback/codec control message has semantic importance, or when only a subset of the members of an RTCP reporting group might want to send RTP/AVPF feedback or a codec control message in response to a particular event. For example, an RTP video sender might choose to treat packet loss feedback received from SSRCs known to be audio receivers with less urgency than feedback that it receives from video receivers when deciding what packets to retransmit, and a multimedia receiver using reporting groups might want to choose the outgoing SSRC for feedback packets to reflect this.

Each member of an RTCP reporting group **SHOULD** therefore send RTP/AVPF feedback/codec control messages independently of the other members of the reporting group, to respect the semantic meaning of the message sender. The suppression rules of [RFC4585] will ensure that only a single copy of each feedback packet is (typically) generated, even if several members of a reporting group send the same feedback. When an endpoint knows that several members of its RTCP reporting group will be sending identical feedback, and that the sender of the feedback is not semantically important, then that endpoint **MAY** choose to send all its feedback from the reporting source and deterministically suppress feedback packets generated by the other sources in the reporting group.

It is important to note that the RTP/AVPF timing rules operate on a per-SSRC basis. Using a single reporting source to send all feedback

for a reporting group will hence limit the amount of feedback that can be sent to that which can be sent by one SSRC. If this limit is a problem, then the reporting group can allow each of its members to send its own feedback, using its own SSRC.

If the RTP/AVPF feedback messages or codec control requests are sent as compound RTCP packets, then those compound RTCP packets MUST include either an RTCP RGRS packet or an RTCP SDES RGRP item, depending on whether they are sent by the reporting source or a non-reporting source in the RTCP reporting group respectively. The contents of non-compound RTCP feedback or codec control messages are not affected by the use of RTCP reporting groups.

#### 3.4. Interactions with RTCP Extended Report (XR) Packets

When using RTCP Extended Reports (XR) [RFC3611] with RTCP reporting groups, it is RECOMMENDED that the reporting source is used to send the RTCP XR packets. If multiple reporting sources are in use, the reporting source that sends the SR/RR packets that relate to a particular remote SSRC SHOULD send the RTCP XR reports about that SSRC. This is motivated as one commonly combine the RTCP XR metrics with the regular report block to more fully understand the situation. Receiving these blocks in different compound packets reduces their value as the measuring intervals are not synchronized in those cases.

Some RTCP XR report blocks are specific to particular types of media, and might be relevant to only some members of a reporting group. For example, it would make no sense for an SSRC that is receiving video to send a VoIP metric RTCP XR report block. Such media specific RTCP XR report blocks MUST be sent by the SSRC to which they are relevant, and MUST NOT be included in the common report sent by the reporting source. This might mean that some SSRCs send RTCP XR packets in compound RTCP packets that contain an empty RTCP SR/RR packet, and that the time period covered by the RTCP XR packet is different to that covered by the RTCP SR/RR packet. If it is important that the RTCP XR packet and RTCP SR/RR packet cover the same time period, then that source SHOULD be removed from the RTCP reporting group, and send standard RTCP packets instead.

#### 3.5. Middlebox Considerations

Many different types of middlebox are used with RTP. RTCP reporting groups are potentially relevant to those types of RTP middlebox that have their own SSRCs and generate RTCP reports for the traffic they receive. RTP middleboxes that do not have their own SSRC, and that don't send RTCP reports on the traffic they receive, cannot use the RTCP reporting groups extension, since they generate no RTCP reports to group.

An RTP middlebox that has several SSRCs of its own can use the RTCP reporting groups extension to group the RTCP reports it generates. This can occur, for example, if a middlebox is acting as an RTP mixer for both audio and video flows that are multiplexed onto a single RTP session, where the middlebox has one SSRC for the audio mixer and one for the video mixer part, and when the middlebox wants to avoid cross reporting between audio and video.

A middlebox cannot use the RTCP reporting groups extension to group RTCP packets from the SSRCs that it is forwarding. It can, however, group the RTCP packets from the SSRCs it is forwarding into compound RTCP packets following the rules in Section 6.1 of [RFC3550] and Section 5.3 of [I-D.ietf-avtcore-rtp-multi-stream]. If the middlebox is using RTCP reporting groups for its own SSRCs, it MAY include RTCP packets from the SSRCs that it is forwarding as part of the compound RTCP packets its reporting source generates.

A middlebox that forwards RTCP SR or RR packets sent by members of a reporting group MUST forward the corresponding RTCP SDES RGRP items, as described in Section 3.2.1. A middlebox that forwards RTCP SR or RR packets sent by member of a reporting group MUST also forward the corresponding RTCP RGRS packets, as described in Section 3.2.2. Failure to forward these packets can cause compatibility problems, as described in Section 4.2.

If a middlebox rewrites SSRC values in the RTP and RTCP packets that it is forwarding, then it MUST make the corresponding changes in RTCP SDES packets containing RGRP items and in RTCP RGRS packets, to allow them to be associated with the rewritten SSRCs.

### 3.6. SDP Signalling for Reporting Groups

This document defines the "a=rtcp-rgrp" Session Description Protocol (SDP) [RFC4566] attribute to indicate if the session participant is capable of supporting RTCP Reporting Groups for applications that use SDP for configuration of RTP sessions. It is a property attribute, and hence takes no value. The multiplexing category [I-D.ietf-mmusic-sdp-mux-attributes] is IDENTICAL, as the functionality applies on RTP session level. A participant that proposes the use of RTCP Reporting Groups SHALL itself support the reception of RTCP Reporting Groups. The formal definition of this attribute is:

Name: rtcp-rgrp  
Value:  
Usage Level: session, media  
Charset Dependent: no  
Example:  
  a=rtcp-rgrp

When using SDP Offer/Answer [RFC3264], the following procedures are to be used:

- o Generating the initial SDP offer: If the offerer supports the RTCP reporting group extensions, and is willing to accept RTCP packets containing those extensions, then it MUST include an "a=rtcp-rgrp" attribute in the initial offer. If the offerer does not support RTCP reporting groups extensions, or is not willing to accept RTCP packets containing those extensions, then it MUST NOT include the "a=rtcp-rgrp" attribute in the offer.
- o Generating the SDP answer: If the SDP offer contains an "a=rtcp-rgrp" attribute, and if the answerer supports RTCP reporting groups and is willing to receive RTCP packets using the RTCP reporting groups extensions, then the answerer MAY include an "a=rtcp-rgrp" attribute in the answer and MAY send RTCP packets containing the RTCP reporting groups extensions. If the offer does not contain an "a=rtcp-rgrp" attribute, or if the offer does contain such an attribute but the answerer does not wish to accept RTCP packets using the RTCP reporting groups extensions, then the answerer MUST NOT include an "a=rtcp-rgrp" attribute.
- o Offerer Processing of the SDP Answer: If the SDP answer contains an "a=rtcp-rgrp" attribute, and the corresponding offer also contained an "a=rtcp-rgrp" attribute, then the offerer MUST be prepared to accept and process RTCP packets that contain the reporting groups extension, and MAY send RTCP packets that contain the reporting groups extension. If the SDP answer contains an "a=rtcp-rgrp" attribute, but the corresponding offer did not contain the "a=rtcp-rgrp" attribute, then the offerer MUST reject the call. If the SDP answer does not contain an "a=rtcp-rgrp" attribute, then the offerer MUST NOT send packets containing the RTCP reporting groups extensions, and does not need to process packet containing the RTCP reporting groups extensions.

In declarative usage of SDP, such as the Real Time Streaming Protocol (RTSP) [RFC2326] and the Session Announcement Protocol (SAP) [RFC2974], the presence of the attribute indicates that the session participant MAY use RTCP Reporting Groups in its RTCP transmissions. An implementation that doesn't explicitly support RTCP Reporting Groups MAY join a RTP session as long as it has been verified that

the implementation doesn't suffer from the problems discussed in Section 4.2.

#### 4. Properties of RTCP Reporting Groups

This section provides additional information on what the resulting properties are with the design specified in Section 3. The content of this section is non-normative.

##### 4.1. Bandwidth Benefits of RTCP Reporting Groups

To understand the benefits of RTCP reporting groups, consider a scenario in which the two endpoints in a session each have a hundred sources, of which eight each are sending within any given reporting interval.

For ease of analysis, we can make the simplifying approximation that the duration of the RTCP reporting interval is equal to the total size of the RTCP packets sent during an RTCP interval, divided by the RTCP bandwidth. (This will be approximately true in scenarios where the bandwidth is not so high that the minimum RTCP interval is reached.) For further simplification, we can assume RTCP senders are following the recommendations regarding Compound RTCP Packets in [I-D.ietf-avtcore-rtp-multi-stream]; thus, the per-packet transport-layer overhead will be small relative to the RTCP data. Thus, only the actual RTCP data itself need be considered.

In a report interval in this scenario, there will, as a baseline, be 200 SDES packets, 184 RR packets, and 16 SR packets. This amounts to approximately 6.5 kB of RTCP per report interval, assuming 16-byte CNAMEs and no other SDES information.

Using the original [RFC3550] everyone-reports-on-every-sender feedback rules, each of the 184 receivers will send 16 report blocks, and each of the 16 senders will send 15. This amounts to approximately 76 kB of report block traffic per interval; 92% of RTCP traffic consists of report blocks.

If reporting groups are used, however, there is only 0.4 kB of reports per interval, with no loss of useful information. Additionally, there will be (assuming 16-byte RGRPs, and a single reporting source per reporting group) an additional 2.4 kB per cycle of RGRP SDES items and RGRS packets. Put another way, the unmodified [RFC3550] reporting interval is approximately 9 times longer than if reporting groups are in use.

#### 4.2. Compatibility of RTCP Reporting Groups

The RTCP traffic generated by receivers using RTCP Reporting Groups might appear, to observers unaware of these semantics, to be generated by receivers who are experiencing a network disconnection, as the non-reporting sources appear not to be receiving a given sender at all.

This could be a potentially critical problem for such a sender using RTCP for congestion control, as such a sender might think that it is sending so much traffic that it is causing complete congestion collapse.

However, such an interpretation of the session statistics would require a fairly sophisticated RTCP analysis. Any receiver of RTCP statistics which is just interested in information about itself needs to be prepared that any given reception report might not contain information about a specific media source, because reception reports in large conferences can be round-robin.

Thus, it is unclear to what extent such backward compatibility issues would actually cause trouble in practice.

#### 5. Security Considerations

The security considerations of [RFC3550] and [I-D.ietf-avtcore-rtp-multi-stream] apply. If the RTP/AVPF profile is in use, then the security considerations of [RFC4585] (and [RFC5104], if used) also apply. If RTCP XR is used, the security consideration of [RFC3611] and any XR report blocks used also apply.

The RTCP SDES RGRP item is vulnerable to malicious modifications unless integrity protected is used. A modification of this item's length field cause the parsing of the RTCP packet in which it is contained to fail. Depending on the implementation, parsing of the full compound RTCP packet can also fail causing the whole packet to be discarded. A modification to the value of this SDES item would make the receiver of the report think that the sender of the report was a member of a different RTCP reporting group. This will potentially create an inconsistency, when the RGRS reports the source as being in the same reporting group as another source with another reporting group identifier. What impact on a receiver implementation such inconsistencies would have are difficult to fully predict. One case is when congestion control or other adaptation mechanisms are used, an inconsistent report can result in a media sender to reduce its bit-rate. However, a direct modification of the receiver report or a feedback message itself would be a more efficient attack, and equally costly to perform.

The new RGRS RTCP Packet type is very simple. The common RTCP packet type header shares the security risks with previous RTCP packet types. Errors or modification of the length field can cause the full compound packet to fail header validation (see Appendix A.2 in [RFC3550]) resulting in the whole compound RTCP packet being discarded. Modification of the SC or P fields would cause inconsistency when processing the RTCP packet, likely resulting it being classified as invalid. A modification of the PT field would cause the packet being interpreted under some other packet type's rules. In such case the result might be more or less predictable but packet type specific. Modification of the SSRC of packet sender would attribute this packet to another sender. Resulting in a receiver believing the reporting group applies also for this SSRC, if it exists. If it doesn't exist, unless also corresponding modifications are done on a SR/RR packet and a SDES packet the RTCP packet SHOULD be discarded. If consistent changes are done, that could be part of a resource exhaustion attack on a receiver implementation. Modification of the "List of SSRCs for the Reporting Source(s)" would change the SSRC the receiver expect to report on behalf of this SSRC. If that SSRC exist, that could potentially change the report group used for this SSRC. A change to another reporting group belonging to another endpoint is likely detectable as there would be a mismatch between the SSRC of the packet sender's endpoint information, transport addresses, SDES CNAME etc and the corresponding information from the reporting group indicated.

In general the reporting group is providing limited impacts attacks. The most significant result from an deliberate attack would be to cause the information to be discarded or be inconsistent, including discard of all RTCP packets that are modified. This causes a lack of information at any receiver entity, possibly disregarding the endpoints participation in the session.

To protect against this type of attacks from external non trusted entities, integrity and source authentication SHOULD be applied. This can be done, for example, by using SRTP [RFC3711] with appropriate key-management, other options exist as discussed in RTP Security Options [RFC7201].

The Report Group Identifier has a potential privacy impacting properties. If this would be generated by an implementation in such a way that is long term stable or predictable, it could be used for tracking a particular end-point. Therefore it is RECOMMENDED that it be generated as a short-term persistent RGRP, following the rules for short-term persistent CNAMEs in [RFC7022]. The rest of the information revealed, i.e. the SSRCs, the size of reporting group and the number of reporting sources in a reporting group is of less sensitive nature, considering that the SSRCs and the communication

would anyway be revealed without this extension. By encrypting the report group extensions the SSRC values would be preserved confidential, but can still be revealed if SRTP [RFC3711] is used. The size of the reporting groups and number of reporting sources are likely determinable from analysis of the packet pattern and sizes. However, this information appears to have limited value.

## 6. IANA Considerations

(Note to the RFC-Editor: in the following, please replace "TBA" with the IANA-assigned value, and "XXXX" with the number of this document, then delete this note)

The IANA is requested to register one new RTCP SDES item in the "RTCP SDES Item Types" registry, as follows:

Value	Abbrev	Name	Reference
TBA	RGRP	Reporting Group Identifier	[RFCXXXX]

The definition of the RTCP SDES RGRP item is given in Section 3.2.1 of this memo.

The IANA is also requested to register one new RTCP packet type in the "RTCP Control Packet Types (PT)" Registry as follows:

Value	Abbrev	Name	Reference
TBA	RGRS	Reporting Group Reporting Sources	[RFCXXXX]

The definition of the RTCP RGRS packet type is given in Section 3.2.2 of this memo.

The IANA is also requested to register one new SDP attribute:

```
SDP Attribute ("att-field"):
Attribute name:      rtcp-rgrp
Long form:          RTCP Reporting Groups
Type of name:       att-field
Type of attribute:  Media or session level
Subject to charset: No
Purpose:            Negotiate or configure the use of the RTCP
                   Reporting Group Extension.
Reference:          [RFCXXXX]
Values:             None
```

The definition of the "a=rtcp-rgrp" SDES attribute is given in Section 3.6 of this memo.

## 7. References

### 7.1. Normative References

- [I-D.ietf-avtcore-rtp-multi-stream]  
Lennox, J., Westerlund, M., Wu, Q., and C. Perkins,  
"Sending Multiple RTP Streams in a Single RTP Session",  
draft-ietf-avtcore-rtp-multi-stream-11 (work in progress),  
December 2015.
- [I-D.ietf-mmusic-sdp-mux-attributes]  
Nandakumar, S., "A Framework for SDP Attributes when  
Multiplexing", draft-ietf-mmusic-sdp-mux-attributes-12  
(work in progress), January 2016.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119,  
DOI 10.17487/RFC2119, March 1997,  
<<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model  
with Session Description Protocol (SDP)", RFC 3264,  
DOI 10.17487/RFC3264, June 2002,  
<<http://www.rfc-editor.org/info/rfc3264>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.  
Jacobson, "RTP: A Transport Protocol for Real-Time  
Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550,  
July 2003, <<http://www.rfc-editor.org/info/rfc3550>>.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session  
Description Protocol", RFC 4566, DOI 10.17487/RFC4566,  
July 2006, <<http://www.rfc-editor.org/info/rfc4566>>.
- [RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla,  
"Guidelines for Choosing RTP Control Protocol (RTCP)  
Canonical Names (CNAMEs)", RFC 7022, DOI 10.17487/RFC7022,  
September 2013, <<http://www.rfc-editor.org/info/rfc7022>>.

### 7.2. Informative References

- [RFC2326] Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time  
Streaming Protocol (RTSP)", RFC 2326,  
DOI 10.17487/RFC2326, April 1998,  
<<http://www.rfc-editor.org/info/rfc2326>>.

- [RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, DOI 10.17487/RFC2974, October 2000, <<http://www.rfc-editor.org/info/rfc2974>>.
- [RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <<http://www.rfc-editor.org/info/rfc3611>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<http://www.rfc-editor.org/info/rfc3711>>.
- [RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <<http://www.rfc-editor.org/info/rfc4585>>.
- [RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, DOI 10.17487/RFC4588, July 2006, <<http://www.rfc-editor.org/info/rfc4588>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<http://www.rfc-editor.org/info/rfc5104>>.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <<http://www.rfc-editor.org/info/rfc5506>>.
- [RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <<http://www.rfc-editor.org/info/rfc6190>>.
- [RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, DOI 10.17487/RFC7201, April 2014, <<http://www.rfc-editor.org/info/rfc7201>>.

Authors' Addresses

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

Email: [jonathan@vidyo.com](mailto:jonathan@vidyo.com)

Magnus Westerlund  
Ericsson  
Farogatan 2  
SE-164 80 Kista  
Sweden

Phone: +46 10 714 82 87  
Email: [magnus.westerlund@ericsson.com](mailto:magnus.westerlund@ericsson.com)

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

Email: [bill.wu@huawei.com](mailto:bill.wu@huawei.com)

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

Email: [csp@csperkins.org](mailto:csp@csperkins.org)

avtcore  
Internet-Draft  
Intended status: Standards Track  
Expires: January 12, 2021

S. Zhao  
S. Wenger  
Tencent  
Y. Sanchez  
Fraunhofer HHI  
July 11, 2020

RTP Payload Format for Versatile Video Coding (VVC)  
draft-ietf-avtcore-rtp-vvc-02

Abstract

This memo describes an RTP payload format for the video coding standard ITU-T Recommendation [H.266] and ISO/IEC International Standard [ISO23090-3], both also known as Versatile Video Coding (VVC) and developed by the Joint Video Experts Team (JVET). The RTP payload format allows for packetization of one or more Network Abstraction Layer (NAL) units in each RTP packet payload as well as fragmentation of a NAL unit into multiple RTP packets. The payload format has wide applicability in videoconferencing, Internet video streaming, and high-bitrate entertainment-quality video, among other applications.

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 <https://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 January 12, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1.	Introduction . . . . .	3
1.1.	Overview of the VVC Codec . . . . .	3
1.1.1.	Coding-Tool Features (informative) . . . . .	4
1.1.2.	Systems and Transport Interfaces . . . . .	6
1.1.3.	Parallel Processing Support (informative) . . . . .	10
1.1.4.	NAL Unit Header . . . . .	11
1.2.	Overview of the Payload Format . . . . .	12
2.	Conventions . . . . .	12
3.	Definitions and Abbreviations . . . . .	12
3.1.	Definitions . . . . .	12
3.1.1.	Definitions from the VVC Specification . . . . .	13
3.1.2.	Definitions Specific to This Memo . . . . .	16
3.2.	Abbreviations . . . . .	16
4.	RTP Payload Format . . . . .	17
4.1.	RTP Header Usage . . . . .	18
4.2.	Payload Header Usage . . . . .	19
4.3.	Payload Structures . . . . .	20
4.3.1.	Single NAL Unit Packets . . . . .	20
4.3.2.	Aggregation Packets (APs) . . . . .	21
4.3.3.	Fragmentation Units . . . . .	25
4.4.	Decoding Order Number . . . . .	28
5.	Packetization Rules . . . . .	29
6.	De-packetization Process . . . . .	30
7.	Payload Format Parameters . . . . .	32
7.1.	Media Type Registration . . . . .	32
7.2.	SDP Parameters . . . . .	32
7.2.1.	Mapping of Payload Type Parameters to SDP . . . . .	32
7.2.2.	Usage with SDP Offer/Answer Model . . . . .	33
8.	Use with Feedback Messages . . . . .	33
8.1.	Picture Loss Indication (PLI) . . . . .	33
8.2.	Slice Loss Indication (SLI) . . . . .	33
8.3.	Reference Picture Selection Indication (RPSI) . . . . .	33
8.4.	Full Intra Request (FIR) . . . . .	34
9.	Frame Marking . . . . .	34
9.1.	Frame Marking Short Extension . . . . .	35
9.2.	Frame Marking Long Extension . . . . .	36
10.	Security Considerations . . . . .	37
11.	Congestion Control . . . . .	38

12. IANA Considerations . . . . .	39
13. Acknowledgements . . . . .	39
14. References . . . . .	39
14.1. Normative References . . . . .	39
14.2. Informative References . . . . .	41
Appendix A. Change History . . . . .	42
Authors' Addresses . . . . .	43

## 1. Introduction

The Versatile Video Coding [VVC] specification, formally published as both ITU-T Recommendation H.266 and ISO/IEC International Standard 23090-3 [ISO23090-3], is currently in the ISO/IEC approval process and is planned for ratification in mid 2020. H.266 is reported to provide significant coding efficiency gains over H.265 and earlier video codec formats.

This memo describes an RTP payload format for VVC. It shares its basic design with the NAL (Network Abstraction Layer) unit-based RTP payload formats of, H.264 Video Coding [RFC6184], Scalable Video Coding (SVC) [RFC6190], High Efficiency Video Coding (HEVC) [RFC7798] and their respective predecessors. With respect to design philosophy, security, congestion control, and overall implementation complexity, it has similar properties to those earlier payload format specifications. This is a conscious choice, as at least RFC 6184 is widely deployed and generally known in the relevant implementer communities. Certain mechanisms known from [RFC6190] were incorporated in VVC, as VVC version 1 supports temporal, spatial, and signal-to-noise ratio (SNR) scalability.

### 1.1. Overview of the VVC Codec

[VVC] and [HEVC] share a similar hybrid video codec design. In this memo, we provide a very brief overview of those features of VVC that are, in some form, addressed by the payload format specified herein. Implementers have to read, understand, and apply the ITU- T/ISO/IEC specifications pertaining to [VVC] to arrive at interoperable, well-performing implementations.

Conceptually, both [VVC] and [HEVC] include a Video Coding Layer (VCL), which is often used to refer to the coding-tool features, and a NAL, which is often used to refer to the systems and transport interface aspects of the codecs.

### 1.1.1.1. Coding-Tool Features (informative)

Coding tool features are described below with occasional reference to the coding tool set of [HEVC], which is well known in the community.

Similar to earlier hybrid-video-coding-based standards, including HEVC, the following basic video coding design is employed by VVC. A prediction signal is first formed by either intra- or motion-compensated prediction, and the residual (the difference between the original and the prediction) is then coded. The gains in coding efficiency are achieved by redesigning and improving almost all parts of the codec over earlier designs. In addition, [VVC] includes several tools to make the implementation on parallel architectures easier.

Finally, [VVC] includes temporal, spatial, and SNR scalability as well as multiview coding support.

#### Coding blocks and transform structure

Among major coding-tool differences between HEVC and VVC, one of the important improvements is the more flexible coding tree structure in VVC, i.e., multi-type tree. In addition to quadtree, binary and ternary trees are also supported, which contributes significant improvement in coding efficiency. Moreover, the maximum size of Coding Tree Unit (CTU) is increased from 64x64 to 128x128. To improve the coding efficiency of chroma signal, luma chroma separated trees at CTU level may be employed for intra-slices. The square transforms in HEVC are extended to non-square transforms for rectangular blocks resulting from binary and ternary tree splits. Besides, [VVC] supports multiple transform sets (MTS), including DCT-2, DST-7, and DCT-8 as well as the non-separable secondary transform. The transforms used in [VVC] can have different sizes with support for larger transform sizes. For DCT-2, the transform sizes range from 2x2 to 64x64, and for DST-7 and DCT-8, the transform sizes range from 4x4 to 32x32. In addition, [VVC] also support sub-block transform for both intra and inter coded blocks. For intra coded blocks, intra sub-partitioning (ISP) may be used to allow sub-block based intra prediction and transform. For inter blocks, sub-block transform may be used assuming that only a part of an inter-block has non-zero transform coefficients.

#### Entropy coding

Similar to HEVC, [VVC] uses a single entropy-coding engine, which is based on Context Adaptive Binary Arithmetic Coding (CABAC) [CABAC], but with the support of multi-window sizes. The window sizes can be initialized differently for different context models. Due to such a

design, it has more efficient adaptation speed and better coding efficiency. A joint chroma residual coding scheme is applied to further exploit the correlation between the residuals of two color components. In VVC, different residual coding schemes are applied for regular transform coefficients and residual samples generated using transform-skip mode.

#### In-loop filtering

[VVC] has more feature support in loop filters than HEVC. The deblocking filter in [VVC] is similar to HEVC but operates at a smaller grid. After deblocking and sample adaptive offset (SAO), an adaptive loop filter (ALF) may be used. As a Wiener filter, ALF reduces distortion of decoded pictures. Besides, [VVC] introduces a new module before deblocking called luma mapping with chroma scaling to fully utilize the dynamic range of signal so that rate-distortion performance of both SDR and HDR content is improved.

#### Motion prediction and coding

Compared to HEVC, [VVC] introduces several improvements in this area. First, there is the Adaptive motion vector resolution (AMVR), which can save bit cost for motion vectors by adaptively signaling motion vector resolution. Then the Affine motion compensation is included to capture complicated motion like zooming and rotation. Meanwhile, prediction refinement with the optical flow with affine mode (PROF) is further deployed to mimic affine motion at the pixel level. Thirdly the decoder side motion vector refinement (DMVR) is a method to derive MV vector at decoder side based on block matching so that fewer bits may be spent on motion vectors. Bi-directional optical flow (BDOF) is a similar method to PROF. BDOF adds a sample wise offset at 4x4 sub-block level that is derived with equations based on gradients of the prediction samples and a motion difference relative to CU motion vectors. Furthermore, merge with motion vector difference (MMVD) is a special mode, which further signals a limited set of motion vector differences on top of merge mode. In addition to MMVD, there are another three types of special merge modes, i.e., sub-block merge, triangle, and combined intra-/inter- prediction (CIIP). Sub-block merge list includes one candidate of sub-block temporal motion vector prediction (SbTMVP) and up to four candidates of affine motion vectors. Triangle is based on triangular block motion compensation. CIIP combines intra- and inter- predictions with weighting. Adaptive weighting may be employed with a block-level tool called bi-prediction with CU based weighting (BCW) which provides more flexibility than in HEVC.

#### Intra prediction and intra-coding

To capture the diversified local image texture directions with finer granularity, [VVC] supports 65 angular directions instead of 33 directions in HEVC. The intra mode coding is based on a 6 most probable mode scheme, and the 6 most probable modes are derived using the neighboring intra prediction directions. In addition, to deal with the different distributions of intra prediction angles for different block aspect ratios, a wide-angle intra prediction (WAIP) scheme is applied in [VVC] by including intra prediction angles beyond those present in HEVC. Unlike HEVC which only allows using the most adjacent line of reference samples for intra prediction, [VVC] also allows using two further reference lines, as known as multi-reference-line (MRL) intra prediction. The additional reference lines can be only used for 6 most probable intra prediction modes. To capture the strong correlation between different colour components, in VVC, a cross-component linear mode (CCLM) is utilized which assumes a linear relationship between the luma sample values and their associated chroma samples. For intra prediction, [VVC] also applies a position-dependent prediction combination (PDPC) for refining the prediction samples closer to the intra prediction block boundary. Matrix-based intra prediction (MIP) modes are also used in [VVC] which generates an up to 8x8 intra prediction block using a weighted sum of downsampled neighboring reference samples, and the weightings are hardcoded constants.

Other coding-tool feature

[VVC] introduces dependent quantization (DQ) to reduce quantization error by state-based switching between two quantizers.

#### 1.1.2. Systems and Transport Interfaces

[VVC] inherits the basic systems and transport interfaces designs from HEVC and H.264. These include the NAL-unit-based syntax structure, the hierarchical syntax and data unit structure, the Supplemental Enhancement Information (SEI) message mechanism, and the video buffering model based on the Hypothetical Reference Decoder (HRD). The scalability features of [VVC] are conceptually similar to the scalable variant of HEVC known as SHVC. The hierarchical syntax and data unit structure consists of parameter sets at various levels (decoder, sequence (pertaining to all), sequence (pertaining to a single), picture), picture-level header parameters, slice-level header parameters, and lower-level parameters.

A number of key components that influenced the Network Abstraction Layer design of [VVC] as well as this memo are described below

Decoding Capability Information

The Decoding capability information includes parameters that stay constant for the lifetime of a Video Bitstream, which in IETF terms can translate to the lifetime of a session. Decoding capability informations can include profile, level, and sub-profile information to determine a maximum complexity interop point that is guaranteed to be never exceeded, even if splicing of video sequences occurs within a session. It further includes constraint flags, which can optionally be set to indicate that the video bitstream will be constraint in the use of certain features as indicated by the values of those flags. With this, a bitstream can be labelled as not using certain tools, which allows among other things for resource allocation in a decoder implementation.

#### Video parameter set

The Video Parameter Set (VPS) pertains to a Coded Video Sequences (CVS) of multiple layers covering the same range of picture units, and includes, among other information decoding dependency expressed as information for reference picture set construction of enhancement layers. The VPS provides a "big picture" of a scalable sequence, including what types of operation points are provided, the profile, tier, and level of the operation points, and some other high-level properties of the bitstream that can be used as the basis for session negotiation and content selection, etc. One VPS may be referenced by one or more Sequence parameter sets.

#### Sequence parameter set

The Sequence Parameter Set (SPS) contains syntax elements pertaining to a coded layer video sequence (CLVS), which is a group of pictures belonging to the same layer, starting with a random access point, and followed by pictures that may depend on each other and the random access point picture. In MPEG-2, the equivalent of a CVS was a Group of Pictures (GOP), which normally started with an I frame and was followed by P and B frames. While more complex in its options of random access points, VVC retains this basic concept. One remarkable difference of VVC is that a CLVS may start with a Gradual Decoding Refresh (GDR) picture, without requiring presence of traditional random access points in the bitstream, such as Instantaneous Decoding Refresh (IDR) or Clean Random Access (CRA) pictures. In many TV-like applications, a CVS contains a few hundred milliseconds to a few seconds of video. In video conferencing (without switching MCUs involved), a CVS can be as long in duration as the whole session.

#### Picture and Adaptation parameter set

The Picture Parameter Set and the Adaptation Parameter Set (PPS and APS, respectively) carry information pertaining to zero or more

pictures and zero or more slices, respectively. The PPS contains information that is likely to stay constant from picture to picture—at least for pictures for a certain type—whereas the APS contains information, such as adaptive loop filter coefficients, that are likely to change from picture to picture or even within a picture. A single APS can be referenced by slices of the same picture if that APS contains information about luma mapping with chroma scaling (LMCS) but different APS can be referenced by slices of the same picture if those APS contain information about ALF.

#### Picture Header

A Picture Header contains information that is common to all slices that belong to the same picture. Being able to send that information as a separate NAL unit when pictures are split into several slices allows for saving bitrate, compared to repeating the same information in all slices. However, there might be scenarios where low-bitrate video is transmitted using a single slice per picture. Having a separate NAL unit to convey that information incurs in an overhead for such scenarios. Therefore, VVC specifies signaling that indicates whether Picture Headers are present in the CLVS or not.

#### Profile, tier, and level

The profile, tier and level syntax structures in DCI, VPS and SPS contain profile, tier, level information for all layers that refer to the DCI, for layers associated with one or more output layer sets specified by the VPS, and for any layer that refers to the SPS, respectively.

#### Sub-Profiles

Within the [VVC] specification, a sub-profile is a 32-bit number coded according to ITU-T Rec. T.35, that does not carry a semantic. It is carried in the `profile_tier_level` structure and hence (potentially) present in the DCI, VPS, and SPS. External registration bodies can register a T.35 codepoint with ITU-T registration authorities and associate with their registration a description of bitstream complexity restrictions beyond the profiles defined by ITU-T and ISO/IEC. This would allow encoder manufacturers to label the bitstreams generated by their encoder as complying with such sub-profile. It is expected that upstream standardization organizations (such as: DVB and ATSC), as well as walled-garden video services will take advantage of this labelling system. In contrast to "normal" profiles, it is expected that sub-profiles may indicate encoder choices traditionally left open in the (decoder-centric) video coding specs, such as GOP structures, minimum/maximum QP values, and the mandatory use of certain tools or SEI messages.

### Constraint Flags

The `profile_tier_level` structure carries a considerable number of constraint flags, which an encoder can use to indicate to a decoder that it will not use a certain tool or technology. They were included in reaction to a perceived market need for labelling a bitstream as not exercising a certain tool that has become commercially unviable.

### Temporal scalability support

Editor notes: need will update along with VVC new draft in the future

[VVC] includes support of temporal scalability, by inclusion of the signaling of `TemporalId` in the NAL unit header, the restriction that pictures of a particular temporal sub-layer cannot be used for inter prediction reference by pictures of a lower temporal sub-layer, the sub-bitstream extraction process, and the requirement that each sub-bitstream extraction output be a conforming bitstream. Media-Aware Network Elements (MANEs) can utilize the `TemporalId` in the NAL unit header for stream adaptation purposes based on temporal scalability.

### Spatial, SNR, View Scalability

[VVC] includes support for spatial, SNR, and View scalability. Scalable video coding is widely considered to have technical benefits and enrich services for various video applications. Until recently, however, the functionality has not been included in the main profiles of video codecs and not wide deployed due to additional costs. In VVC, however, all those forms of scalability are supported natively through the signaling of the `layer_id` in the NAL unit header, the VPS which associates layers with given `layer_ids` to each other, reference picture selection, reference picture resampling for spatial scalability, and a number of other mechanisms not relevant for this memo. Scalability support can be implemented in a single decoding "loop" and is widely considered a comparatively lightweight operation.

### Spatial Scalability

With the existence of Reference Picture Resampling (RPR), in the "main" profile of VVC, the additional burden for scalability support is just a minor modification of the high-level syntax (HLS). In technical aspects, the inter-layer prediction is employed in a scalable system to improve the

coding efficiency of the enhancement layers. In addition to the spatial and temporal motion-compensated predictions that are available in a single-layer codec, the inter-layer prediction in [VVC] uses the resampled video data of the reconstructed reference picture from a reference layer to predict the current enhancement layer. Then, the resampling process for inter-layer prediction is performed at the block-level, without modifying the existing interpolation process for motion compensation compared to non-scalable RPR. It means that no additional resampling process is needed to support scalability.

#### SNR Scalability

SNR scalability is similar to Spatial Scalability except that the resampling factors are 1:1—in other words, there is no change in resolution, but there is inter-layer prediction.

#### SEI Messages

Supplementary Enhancement Information (SEI) messages are codepoints in the bitstream that do not influence the decoding process as specified in the [VVC] spec, but address issues of representation/rendering of the decoded bitstream, label the bitstream for certain applications, among other, similar tasks. The overall concept of SEI messages and many of the messages themselves has been inherited from the H.264 and HEVC specs. In the [VVC] environment, some of the SEI messages considered to be generally useful also in other video coding technologies have been moved out of the main specification into a companion document (TO DO: add reference once ITU designation is known).

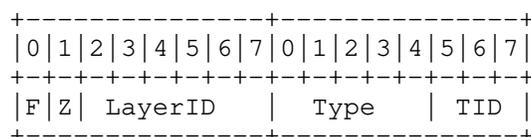
#### 1.1.3. Parallel Processing Support (informative)

Compared to HEVC, the [VVC] design to support parallelization offers numerous improvements. Some of those improvements are still undergoing changes in JVET. Information, to the extent relevant for this memo, will be added in future versions of this memo as the standardization in JVET progresses and the technology stabilizes.

Editor notes: update on sub-picture/slice/tile is needed following new VVC draft

## 1.1.4. NAL Unit Header

[VVC] maintains the NAL unit concept of HEVC with modifications. VVC uses a two-byte NAL unit header, as shown in Figure 1. The payload of a NAL unit refers to the NAL unit excluding the NAL unit header.



The Structure of the VVC NAL Unit Header.

Figure 1

The semantics of the fields in the NAL unit header are as specified in [VVC] and described briefly below for convenience. In addition to the name and size of each field, the corresponding syntax element name in [VVC] is also provided.

F: 1 bit

forbidden\_zero\_bit. Required to be zero in VVC. Note that the inclusion of this bit in the NAL unit header was to enable transport of [VVC] video over MPEG-2 transport systems (avoidance of start code emulations) [MPEG2S]. In the context of this memo the value 1 may be used to indicate a syntax violation, e.g., for a NAL unit resulted from aggregating a number of fragmented units of a NAL unit but missing the last fragment, as described in Section TBD.

Z: 1 bit

nuh\_reserved\_zero\_bit. Required to be zero in VVC, and reserved for future extensions by ITU-T and ISO/IEC. This memo does not overload the "Z" bit for local extensions, as a) overloading the "F" bit is sufficient and b) to preserve the usefulness of this memo to possible future versions of [VVC].

LayerId: 6 bits

nuh\_layer\_id. Identifies the layer a NAL unit belongs to, wherein a layer may be, e.g., a spatial scalable layer, a quality scalable layer .

Type: 5 bits

`nal_unit_type`. This field specifies the NAL unit type as defined in Table 7-1 of VVC. For a reference of all currently defined NAL unit types and their semantics, please refer to Section 7.4.2.2 in [VVC].

TID: 3 bits

`nuh_temporal_id_plus1`. This field specifies the temporal identifier of the NAL unit plus 1. The value of `TemporalId` is equal to TID minus 1. A TID value of 0 is illegal to ensure that there is at least one bit in the NAL unit header equal to 1, so to enable independent considerations of start code emulations in the NAL unit header and in the NAL unit payload data.

## 1.2. Overview of the Payload Format

This payload format defines the following processes required for transport of [VVC] coded data over RTP [RFC3550]:

- o Usage of RTP header with this payload format
- o Packetization of [VVC] coded NAL units into RTP packets using three types of payload structures: a single NAL unit packet, aggregation packet, and fragment unit
- o Transmission of [VVC] NAL units of the same bitstream within a single RTP stream.
- o Media type parameters to be used with the Session Description Protocol (SDP) [RFC4566]
- o Frame-marking mapping [FrameMarking]

## 2. Conventions

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 above.

## 3. Definitions and Abbreviations

### 3.1. Definitions

This document uses the terms and definitions of VVC. Section 3.1.1 lists relevant definitions from [VVC] for convenience. Section 3.1.2 provides definitions specific to this memo.

### 3.1.1. Definitions from the VVC Specification

Editor notes:

**Access unit (AU):** A set of PUs that belong to different layers and contain coded pictures associated with the same time for output from the DPB.

**Adaptation parameter set (APS):** A syntax structure containing syntax elements that apply to zero or more slices as determined by zero or more syntax elements found in slice headers.

**Bitstream:** A sequence of bits, in the form of a NAL unit stream or a byte stream, that forms the representation of a sequence of AUs forming one or more coded video sequences (CVSSs).

**Coded picture:** A coded representation of a picture comprising VCL NAL units with a particular value of `nuh_layer_id` within an AU and containing all CTUs of the picture.

**Clean random access (CRA) PU:** A PU in which the coded picture is a CRA picture.

**Clean random access (CRA) picture:** An IRAP picture for which each VCL NAL unit has `nal_unit_type` equal to `CRA_NUT`.

**Coded video sequence (CVS):** A sequence of AUs that consists, in decoding order, of a CVSS AU, followed by zero or more AUs that are not CVSS AUs, including all subsequent AUs up to but not including any subsequent AU that is a CVSS AU.

**Coded video sequence start (CVSS) AU:** An AU in which there is a PU for each layer in the CVS and the coded picture in each PU is a CLVSS picture.

**Coded layer video sequence (CLVS):** A sequence of PUs with the same value of `nuh_layer_id` that consists, in decoding order, of a CLVSS PU, followed by zero or more PUs that are not CLVSS PUs, including all subsequent PUs up to but not including any subsequent PU that is a CLVSS PU.

**Coded layer video sequence start (CLVSS) PU:** A PU in which the coded picture is a CLVSS picture.

**Coded layer video sequence start (CLVSS) picture:** A coded picture that is an IRAP picture with `NoOutputBeforeRecoveryFlag` equal to 1 or a GDR picture with `NoOutputBeforeRecoveryFlag` equal to 1.

**Coding tree unit (CTU):** A CTB of luma samples, two corresponding CTBs of chroma samples of a picture that has three sample arrays, or a CTB of samples of a monochrome picture or a picture that is coded using three separate colour planes and syntax structures used to code the samples.

**Decoding Capability Information (DCI):** A syntax structure containing syntax elements that apply to the entire bitstream.

**Decoded picture buffer (DPB):** A buffer holding decoded pictures for reference, output reordering, or output delay specified for the hypothetical reference decoder.

**Gradual decoding refresh (GDR) picture:** A picture for which each VCL NAL unit has `nal_unit_type` equal to `GDR_NUT`.

**Instantaneous decoding refresh (IDR) PU:** A PU in which the coded picture is an IDR picture.

**Instantaneous decoding refresh (IDR) picture:** An IRAP picture for which each VCL NAL unit has `nal_unit_type` equal to `IDR_W_RADL` or `IDR_N_LP`.

**Intra random access point (IRAP) AU:** An AU in which there is a PU for each layer in the CVS and the coded picture in each PU is an IRAP picture.

**Intra random access point (IRAP) PU:** A PU in which the coded picture is an IRAP picture.

**Intra random access point (IRAP) picture:** A coded picture for which all VCL NAL units have the same value of `nal_unit_type` in the range of `IDR_W_RADL` to `CRA_NUT`, inclusive.

**Layer:** A set of VCL NAL units that all have a particular value of `nuh_layer_id` and the associated non-VCL NAL units.

**Network abstraction layer (NAL) unit:** A syntax structure containing an indication of the type of data to follow and bytes containing that data in the form of an RBSP interspersed as necessary with emulation prevention bytes.

**Network abstraction layer (NAL) unit stream:** A sequence of NAL units.

**Operation point (OP):** A temporal subset of an OLS, identified by an OLS index and a highest value of `TemporalId`.

**Picture parameter set (PPS):** A syntax structure containing syntax elements that apply to zero or more entire coded pictures as determined by a syntax element found in each slice header.

**Picture unit (PU):** A set of NAL units that are associated with each other according to a specified classification rule, are consecutive in decoding order, and contain exactly one coded picture.

**Random access:** The act of starting the decoding process for a bitstream at a point other than the beginning of the stream.

**Sequence parameter set (SPS):** A syntax structure containing syntax elements that apply to zero or more entire CLVSS as determined by the content of a syntax element found in the PPS referred to by a syntax element found in each picture header.

**Slice:** An integer number of complete tiles or an integer number of consecutive complete CTU rows within a tile of a picture that are exclusively contained in a single NAL unit.

**Sub-layer:** A temporal scalable layer of a temporal scalable bitstream consisting of VCL NAL units with a particular value of the TemporalId variable, and the associated non-VCL NAL units.

**Subpicture:** An rectangular region of one or more slices within a picture.

**Sub-layer representation:** A subset of the bitstream consisting of NAL units of a particular sub-layer and the lower sub-layers.

**Tile:** A rectangular region of CTUs within a particular tile column and a particular tile row in a picture.

**Tile column:** A rectangular region of CTUs having a height equal to the height of the picture and a width specified by syntax elements in the picture parameter set.

**Tile row:** A rectangular region of CTUs having a height specified by syntax elements in the picture parameter set and a width equal to the width of the picture.

**Video coding layer (VCL) NAL unit:** A collective term for coded slice NAL units and the subset of NAL units that have reserved values of nal\_unit\_type that are classified as VCL NAL units in this Specification.

### 3.1.2. Definitions Specific to This Memo

**Media-Aware Network Element (MANE):** A network element, such as a middlebox, selective forwarding unit, or application-layer gateway that is capable of parsing certain aspects of the RTP payload headers or the RTP payload and reacting to their contents.

**Editor Notes:** the following informative needs to be updated along with frame marking update

**Informative note:** The concept of a MANE goes beyond normal routers or gateways in that a MANE has to be aware of the signaling (e.g., to learn about the payload type mappings of the media streams), and in that it has to be trusted when working with Secure RTP (SRTP). The advantage of using MANEs is that they allow packets to be dropped according to the needs of the media coding. For example, if a MANE has to drop packets due to congestion on a certain link, it can identify and remove those packets whose elimination produces the least adverse effect on the user experience. After dropping packets, MANEs must rewrite RTCP packets to match the changes to the RTP stream, as specified in Section 7 of [RFC3550].

**NAL unit decoding order:** A NAL unit order that conforms to the constraints on NAL unit order given in Section 7.4.2.4 in [VVC], follow the Order of NAL units in the bitstream.

**NAL unit output order:** A NAL unit order in which NAL units of different access units are in the output order of the decoded pictures corresponding to the access units, as specified in [VVC], and in which NAL units within an access unit are in their decoding order.

**RTP stream:** See [RFC7656]. Within the scope of this memo, one RTP stream is utilized to transport one or more temporal sub-layers.

**Transmission order:** The order of packets in ascending RTP sequence number order (in modulo arithmetic). Within an aggregation packet, the NAL unit transmission order is the same as the order of appearance of NAL units in the packet.

### 3.2. Abbreviations

AU	Access Unit
AP	Aggregation Packet
CTU	Coding Tree Unit

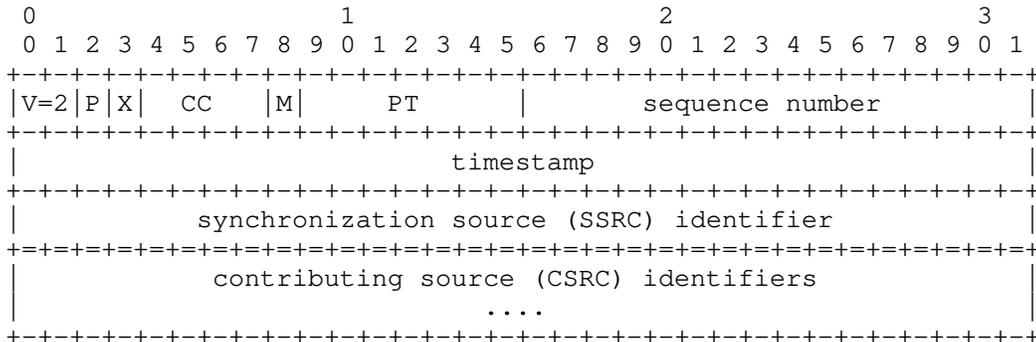
CVS	Coded Video Sequence
DPB	Decoded Picture Buffer
DCI	Decoding capability information
DON	Decoding Order Number
FIR	Full Intra Request
FU	Fragmentation Unit
HRD	Hypothetical Reference Decoder
IDR	Instantaneous Decoding Refresh
MANE	Media-Aware Network Element
MTU	Maximum Transfer Unit
NAL	Network Abstraction Layer
NALU	Network Abstraction Layer Unit
PLI	Picture Loss Indication
PPS	Picture Parameter Set
RPS	Reference Picture Set
RPSI	Reference Picture Selection Indication
SEI	Supplemental Enhancement Information
SLI	Slice Loss Indication
SPS	Sequence Parameter Set
VCL	Video Coding Layer
VPS	Video Parameter Set

#### 4. RTP Payload Format

4.1. RTP Header Usage

The format of the RTP header is specified in [RFC3550] (reprinted as Figure 2 for convenience). This payload format uses the fields of the header in a manner consistent with that specification.

The RTP payload (and the settings for some RTP header bits) for aggregation packets and fragmentation units are specified in Section 4.3.2 and Section 4.3.3, respectively.



RTP Header According to {{RFC3550}}

Figure 2

The RTP header information to be set according to this RTP payload format is set as follows:

Marker bit (M): 1 bit

Set for the last packet of the access unit, carried in the current RTP stream. This is in line with the normal use of the M bit in video formats to allow an efficient playout buffer handling.

Editor notes: The informative note below needs updating once the NAL unit type table is stable in the [VVC] spec.

Informative note: The content of a NAL unit does not tell whether or not the NAL unit is the last NAL unit, in decoding order, of an access unit. An RTP sender implementation may obtain this information from the video encoder. If, however, the implementation cannot obtain this information directly from

the encoder, e.g., when the bitstream was pre-encoded, and also there is no timestamp allocated for each NAL unit, then the sender implementation can inspect subsequent NAL units in decoding order to determine whether or not the NAL unit is the last NAL unit of an access unit as follows. A NAL unit is determined to be the last NAL unit of an access unit if it is the last NAL unit of the bitstream. A NAL unit `nal_uX` is also determined to be the last NAL unit of an access unit if both the following conditions are true: 1) the next VCL NAL unit `nal_uY` in decoding order has the high-order bit of the first byte after its NAL unit header equal to 1 or `nal_unit_type` equal to 19, and 2) all NAL units between `nal_uX` and `nal_uY`, when present, have `nal_unit_type` in the range of 13 to 17, inclusive, equal to 20, equal to 23 or equal to 26.

Payload Type (PT): 7 bits

The assignment of an RTP payload type for this new packet format is outside the scope of this document and will not be specified here. The assignment of a payload type has to be performed either through the profile used or in a dynamic way.

Sequence Number (SN): 16 bits

Set and used in accordance with [RFC3550].

Timestamp: 32 bits

The RTP timestamp is set to the sampling timestamp of the content. A 90 kHz clock rate MUST be used. If the NAL unit has no timing properties of its own (e.g., parameter set and SEI NAL units), the RTP timestamp MUST be set to the RTP timestamp of the coded picture of the access unit in which the NAL unit (according to Annex D of VVC) is included. Receivers MUST use the RTP timestamp for the display process, even when the bitstream contains picture timing SEI messages or decoding unit information SEI messages as specified in VVC.

Synchronization source (SSRC): 32 bits

Used to identify the source of the RTP packets. A single SSRC is used for all parts of a single bitstream.

#### 4.2. Payload Header Usage

The first two bytes of the payload of an RTP packet are referred to as the payload header. The payload header consists of the same

fields (F, Z, LayerId, Type, and TID) as the NAL unit header as shown in Section 1.1.4, irrespective of the type of the payload structure.

The TID value indicates (among other things) the relative importance of an RTP packet, for example, because NAL units belonging to higher temporal sub-layers are not used for the decoding of lower temporal sub-layers. A lower value of TID indicates a higher importance. More-important NAL units MAY be better protected against transmission losses than less-important NAL units.

For Discussion: quite possibly something similar can be said for the Layer\_id in layered coding, but perhaps not in multiview coding. (The relevant part of the spec is relatively new, therefore the soft language). However, for serious layer pruning, interpretation of the VPS is required. We can add language about the need for stateful interpretation of LayerID vis-a-vis stateless interpretation of TID later.

#### 4.3. Payload Structures

Three different types of RTP packet payload structures are specified. A receiver can identify the type of an RTP packet payload through the Type field in the payload header.

The three different payload structures are as follows:

- o Single NAL unit packet: Contains a single NAL unit in the payload, and the NAL unit header of the NAL unit also serves as the payload header. This payload structure is specified in Section 4.4.1.
- o Aggregation Packet (AP): Contains more than one NAL unit within one access unit. This payload structure is specified in Section 4.3.2.
- o Fragmentation Unit (FU): Contains a subset of a single NAL unit. This payload structure is specified in Section 4.3.3.

##### 4.3.1. Single NAL Unit Packets

Editor notes: its better to add a section to describe DONL and sprop-max\_don\_diff. sprop-max\_don\_diff is used but not specified as parameters in section 7 are not yet specified. A value of sprop-max\_don\_diff greater than 0 indicates that the transmission order may not correspond to the decoding order and that the DON is included in the payload header.

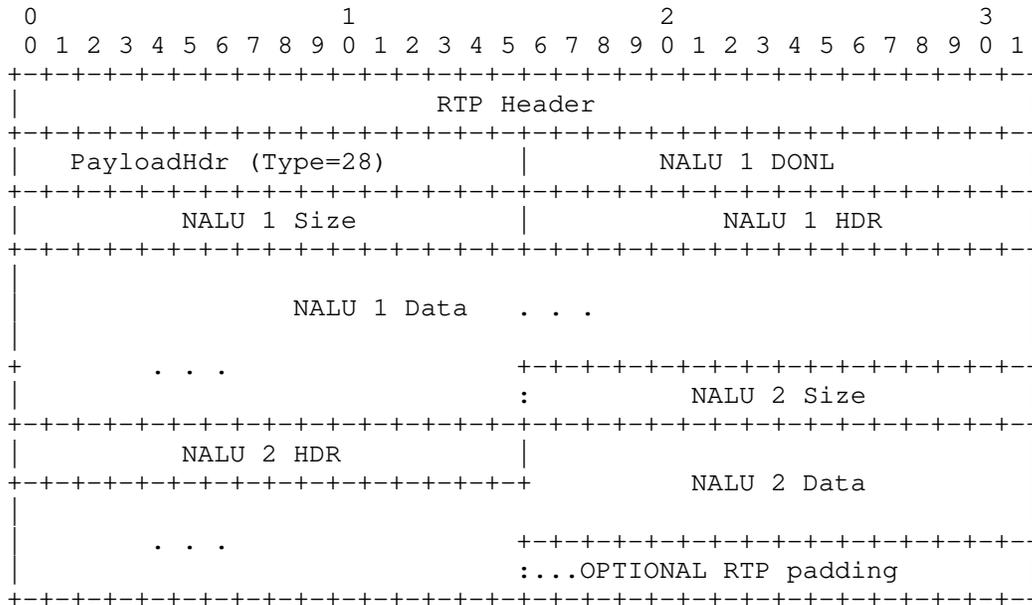
A single NAL unit packet contains exactly one NAL unit, and consists of a payload header (denoted as PayloadHdr), a conditional 16-bit











An Example of an AP Containing  
Two Aggregation Units with the DONL Field

Figure 8

4.3.3. Fragmentation Units

Fragmentation Units (FUs) are introduced to enable fragmenting a single NAL unit into multiple RTP packets, possibly without cooperation or knowledge of the [VVC] encoder. A fragment of a NAL unit consists of an integer number of consecutive octets of that NAL unit. Fragments of the same NAL unit MUST be sent in consecutive order with ascending RTP sequence numbers (with no other RTP packets within the same RTP stream being sent between the first and last fragment).

When a NAL unit is fragmented and conveyed within FUs, it is referred to as a fragmented NAL unit. APs MUST NOT be fragmented. FUs MUST NOT be nested; i.e., an FU can not contain a subset of another FU.

The RTP timestamp of an RTP packet carrying an FU is set to the NALU-time of the fragmented NAL unit.

An FU consists of a payload header (denoted as PayloadHdr), an FU header of one octet, a conditional 16-bit DONL field (in network byte order), and an FU payload, as shown in Figure 9).



When set to 1, the E bit indicates the end of a fragmented NAL unit, i.e., the last byte of the payload is also the last byte of the fragmented NAL unit. When the FU payload is not the last fragment of a fragmented NAL unit, the E bit MUST be set to 0.

Reserved: 1 bit

Placeholder

FuType: 5 bits

The field FuType MUST be equal to the field Type of the fragmented NAL unit.

The DONL field, when present, specifies the value of the 16 least significant bits of the decoding order number of the fragmented NAL unit.

If sprop-max-don-diff is greater than 0, and the S bit is equal to 1, the DONL field MUST be present in the FU, and the variable DON for the fragmented NAL unit is derived as equal to the value of the DONL field. Otherwise (sprop-max-don-diff is equal to 0, or the S bit is equal to 0), the DONL field MUST NOT be present in the FU.

A non-fragmented NAL unit MUST NOT be transmitted in one FU; i.e., the Start bit and End bit must not both be set to 1 in the same FU header.

The FU payload consists of fragments of the payload of the fragmented NAL unit so that if the FU payloads of consecutive FUs, starting with an FU with the S bit equal to 1 and ending with an FU with the E bit equal to 1, are sequentially concatenated, the payload of the fragmented NAL unit can be reconstructed. The NAL unit header of the fragmented NAL unit is not included as such in the FU payload, but rather the information of the NAL unit header of the fragmented NAL unit is conveyed in F, LayerId, and TID fields of the FU payload headers of the FUs and the FuType field of the FU header of the FUs. An FU payload MUST NOT be empty.

If an FU is lost, the receiver SHOULD discard all following fragmentation units in transmission order corresponding to the same fragmented NAL unit, unless the decoder in the receiver is known to be prepared to gracefully handle incomplete NAL units.

A receiver in an endpoint or in a MANE MAY aggregate the first n-1 fragments of a NAL unit to an (incomplete) NAL unit, even if fragment n of that NAL unit is not received. In this case, the

forbidden\_zero\_bit of the NAL unit MUST be set to 1 to indicate a syntax violation.

#### 4.4. Decoding Order Number

For each NAL unit, the variable AbsDon is derived, representing the decoding order number that is indicative of the NAL unit decoding order.

Let NAL unit  $n$  be the  $n$ -th NAL unit in transmission order within an RTP stream.

If sprop-max-don-diff is equal to 0, AbsDon[ $n$ ], the value of AbsDon for NAL unit  $n$ , is derived as equal to  $n$ .

Otherwise (sprop-max-don-diff is greater than 0), AbsDon[ $n$ ] is derived as follows, where DON[ $n$ ] is the value of the variable DON for NAL unit  $n$ :

- o If  $n$  is equal to 0 (i.e., NAL unit  $n$  is the very first NAL unit in transmission order), AbsDon[0] is set equal to DON[0].
- o Otherwise ( $n$  is greater than 0), the following applies for derivation of AbsDon[ $n$ ]:

If DON[ $n$ ] == DON[ $n-1$ ],  
AbsDon[ $n$ ] = AbsDon[ $n-1$ ]

If (DON[ $n$ ] > DON[ $n-1$ ] and DON[ $n$ ] - DON[ $n-1$ ] < 32768),  
AbsDon[ $n$ ] = AbsDon[ $n-1$ ] + DON[ $n$ ] - DON[ $n-1$ ]

If (DON[ $n$ ] < DON[ $n-1$ ] and DON[ $n-1$ ] - DON[ $n$ ] >= 32768),  
AbsDon[ $n$ ] = AbsDon[ $n-1$ ] + 65536 - DON[ $n-1$ ] + DON[ $n$ ]

If (DON[ $n$ ] > DON[ $n-1$ ] and DON[ $n$ ] - DON[ $n-1$ ] >= 32768),  
AbsDon[ $n$ ] = AbsDon[ $n-1$ ] - (DON[ $n-1$ ] + 65536 -  
DON[ $n$ ])

If (DON[ $n$ ] < DON[ $n-1$ ] and DON[ $n-1$ ] - DON[ $n$ ] < 32768),  
AbsDon[ $n$ ] = AbsDon[ $n-1$ ] - (DON[ $n-1$ ] - DON[ $n$ ])

For any two NAL units  $m$  and  $n$ , the following applies:

- o AbsDon[ $n$ ] greater than AbsDon[ $m$ ] indicates that NAL unit  $n$  follows NAL unit  $m$  in NAL unit decoding order.
- o When AbsDon[ $n$ ] is equal to AbsDon[ $m$ ], the NAL unit decoding order of the two NAL units can be in either order.

- o AbsDon[n] less than AbsDon[m] indicates that NAL unit n precedes NAL unit m in decoding order.

Informative note: When two consecutive NAL units in the NAL unit decoding order have different values of AbsDon, the absolute difference between the two AbsDon values may be greater than or equal to 1.

Informative note: There are multiple reasons to allow for the absolute difference of the values of AbsDon for two consecutive NAL units in the NAL unit decoding order to be greater than one. An increment by one is not required, as at the time of associating values of AbsDon to NAL units, it may not be known whether all NAL units are to be delivered to the receiver. For example, a gateway might not forward VCL NAL units of higher sub-layers or some SEI NAL units when there is congestion in the network. In another example, the first intra-coded picture of a pre-encoded clip is transmitted in advance to ensure that it is readily available in the receiver, and when transmitting the first intra-coded picture, the originator does not exactly know how many NAL units will be encoded before the first intra-coded picture of the pre-encoded clip follows in decoding order. Thus, the values of AbsDon for the NAL units of the first intra-coded picture of the pre-encoded clip have to be estimated when they are transmitted, and gaps in values of AbsDon may occur.

## 5. Packetization Rules

The following packetization rules apply:

- o If sprop-max-don-diff is greater than 0, the transmission order of NAL units carried in the RTP stream MAY be different than the NAL unit decoding order and the NAL unit output order.
- o A NAL unit of a small size SHOULD be encapsulated in an aggregation packet together one or more other NAL units in order to avoid the unnecessary packetization overhead for small NAL units. For example, non-VCL NAL units such as access unit delimiters, parameter sets, or SEI NAL units are typically small and can often be aggregated with VCL NAL units without violating MTU size constraints.
- o Each non-VCL NAL unit SHOULD, when possible from an MTU size match viewpoint, be encapsulated in an aggregation packet together with its associated VCL NAL unit, as typically a non-VCL NAL unit would be meaningless without the associated VCL NAL unit being available.

- o For carrying exactly one NAL unit in an RTP packet, a single NAL unit packet MUST be used.

## 6. De-packetization Process

The general concept behind de-packetization is to get the NAL units out of the RTP packets in an RTP stream and pass them to the decoder in the NAL unit decoding order.

The de-packetization process is implementation dependent. Therefore, the following description should be seen as an example of a suitable implementation. Other schemes may be used as well, as long as the output for the same input is the same as the process described below. The output is the same when the set of output NAL units and their order are both identical. Optimizations relative to the described algorithms are possible.

All normal RTP mechanisms related to buffer management apply. In particular, duplicated or outdated RTP packets (as indicated by the RTP sequences number and the RTP timestamp) are removed. To determine the exact time for decoding, factors such as a possible intentional delay to allow for proper inter-stream synchronization MUST be factored in.

NAL units with NAL unit type values in the range of 0 to 27, inclusive, may be passed to the decoder. NAL-unit-like structures with NAL unit type values in the range of 28 to 31, inclusive, MUST NOT be passed to the decoder.

The receiver includes a receiver buffer, which is used to compensate for transmission delay jitter within individual RTP streams and across RTP streams, to reorder NAL units from transmission order to the NAL unit decoding order. In this section, the receiver operation is described under the assumption that there is no transmission delay jitter within an RTP stream and across RTP streams. To make a difference from a practical receiver buffer that is also used for compensation of transmission delay jitter, the receiver buffer is hereafter called the de-packetization buffer in this section. Receivers should also prepare for transmission delay jitter; that is, either reserve separate buffers for transmission delay jitter buffering and de-packetization buffering or use a receiver buffer for both transmission delay jitter and de-packetization. Moreover, receivers should take transmission delay jitter into account in the buffering operation, e.g., by additional initial buffering before starting of decoding and playback.

When `sprop-max-don-diff` is equal to 0, the de-packetization buffer size is zero bytes, and the process described in the remainder of this paragraph applies.

The NAL units carried in the single RTP stream are directly passed to the decoder in their transmission order, which is identical to their decoding order. When there are several NAL units of the same RTP stream with the same NTP timestamp, the order to pass them to the decoder is their transmission order.

Informative note: The mapping between RTP and NTP timestamps is conveyed in RTCP SR packets. In addition, the mechanisms for faster media timestamp synchronization discussed in [RFC6051] may be used to speed up the acquisition of the RTP-to-wall-clock mapping.

When `sprop-max-don-diff` is greater than 0, the process described in the remainder of this section applies.

There are two buffering states in the receiver: initial buffering and buffering while playing. Initial buffering starts when the reception is initialized. After initial buffering, decoding and playback are started, and the buffering-while-playing mode is used.

Regardless of the buffering state, the receiver stores incoming NAL units, in reception order, into the de-packetization buffer. NAL units carried in RTP packets are stored in the de-packetization buffer individually, and the value of `AbsDon` is calculated and stored for each NAL unit.

Initial buffering lasts until condition A (the difference between the greatest and smallest `AbsDon` values of the NAL units in the de-packetization buffer is greater than or equal to the value of `sprop-max-don-diff`) or condition B (the number of NAL units in the de-packetization buffer is greater than the value of `sprop-depack-buf-nalus`) is true.

After initial buffering, whenever condition A or condition B is true, the following operation is repeatedly applied until both condition A and condition B become false:

- o The NAL unit in the de-packetization buffer with the smallest value of `AbsDon` is removed from the de-packetization buffer and passed to the decoder.

When no more NAL units are flowing into the de-packetization buffer, all NAL units remaining in the de-packetization buffer are removed from the buffer and passed to the decoder in the order of increasing `AbsDon` values.

## 7. Payload Format Parameters

This section specifies the optional parameters. A mapping of the parameters with Session Description Protocol (SDP) [RFC4556] is also provided for applications that use SDP.

### 7.1. Media Type Registration

The receiver MUST ignore any parameter unspecified in this memo.

Type name: Video

Subtype name: H266

Required parameters: none

Optional parameters:

Editor's notes: To be added

### 7.2. SDP Parameters

The receiver MUST ignore any parameter unspecified in this memo.

#### 7.2.1. Mapping of Payload Type Parameters to SDP

The media type video/H266 string is mapped to fields in the Session Description Protocol (SDP) [RFC4566] as follows:

- o The media name in the "m=" line of SDP MUST be video.
- o The encoding name in the "a=rtpmap" line of SDP MUST be H266 (the media subtype).
- o The clock rate in the "a=rtpmap" line MUST be 90000.
- o OPTIONAL PARAMETERS:

Editor's notes: To be dicussed here

##### 7.2.1.1. SDP Example

An example of media representation in SDP is as follows:

```
m=video 49170 RTP/AVP 98
a=rtpmap:98 H266/90000
a=fmtp:98 profile-id=1; sprop-vps=<video parameter sets data>
```

### 7.2.2. Usage with SDP Offer/Answer Model

When [VVC] is offered over RTP using SDP in an offer/answer model [RFC3264] for negotiation for unicast usage, the following limitations and rules apply:

Placeholder: To add limitations and considerations.

## 8. Use with Feedback Messages

The following subsections define the use of the Picture Loss Indication (PLI), Slice Loss Indication (SLI), Reference Picture Selection Indication (RPSI), and Full Intra Request (FIR) feedback messages with HEVC. The PLI, SLI, and RPSI messages are defined in [RFC4585], and the FIR message is defined in [RFC5104].

### 8.1. Picture Loss Indication (PLI)

As specified in RFC 4585, Section 6.3.1, the reception of a PLI by a media sender indicates "the loss of an undefined amount of coded video data belonging to one or more pictures". Without having any specific knowledge of the setup of the bitstream (such as use and location of in-band parameter sets, non-IRAP decoder refresh points, picture structures, and so forth), a reaction to the reception of an PLI by a [VVC] sender SHOULD be to send an IRAP picture and relevant parameter sets; potentially with sufficient redundancy so to ensure correct reception. However, sometimes information about the bitstream structure is known. For example, state could have been established outside of the mechanisms defined in this document that parameter sets are conveyed out of band only, and stay static for the duration of the session. In that case, it is obviously unnecessary to send them in-band as a result of the reception of a PLI. Other examples could be devised based on a priori knowledge of different aspects of the bitstream structure. In all cases, the timing and congestion control mechanisms of RFC 4585 MUST be observed.

### 8.2. Slice Loss Indication (SLI)

For further study. Maybe remove as there are no known implementations of SDLI in [HEVC] based systems

### 8.3. Reference Picture Selection Indication (RPSI)

Feedback-based reference picture selection has been shown as a powerful tool to stop temporal error propagation for improved error resilience [Girod99] [Wang05]. In one approach, the decoder side tracks errors in the decoded pictures and informs the encoder side that a particular picture that has been decoded relatively earlier is

correct and still present in the decoded picture buffer; it requests the encoder to use that correct picture-availability information when encoding the next picture, so to stop further temporal error propagation. For this approach, the decoder side should use the RPSI feedback message.

Encoders can encode some long-term reference pictures as specified in [VVC] for purposes described in the previous paragraph without the need of a huge decoded picture buffer. As shown in [Wang05], with a flexible reference picture management scheme, as in VVC, even a decoded picture buffer size of two picture storage buffers would work for the approach described in the previous paragraph.

The text above is copy-paste from RFC 7798. If we keep the RPSI message, it needs adaptation to the [VVC] syntax. Doing so shouldn't be too hard as the [VVC] reference picture mechanism is not too different from the [HEVC] one.

#### 8.4. Full Intra Request (FIR)

The purpose of the FIR message is to force an encoder to send an independent decoder refresh point as soon as possible, while observing applicable congestion-control-related constraints, such as those set out in [RFC8082]).

Upon reception of a FIR, a sender MUST send an IDR picture. Parameter sets MUST also be sent, except when there is a priori knowledge that the parameter sets have been correctly established. A typical example for that is an understanding between sender and receiver, established by means outside this document, that parameter sets are exclusively sent out-of-band.

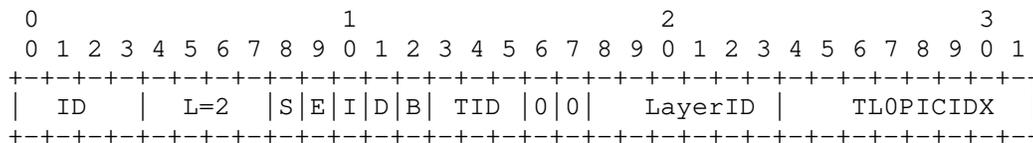
#### 9. Frame Marking

[FrameMarking] provides an extension mechanism for RTP. The codec-agnostic meta-data in the [FrameMarking] header provides valuable video frame information. Its usage with [VVC] is defined in this section. Refer [FrameMarking] for any unspecified fields. Two header extensions are RECOMMENDED:

- o The short extension for non-scalable streams.
- o The long extension for scalable streams.



9.2. Frame Marking Long Extension



Long Frame Marking RTP Extension for [VVC]

Figure 12

The fields for the long extension for scalable streams, as shown in Figure 12, are used as described in the following.

The LayerID (6 bits) and TID (3 bits) from the NAL unit header Section 1.1.4 are mapped to the generic LID and TID fields in [FrameMarking] as shown in Figure 12.

The I bit MUST be 1 when the NAL unit type is 7-9 (inclusive), otherwise it MUST be 0.

The D bit MUST be 1 when the syntax element `ph_non_ref_pic_flag` for a picture is equal to 1, otherwise it MUST be 0.

The S bit MUST be set to 1 if any of the following conditions is true and MUST be set to 0 otherwise:

- o The RTP packet is a single NAL unit packet and it is the first VCL NAL unit, in decoding order, of a picture.
- o The RTP packet is an AP, and the NAL unit in the first contained aggregation unit is the first VCL NAL unit, in decoding order, of a picture.
- o The RTP packet is a FU with its S bit equal to 1 and the FU payload contains a fragment of the first VCL NAL unit, in decoding order, of a picture.

The E bit MUST be set to 1 if any of the following conditions is true and MUST be set to 0 otherwise:

- o The RTP packet is a single NAL unit packet and it is the last VCL NAL unit, in decoding order, of a picture.
- o The RTP packet is an AP and the NAL unit in the last contained aggregation unit is the last VCL NAL unit, in decoding order, of a picture.

- o The RTP packet is a FU with its E bit equal to 1 and the FU payload contains a fragment of the last VCL NAL unit, in decoding order, of a picture.

## 10. Security Considerations

The scope of this Security Considerations section is limited to the payload format itself and to one feature of [VVC] that may pose a particularly serious security risk if implemented naively. The payload format, in isolation, does not form a complete system. Implementers are advised to read and understand relevant security-related documents, especially those pertaining to RTP (see the Security Considerations section in [RFC3550] ), and the security of the call-control stack chosen (that may make use of the media type registration of this memo). Implementers should also consider known security vulnerabilities of video coding and decoding implementations in general and avoid those.

Within this RTP payload format, and with the exception of the user data SEI message as described below, no security threats other than those common to RTP payload formats are known. In other words, neither the various media-plane-based mechanisms, nor the signaling part of this memo, seems to pose a security risk beyond those common to all RTP-based systems.

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550] , and in any applicable RTP profile such as RTP/AVP [RFC3551] , RTP/AVPF [RFC4585] , RTP/SAVP [RFC3711] , or RTP/SAVPF [RFC5124] . However, as "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format's responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201] . The rest of this section discusses the security impacting properties of the payload format itself.

Because the data compression used with this payload format is applied end-to-end, any encryption needs to be performed after compression. A potential denial-of-service threat exists for data encodings using compression techniques that have non-uniform receiver-end computational load. The attacker can inject pathological datagrams into the bitstream that are complex to decode and that cause the receiver to be overloaded. [VVC] is particularly vulnerable to such attacks, as it is extremely simple to generate datagrams containing

NAL units that affect the decoding process of many future NAL units. Therefore, the usage of data origin authentication and data integrity protection of at least the RTP packet is RECOMMENDED, for example, with SRTP [RFC3711] .

Like HEVC [RFC7798], [VVC] includes a user data Supplemental Enhancement Information (SEI) message. This SEI message allows inclusion of an arbitrary bitstring into the video bitstream. Such a bitstring could include JavaScript, machine code, and other active content. [VVC] leaves the handling of this SEI message to the receiving system. In order to avoid harmful side effects the user data SEI message, decoder implementations cannot naively trust its content. For example, it would be a bad and insecure implementation practice to forward any JavaScript a decoder implementation detects to a web browser. The safest way to deal with user data SEI messages is to simply discard them, but that can have negative side effects on the quality of experience by the user.

End-to-end security with authentication, integrity, or confidentiality protection will prevent a MANE from performing media-aware operations other than discarding complete packets. In the case of confidentiality protection, it will even be prevented from discarding packets in a media-aware way. To be allowed to perform such operations, a MANE is required to be a trusted entity that is included in the security context establishment.

## 11. Congestion Control

Congestion control for RTP SHALL be used in accordance with RTP [RFC3550] and with any applicable RTP profile, e.g., AVP [RFC3551]. If best-effort service is being used, an additional requirement is that users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within an acceptable range. Packet loss is considered acceptable if a TCP flow across the same network path, and experiencing the same network conditions, would achieve an average throughput, measured on a reasonable timescale, that is not less than all RTP streams combined are achieving. This condition can be satisfied by implementing congestion-control mechanisms to adapt the transmission rate, the number of layers subscribed for a layered multicast session, or by arranging for a receiver to leave the session if the loss rate is unacceptably high.

The bitrate adaptation necessary for obeying the congestion control principle is easily achievable when real-time encoding is used, for example, by adequately tuning the quantization parameter. However, when pre-encoded content is being transmitted, bandwidth adaptation requires the pre-coded bitstream to be tailored for such adaptivity. The key mechanisms available in [VVC] are temporal scalability, and

spatial/SNR scalability. A media sender can remove NAL units belonging to higher temporal sub-layers (i.e., those NAL units with a high value of TID) or higher spatio-SNR layers (as indicated by interpreting the VPS) until the sending bitrate drops to an acceptable range.

The mechanisms mentioned above generally work within a defined profile and level and, therefore, no renegotiation of the channel is required. Only when non-downgradable parameters (such as profile) are required to be changed does it become necessary to terminate and restart the RTP stream(s). This may be accomplished by using different RTP payload types.

MANES MAY remove certain unusable packets from the RTP stream when that RTP stream was damaged due to previous packet losses. This can help reduce the network load in certain special cases. For example, MANES can remove those FUs where the leading FUs belonging to the same NAL unit have been lost or those dependent slice segments when the leading slice segments belonging to the same slice have been lost, because the trailing FUs or dependent slice segments are meaningless to most decoders. MANES can also remove higher temporal scalable layers if the outbound transmission (from the MANE's viewpoint) experiences congestion.

## 12. IANA Considerations

Placeholder

## 13. Acknowledgements

Dr. Byeongdoo Choi is thanked for the video codec related technical discussion and other aspects in this memo. Xin Zhao and Dr. Xiang Li are thanked for their contributions on [VVC] specification descriptive content. Spencer Dawkins is thanked for his valuable review comments that led to great improvements of this memo. Some parts of this specification share text with the RTP payload format for HEVC [RFC7798]. We thank the authors of that specification for their excellent work.

## 14. References

### 14.1. Normative References

[H.266] "ITU-T, Versatile Video Coding", n.d..

- [ISO23090-3] "ISO/IEC DIS Information technology --- Coded representation of immersive media --- Part 3 Versatile video codings", n.d., <<https://www.iso.org/standard/73022.html>>.
- [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>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.
- [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>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC4556] Zhu, L. and B. Tung, "Public Key Cryptography for Initial Authentication in Kerberos (PKINIT)", RFC 4556, DOI 10.17487/RFC4556, June 2006, <<https://www.rfc-editor.org/info/rfc4556>>.
- [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>>.

- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.
- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/info/rfc5124>>.
- [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>>.
- [RFC8082] Wenger, S., Lennox, J., Burman, B., and M. Westerlund, "Using Codec Control Messages in the RTP Audio-Visual Profile with Feedback with Layered Codecs", RFC 8082, DOI 10.17487/RFC8082, March 2017, <<https://www.rfc-editor.org/info/rfc8082>>.
- [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>>.
- [VVC] "Versatile Video Coding (Draft 10), Joint Video Experts Team (JVET)", July 2020.

#### 14.2. Informative References

- [CABAC] Sole, J, . and . et al, "Transform coefficient coding in HEVC, IEEE Transactions on Circuits and Systems for Video Technology", DOI 10.1109/TCSVT.2012.2223055, December 2012.
- [FrameMarking] Berger, E, ., Nandakumar, S, ., and . Zanaty M, "Frame Marking RTP Header Extension", Work in Progress draft-berger-avtext-framemarking , 2015.
- [Girod99] Girod, B, . and . et al, "Feedback-based error control for mobile video transmission, Proceedings of the IEEE", DOI 110.1109/5.790632, October 1999.
- [HEVC] "High efficiency video coding, ITU-T Recommendation H.265", April 2013.

- [MPEG2S] ISO/IEC, ., "Information technology - Generic coding of moving pictures and associated audio information - Part 1: Systems, ISO International Standard 13818-1", 2013.
- [RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, DOI 10.17487/RFC6051, November 2010, <<https://www.rfc-editor.org/info/rfc6051>>.
- [RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <<https://www.rfc-editor.org/info/rfc6184>>.
- [RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <<https://www.rfc-editor.org/info/rfc6190>>.
- [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>>.
- [RFC7798] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S., and M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <<https://www.rfc-editor.org/info/rfc7798>>.
- [Wang05] Wang, YK, ., Zhu, C, ., and . Li, H, "Error resilient video coding using flexible reference frames", Visual Communications and Image Processing 2005 (VCIP 2005) , July 2005.

#### Appendix A. Change History

draft-zhao-payload-rtp-vvc-00 ..... initial version

draft-zhao-payload-rtp-vvc-01 ..... editorial clarifications and corrections

draft-ietf-payload-rtp-vvc-00 ..... initial WG draft

draft-ietf-payload-rtp-vvc-01 ..... VVC specification update

Authors' Addresses

Shuai Zhao  
Tencent  
2747 Park Blvd  
Palo Alto 94588  
USA

Email: [shuai.zhao@ieee.org](mailto:shuai.zhao@ieee.org)

Stephan Wenger  
Tencent  
2747 Park Blvd  
Palo Alto 94588

Email: [stewe@stewe.org](mailto:stewe@stewe.org)

Yago Sanchez  
Fraunhofer HHI  
Einsteinufer 37  
Berlin 10587  
Germany

Email: [yago.sanchez@hhi.fraunhofer.de](mailto:yago.sanchez@hhi.fraunhofer.de)

avtcore  
Internet-Draft  
Intended status: Standards Track  
Expires: 2 February 2023

S. Zhao  
Intel  
S. Wenger  
Tencent  
Y. Sanchez  
Fraunhofer HHI  
Y.-K. Wang  
Bytedance Inc.  
M. M Hannuksela  
Nokia Technologies  
1 August 2022

RTP Payload Format for Versatile Video Coding (VVC)  
draft-ietf-avtcore-rtp-vvc-18

Abstract

This memo describes an RTP payload format for the video coding standard ITU-T Recommendation H.266 and ISO/IEC International Standard 23090-3, both also known as Versatile Video Coding (VVC) and developed by the Joint Video Experts Team (JVET). The RTP payload format allows for packetization of one or more Network Abstraction Layer (NAL) units in each RTP packet payload as well as fragmentation of a NAL unit into multiple RTP packets. The payload format has wide applicability in videoconferencing, Internet video streaming, and high-bitrate entertainment-quality video, among other applications.

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 <https://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 2 February 2023.

## Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1.	Introduction . . . . .	3
1.1.	Overview of the VVC Codec . . . . .	3
1.1.1.	Coding-Tool Features (informative) . . . . .	4
1.1.2.	Systems and Transport Interfaces (informative) . . . . .	6
1.1.3.	High-Level Picture Partitioning (informative) . . . . .	11
1.1.4.	NAL Unit Header . . . . .	13
1.2.	Overview of the Payload Format . . . . .	15
2.	Conventions . . . . .	15
3.	Definitions and Abbreviations . . . . .	15
3.1.	Definitions . . . . .	15
3.1.1.	Definitions from the VVC Specification . . . . .	16
3.1.2.	Definitions Specific to This Memo . . . . .	19
3.2.	Abbreviations . . . . .	19
4.	RTP Payload Format . . . . .	20
4.1.	RTP Header Usage . . . . .	21
4.2.	Payload Header Usage . . . . .	22
4.3.	Payload Structures . . . . .	22
4.3.1.	Single NAL Unit Packets . . . . .	23
4.3.2.	Aggregation Packets (APs) . . . . .	23
4.3.3.	Fragmentation Units . . . . .	28
4.4.	Decoding Order Number . . . . .	31
5.	Packetization Rules . . . . .	33
6.	De-packetization Process . . . . .	34
7.	Payload Format Parameters . . . . .	36
7.1.	Media Type Registration . . . . .	36
7.2.	Optional Parameters Definition . . . . .	37
7.3.	SDP Parameters . . . . .	47
7.3.1.	Mapping of Payload Type Parameters to SDP . . . . .	48
7.3.2.	Usage with SDP Offer/Answer Model . . . . .	50
7.3.3.	Multicast . . . . .	59
7.3.4.	Usage in Declarative Session Descriptions . . . . .	59
7.3.5.	Considerations for Parameter Sets . . . . .	61

8.	Use with Feedback Messages . . . . .	61
8.1.	Picture Loss Indication (PLI) . . . . .	61
8.2.	Full Intra Request (FIR) . . . . .	61
9.	Security Considerations . . . . .	62
10.	Congestion Control . . . . .	63
11.	IANA Considerations . . . . .	64
12.	Acknowledgements . . . . .	64
13.	References . . . . .	64
13.1.	Normative References . . . . .	64
13.2.	Informative References . . . . .	66
Appendix A.	Change History . . . . .	68
Authors'	Addresses . . . . .	68

## 1. Introduction

The Versatile Video Coding specification was formally published as both ITU-T Recommendation H.266 [VVC] and ISO/IEC International Standard 23090-3 [ISO23090-3]. VVC is reported to provide significant coding efficiency gains over High Efficiency Video Coding [HEVC], also known as H.265, and other earlier video codecs.

This memo specifies an RTP payload format for VVC. It shares its basic design with the NAL (Network Abstraction Layer) unit based RTP payload formats of AVC Video Coding [RFC6184], Scalable Video Coding (SVC) [RFC6190], High Efficiency Video Coding (HEVC) [RFC7798] and their respective predecessors. With respect to design philosophy, security, congestion control, and overall implementation complexity, it has similar properties to those earlier payload format specifications. This is a conscious choice, as at least RFC 6184 is widely deployed and generally known in the relevant implementer communities. Certain scalability-related mechanisms known from [RFC6190] were incorporated into this document, as VVC version 1 supports temporal, spatial, and signal-to-noise ratio (SNR) scalability.

### 1.1. Overview of the VVC Codec

VVC and HEVC share a similar hybrid video codec design. In this memo, we provide a very brief overview of those features of VVC that are, in some form, addressed by the payload format specified herein. Implementers have to read, understand, and apply the ITU-T/ISO/IEC specifications pertaining to VVC to arrive at interoperable, well-performing implementations.

Conceptually, both VVC and HEVC include a Video Coding Layer (VCL), which is often used to refer to the coding-tool features, and a NAL, which is often used to refer to the systems and transport interface aspects of the codecs.

### 1.1.1.1. Coding-Tool Features (informative)

Coding tool features are described below with occasional reference to the coding tool set of HEVC, which is well known in the community.

Similar to earlier hybrid-video-coding-based standards, including HEVC, the following basic video coding design is employed by VVC. A prediction signal is first formed by either intra- or motion-compensated prediction, and the residual (the difference between the original and the prediction) is then coded. The gains in coding efficiency are achieved by redesigning and improving almost all parts of the codec over earlier designs. In addition, VVC includes several tools to make the implementation on parallel architectures easier.

Finally, VVC includes temporal, spatial, and SNR scalability as well as multiview coding support.

#### Coding blocks and transform structure

Among major coding-tool differences between HEVC and VVC, one of the important improvements is the more flexible coding tree structure in VVC, i.e., multi-type tree. In addition to quadtree, binary and ternary trees are also supported, which contributes significant improvement in coding efficiency. Moreover, the maximum size of a coding tree unit (CTU) is increased from 64x64 to 128x128. To improve the coding efficiency of chroma signal, luma chroma separated trees at CTU level may be employed for intra-slices. The square transforms in HEVC are extended to non-square transforms for rectangular blocks resulting from binary and ternary tree splits. Besides, VVC supports multiple transform sets (MTS), including DCT-2, DST-7, and DCT-8 as well as the non-separable secondary transform. The transforms used in VVC can have different sizes with support for larger transform sizes. For DCT-2, the transform sizes range from 2x2 to 64x64, and for DST-7 and DCT-8, the transform sizes range from 4x4 to 32x32. In addition, VVC also support sub-block transform for both intra and inter coded blocks. For intra coded blocks, intra sub-partitioning (ISP) may be used to allow sub-block based intra prediction and transform. For inter blocks, sub-block transform may be used assuming that only a part of an inter-block has non-zero transform coefficients.

#### Entropy coding

Similar to HEVC, VVC uses a single entropy-coding engine, which is based on context adaptive binary arithmetic coding [CABAC], but with the support of multi-window sizes. The window sizes can be initialized differently for different context models. Due to such a design, it has more efficient adaptation speed and better coding

efficiency. A joint chroma residual coding scheme is applied to further exploit the correlation between the residuals of two color components. In VVC, different residual coding schemes are applied for regular transform coefficients and residual samples generated using transform-skip mode.

#### In-loop filtering

VVC has more feature support in loop filters than HEVC. The deblocking filter in VVC is similar to HEVC but operates at a smaller grid. After deblocking and sample adaptive offset (SAO), an adaptive loop filter (ALF) may be used. As a Wiener filter, ALF reduces distortion of decoded pictures. Besides, VVC introduces a new module called luma mapping with chroma scaling to fully utilize the dynamic range of signal so that rate-distortion performance of both Standard Dynamic Range (SDR) and High Dynamic Range (HDR) content is improved.

#### Motion prediction and coding

Compared to HEVC, VVC introduces several improvements in this area. First, there is the adaptive motion vector resolution (AMVR), which can save bit cost for motion vectors by adaptively signaling motion vector resolution. Then the affine motion compensation is included to capture complicated motion like zooming and rotation. Meanwhile, prediction refinement with the optical flow with affine mode (PROF) is further deployed to mimic affine motion at the pixel level. Thirdly the decoder side motion vector refinement (DMVR) is a method to derive MV vector at decoder side based on block matching so that fewer bits may be spent on motion vectors. Bi-directional optical flow (BDOF) is a similar method to PROF. BDOF adds a sample wise offset at 4x4 sub-block level that is derived with equations based on gradients of the prediction samples and a motion difference relative to CU motion vectors. Furthermore, merge with motion vector difference (MMVD) is a special mode, which further signals a limited set of motion vector differences on top of merge mode. In addition to MMVD, there are another three types of special merge modes, i.e., sub-block merge, triangle, and combined intra-/inter-prediction (CIIP). Sub-block merge list includes one candidate of sub-block temporal motion vector prediction (SbTMVP) and up to four candidates of affine motion vectors. Triangle is based on triangular block motion compensation. CIIP combines intra- and inter- predictions with weighting. Adaptive weighting may be employed with a block-level tool called bi-prediction with CU based weighting (BCW) which provides more flexibility than in HEVC.

#### Intra prediction and intra-coding

To capture the diversified local image texture directions with finer granularity, VVC supports 65 angular directions instead of 33 directions in HEVC. The intra mode coding is based on a 6-most-probable-mode scheme, and the 6 most probable modes are derived using the neighboring intra prediction directions. In addition, to deal with the different distributions of intra prediction angles for different block aspect ratios, a wide-angle intra prediction (WAIP) scheme is applied in VVC by including intra prediction angles beyond those present in HEVC. Unlike HEVC which only allows using the most adjacent line of reference samples for intra prediction, VVC also allows using two further reference lines, as known as multi-reference-line (MRL) intra prediction. The additional reference lines can be only used for the 6 most probable intra prediction modes. To capture the strong correlation between different colour components, in VVC, a cross-component linear mode (CCLM) is utilized which assumes a linear relationship between the luma sample values and their associated chroma samples. For intra prediction, VVC also applies a position-dependent prediction combination (PDPC) for refining the prediction samples closer to the intra prediction block boundary. Matrix-based intra prediction (MIP) modes are also used in VVC which generates an up to 8x8 intra prediction block using a weighted sum of downsampled neighboring reference samples, and the weights are hardcoded constants.

Other coding-tool features

VVC introduces dependent quantization (DQ) to reduce quantization error by state-based switching between two quantizers.

#### 1.1.2. Systems and Transport Interfaces (informative)

VVC inherits the basic systems and transport interfaces designs from HEVC and AVC. These include the NAL-unit-based syntax structure, the hierarchical syntax and data unit structure, the supplemental enhancement information (SEI) message mechanism, and the video buffering model based on the hypothetical reference decoder (HRD). The scalability features of VVC are conceptually similar to the scalable variant of HEVC known as SHVC. The hierarchical syntax and data unit structure consists of parameter sets at various levels (decoder, sequence (pertaining to all), sequence (pertaining to a single), picture), picture-level header parameters, slice-level header parameters, and lower-level parameters.

A number of key components that influenced the network abstraction layer design of VVC as well as this memo are described below

Decoding capability information

The decoding capability information includes parameters that stay constant for the lifetime of a VVC bitstream in the duration of a video conference, continuous video stream, and similar--any video that is processed by a decoder between setup and teardown. For streaming, the requirement of constant parameters pertains through splicing. Such information includes profile, level, and sub-profile information to determine a maximum capability interop point that is guaranteed to be never exceeded, even if splicing of video sequences occurs within a session. It further includes constraint fields (most of which are flags), which can optionally be set to indicate that the video bitstream will be constrained in the use of certain features as indicated by the values of those fields. With this, a bitstream can be labeled as not using certain tools, which allows among other things for resource allocation in a decoder implementation.

#### Video parameter set

The video parameter set (VPS) pertains to one or more coded video sequences (CVSs) of multiple layers covering the same range of access units, and includes, among other information, decoding dependency expressed as information for reference picture list construction of enhancement layers. The VPS provides a "big picture" of a scalable sequence, including what types of operation points are provided, the profile, tier, and level of the operation points, and some other high-level properties of the bitstream that can be used as the basis for session negotiation and content selection, etc. One VPS may be referenced by one or more sequence parameter sets.

#### Sequence parameter set

The sequence parameter set (SPS) contains syntax elements pertaining to a coded layer video sequence (CLVS), which is a group of pictures belonging to the same layer, starting with a random access point, and followed by pictures that may depend on each other, until the next random access point picture. In MPEG-2, the equivalent of a CVS was a group of pictures (GOP), which normally started with an I frame and was followed by P and B frames. While more complex in its options of random access points, VVC retains this basic concept. One remarkable difference of VVC is that a CLVS may start with a Gradual Decoding Refresh (GDR) picture, without requiring presence of traditional random access points in the bitstream, such as instantaneous decoding refresh (IDR) or clean random access (CRA) pictures. In many TV-like applications, a CVS contains a few hundred milliseconds to a few seconds of video. In video conferencing (without switching MCUs involved), a CVS can be as long in duration as the whole session.

#### Picture and adaptation parameter set

The picture parameter set and the adaptation parameter set (PPS and APS, respectively) carry information pertaining to zero or more pictures and zero or more slices, respectively. The PPS contains information that is likely to stay constant from picture to picture, at least for pictures for a certain type—whereas the APS contains information, such as adaptive loop filter coefficients, that are likely to change from picture to picture or even within a picture. A single APS is referenced by all slices of the same picture if that APS contains information about luma mapping with chroma scaling (LMCS) or scaling list. Different APSs containing ALF parameters can be referenced by slices of the same picture.

#### Picture header

A Picture Header contains information that is common to all slices that belong to the same picture. Being able to send that information as a separate NAL unit when pictures are split into several slices allows for saving bitrate, compared to repeating the same information in all slices. However, there might be scenarios where low-bitrate video is transmitted using a single slice per picture. Having a separate NAL unit to convey that information incurs in an overhead for such scenarios. For such scenarios, the picture header syntax structure is directly included in the slice header, instead of its own NAL unit. The mode of the picture header syntax structure being included in its own NAL unit or not can only be switched on/off for an entire CLVS, and can only be switched off when in the entire CLVS each picture contains only one slice.

#### Profile, tier, and level

The profile, tier and level syntax structures in DCI, VPS and SPS contain profile, tier, level information for all layers that refer to the DCI, for layers associated with one or more output layer sets specified by the VPS, and for any layer that refers to the SPS, respectively.

#### Sub-profiles

Within the VVC specification, a sub-profile is a 32-bit number, coded according to ITU-T Rec. T.35, that does not carry a semantics. It is carried in the `profile_tier_level` structure and hence (potentially) present in the DCI, VPS, and SPS. External registration bodies can register a T.35 codepoint with ITU-T registration authorities and associate with their registration a description of bitstream restrictions beyond the profiles defined by ITU-T and ISO/IEC. This would allow encoder manufacturers to label the bitstreams generated by their encoder as complying with such sub-profile. It is expected that upstream standardization organizations (such as: DVB and ATSC),

as well as walled-garden video services will take advantage of this labeled system. In contrast to "normal" profiles, it is expected that sub-profiles may indicate encoder choices traditionally left open in the (decoder-centric) video coding specs, such as GOP structures, minimum/maximum QP values, and the mandatory use of certain tools or SEI messages.

#### General constraint fields

The `profile_tier_level` structure carries a considerable number of constraint fields (most of which are flags), which an encoder can use to indicate to a decoder that it will not use a certain tool or technology. They were included in reaction to a perceived market need for labeled a bitstream as not exercising a certain tool that has become commercially unviable.

#### Temporal scalability support

VVC includes support of temporal scalability, by inclusion of the signaling of `TemporalId` in the NAL unit header, the restriction that pictures of a particular temporal sublayer cannot be used for inter prediction reference by pictures of a lower temporal sublayer, the sub-bitstream extraction process, and the requirement that each sub-bitstream extraction output be a conforming bitstream. Media-Aware Network Elements (MANEs) can utilize the `TemporalId` in the NAL unit header for stream adaptation purposes based on temporal scalability.

#### Reference picture resampling (RPR)

In AVC and HEVC, the spatial resolution of pictures cannot change unless a new sequence using a new SPS starts, with an Intra random access point (IRAP) picture. VVC enables picture resolution change within a sequence at a position without encoding an IRAP picture, which is always intra-coded. This feature is sometimes referred to as reference picture resampling (RPR), as the feature needs resampling of a reference picture used for inter prediction when that reference picture has a different resolution than the current picture being decoded. RPR allows resolution change without the need of coding an IRAP picture and hence avoids a momentary bit rate spike caused by an IRAP picture in streaming or video conferencing scenarios, e.g., to cope with network condition changes. RPR can also be used in application scenarios wherein zooming of the entire video region or some region of interest is needed.

#### Spatial, SNR, and multiview scalability

VVC includes support for spatial, SNR, and multiview scalability. Scalable video coding is widely considered to have technical benefits and enrich services for various video applications. Until recently, however, the functionality has not been included in the first version of specifications of the video codecs. In VVC, however, all those forms of scalability are supported in the first version of VVC natively through the signaling of the `nuh_layer_id` in the NAL unit header, the VPS which associates layers with given `nuh_layer_id` to each other, reference picture selection, reference picture resampling for spatial scalability, and a number of other mechanisms not relevant for this memo.

#### Spatial scalability

With the existence of Reference Picture Resampling (RPR), the additional burden for scalability support is just a modification of the high-level syntax (HLS). The inter-layer prediction is employed in a scalable system to improve the coding efficiency of the enhancement layers. In addition to the spatial and temporal motion-compensated predictions that are available in a single-layer codec, the inter-layer prediction in VVC uses the possibly resampled video data of the reconstructed reference picture from a reference layer to predict the current enhancement layer. The resampling process for inter-layer prediction, when used, is performed at the block-level, reusing the existing interpolation process for motion compensation in single-layer coding. It means that no additional resampling process is needed to support spatial scalability.

#### SNR scalability

SNR scalability is similar to spatial scalability except that the resampling factors are 1:1. In other words, there is no change in resolution, but there is inter-layer prediction.

#### Multiview scalability

The first version of VVC also supports multiview scalability, wherein a multi-layer bitstream carries layers representing multiple views, and one or more of the represented views can be output at the same time.

#### SEI messages

Supplemental enhancement information (SEI) messages are information in the bitstream that do not influence the decoding process as specified in the VVC spec, but address issues of representation/

rendering of the decoded bitstream, label the bitstream for certain applications, among other, similar tasks. The overall concept of SEI messages and many of the messages themselves has been inherited from the AVC and HEVC specs. Except for the SEI messages that affect the specification of the hypothetical reference decoder (HRD), other SEI messages for use in the VVC environment, which are generally useful also in other video coding technologies, are not included in the main VVC specification but in a companion specification [VSEI].

### 1.1.3. High-Level Picture Partitioning (informative)

VVC inherited the concept of tiles and wavefront parallel processing (WPP) from HEVC, with some minor to moderate differences. The basic concept of slices was kept in VVC but designed in an essentially different form. VVC is the first video coding standard that includes subpictures as a feature, which provides the same functionality as HEVC motion-constrained tile sets (MCTSs) but designed differently to have better coding efficiency and to be friendlier for usage in application systems. More details of these differences are described below.

#### Tiles and WPP

Same as in HEVC, a picture can be split into tile rows and tile columns in VVC, in-picture prediction across tile boundaries is disallowed, etc. However, the syntax for signaling of tile partitioning has been simplified, by using a unified syntax design for both the uniform and the non-uniform mode. In addition, signaling of entry point offsets for tiles in the slice header is optional in VVC while it is mandatory in HEVC. The WPP design in VVC has two differences compared to HEVC: i) The CTU row delay is reduced from two CTUs to one CTU; ii) signaling of entry point offsets for WPP in the slice header is optional in VVC while it is mandatory in HEVC.

#### Slices

In VVC, the conventional slices based on CTUs (as in HEVC) or macroblocks (as in AVC) have been removed. The main reasoning behind this architectural change is as follows. The advances in video coding since 2003 (the publication year of AVC v1) have been such that slice-based error concealment has become practically impossible, due to the ever-increasing number and efficiency of in-picture and inter-picture prediction mechanisms. An error-concealed picture is the decoding result of a transmitted coded picture for which there is some data loss (e.g., loss of some slices) of the coded picture or a reference picture for at least some part of the coded picture is not error-free (e.g., that reference picture was an error-concealed

picture). For example, when one of the multiple slices of a picture is lost, it may be error-concealed using an interpolation of the neighboring slices. While advanced video coding prediction mechanisms provide significantly higher coding efficiency, they also make it harder for machines to estimate the quality of an error-concealed picture, which was already a hard problem with the use of simpler prediction mechanisms. Advanced in-picture prediction mechanisms also cause the coding efficiency loss due to splitting a picture into multiple slices to be more significant. Furthermore, network conditions become significantly better while at the same time techniques for dealing with packet losses have become significantly improved. As a result, very few implementations have recently used slices for maximum transmission unit size matching. Instead, substantially all applications where low-delay error resilience is required (e.g., video telephony and video conferencing) rely on system/transport-level error resilience (e.g., retransmission, forward error correction) and/or picture-based error resilience tools (feedback-based error resilience, insertion of IRAPs, scalability with higher protection level of the base layer, and so on). Considering all the above, nowadays it is very rare that a picture that cannot be correctly decoded is passed to the decoder, and when such a rare case occurs, the system can afford to wait for an error-free picture to be decoded and available for display without resulting in frequent and long periods of picture freezing seen by end users.

Slices in VVC have two modes: rectangular slices and raster-scan slices. The rectangular slice, as indicated by its name, covers a rectangular region of the picture. Typically, a rectangular slice consists of several complete tiles. However, it is also possible that a rectangular slice is a subset of a tile and consists of one or more consecutive, complete CTU rows within a tile. A raster-scan slice consists of one or more complete tiles in a tile raster scan order, hence the region covered by a raster-scan slices need not but could have a non-rectangular shape, but it may also happen to have the shape of a rectangle. The concept of slices in VVC is therefore strongly linked to or based on tiles instead of CTUs (as in HEVC) or macroblocks (as in AVC).

#### Subpictures

VVC is the first video coding standard that includes the support of subpictures as a feature. Each subpicture consists of one or more complete rectangular slices that collectively cover a rectangular region of the picture. A subpicture may be either specified to be extractable (i.e., coded independently of other subpictures of the same picture and of earlier pictures in decoding order) or not extractable. Regardless of whether a subpicture is extractable or

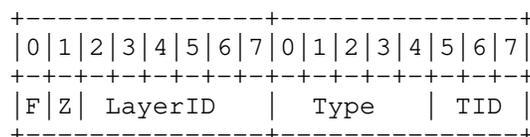
not, the encoder can control whether in-loop filtering (including deblocking, SAO, and ALF) is applied across the subpicture boundaries individually for each subpicture.

Functionally, subpictures are similar to the motion-constrained tile sets (MCTSS) in HEVC. They both allow independent coding and extraction of a rectangular subset of a sequence of coded pictures, for use cases like viewport-dependent 360o video streaming optimization and region of interest (ROI) applications.

There are several important design differences between subpictures and MCTSSs. First, the subpictures feature in VVC allows motion vectors of a coding block pointing outside of the subpicture even when the subpicture is extractable by applying sample padding at subpicture boundaries in this case, similarly as at picture boundaries. Second, additional changes were introduced for the selection and derivation of motion vectors in the merge mode and in the decoder side motion vector refinement process of VVC. This allows higher coding efficiency compared to the non-normative motion constraints applied at the encoder-side for MCTSSs. Third, rewriting of SHs (and PH NAL units, when present) is not needed when extracting one or more extractable subpictures from a sequence of pictures to create a sub-bitstream that is a conforming bitstream. In sub-bitstream extractions based on HEVC MCTSSs, rewriting of SHs is needed. Note that in both HEVC MCTSSs extraction and VVC subpictures extraction, rewriting of SPSs and PPSs is needed. However, typically there are only a few parameter sets in a bitstream, while each picture has at least one slice, therefore rewriting of SHs can be a significant burden for application systems. Fourth, slices of different subpictures within a picture are allowed to have different NAL unit types. Fifth, VVC specifies HRD and level definitions for subpicture sequences, thus the conformance of the sub-bitstream of each extractable subpicture sequence can be ensured by encoders.

#### 1.1.4. NAL Unit Header

VVC maintains the NAL unit concept of HEVC with modifications. VVC uses a two-byte NAL unit header, as shown in Figure 1. The payload of a NAL unit refers to the NAL unit excluding the NAL unit header.



The Structure of the VVC NAL Unit Header.

Figure 1

The semantics of the fields in the NAL unit header are as specified in VVC and described briefly below for convenience. In addition to the name and size of each field, the corresponding syntax element name in VVC is also provided.

F: 1 bit

forbidden\_zero\_bit. Required to be zero in VVC. Note that the inclusion of this bit in the NAL unit header was to enable transport of VVC video over MPEG-2 transport systems (avoidance of start code emulations) [MPEG2S]. In the context of this payload format, the value 1 may be used to indicate a syntax violation, e.g., for a NAL unit resulted from aggregating a number of fragmented units of a NAL unit but missing the last fragment, as described in the last sentence of section 4.3.3.

Z: 1 bit

nuh\_reserved\_zero\_bit. Required to be zero in VVC, and reserved for future extensions by ITU-T and ISO/IEC. This memo does not overload the "Z" bit for local extensions, as a) overloading the "F" bit is sufficient and b) to preserve the usefulness of this memo to possible future versions of [VVC].

LayerId: 6 bits

nuh\_layer\_id. Identifies the layer a NAL unit belongs to, wherein a layer may be, e.g., a spatial scalable layer, a quality scalable layer, a layer containing a different view, etc.

Type: 5 bits

nal\_unit\_type. This field specifies the NAL unit type as defined in Table 5 of [VVC]. For a reference of all currently defined NAL unit types and their semantics, please refer to Section 7.4.2.2 in [VVC].

TID: 3 bits

nuh\_temporal\_id\_plus1. This field specifies the temporal identifier of the NAL unit plus 1. The value of TemporalId is equal to TID minus 1. A TID value of 0 is illegal to ensure that there is at least one bit in the NAL unit header equal to 1, so to enable the consideration of start code emulations in the NAL unit payload data independent of the NAL unit header.

## 1.2. Overview of the Payload Format

This payload format defines the following processes required for transport of VVC coded data over RTP [RFC3550]:

- \* Usage of RTP header with this payload format
- \* Packetization of VVC coded NAL units into RTP packets using three types of payload structures: a single NAL unit packet, aggregation packet, and fragment unit
- \* Transmission of VVC NAL units of the same bitstream within a single RTP stream
- \* Media type parameters to be used with the Session Description Protocol (SDP) [RFC8866]
- \* Usage of RTCP feedback messages

## 2. Conventions

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. Definitions and Abbreviations

### 3.1. Definitions

This document uses the terms and definitions of VVC. Section 3.1.1 lists relevant definitions from [VVC] for convenience. Section 3.1.2 provides definitions specific to this memo. All the used terms and definitions in this memo are verbatim copies of [VVC] specification.

### 3.1.1. Definitions from the VVC Specification

**Access unit (AU):** A set of PUs that belong to different layers and contain coded pictures associated with the same time for output from the DPB.

**Adaptation parameter set (APS):** A syntax structure containing syntax elements that apply to zero or more slices as determined by zero or more syntax elements found in slice headers.

**Bitstream:** A sequence of bits, in the form of a NAL unit stream or a byte stream, that forms the representation of a sequence of AUs forming one or more coded video sequences (CVSSs).

**Coded picture:** A coded representation of a picture comprising VCL NAL units with a particular value of `nuh_layer_id` within an AU and containing all CTUs of the picture.

**Clean random access (CRA) PU:** A PU in which the coded picture is a CRA picture.

**Clean random access (CRA) picture:** An IRAP picture for which each VCL NAL unit has `nal_unit_type` equal to `CRA_NUT`.

**Coded video sequence (CVS):** A sequence of AUs that consists, in decoding order, of a CVSS AU, followed by zero or more AUs that are not CVSS AUs, including all subsequent AUs up to but not including any subsequent AU that is a CVSS AU.

**Coded video sequence start (CVSS) AU:** An AU in which there is a PU for each layer in the CVS and the coded picture in each PU is a CLVSS picture.

**Coded layer video sequence (CLVS):** A sequence of PUs with the same value of `nuh_layer_id` that consists, in decoding order, of a CLVSS PU, followed by zero or more PUs that are not CLVSS PUs, including all subsequent PUs up to but not including any subsequent PU that is a CLVSS PU.

**Coded layer video sequence start (CLVSS) PU:** A PU in which the coded picture is a CLVSS picture.

**Coded layer video sequence start (CLVSS) picture:** A coded picture that is an IRAP picture with `NoOutputBeforeRecoveryFlag` equal to 1 or a GDR picture with `NoOutputBeforeRecoveryFlag` equal to 1.

**Coding tree unit (CTU):** A CTB of luma samples, two corresponding CTBs of chroma samples of a picture that has three sample arrays, or a CTB of samples of a monochrome picture or a picture that is coded using three separate colour planes and syntax structures used to code the samples.

**Decoding Capability Information (DCI):** A syntax structure containing syntax elements that apply to the entire bitstream.

**Decoded picture buffer (DPB):** A buffer holding decoded pictures for reference, output reordering, or output delay specified for the hypothetical reference decoder.

**Gradual decoding refresh (GDR) picture:** A picture for which each VCL NAL unit has `nal_unit_type` equal to `GDR_NUT`.

**Instantaneous decoding refresh (IDR) PU:** A PU in which the coded picture is an IDR picture.

**Instantaneous decoding refresh (IDR) picture:** An IRAP picture for which each VCL NAL unit has `nal_unit_type` equal to `IDR_W_RADL` or `IDR_N_LP`.

**Intra random access point (IRAP) AU:** An AU in which there is a PU for each layer in the CVS and the coded picture in each PU is an IRAP picture.

**Intra random access point (IRAP) PU:** A PU in which the coded picture is an IRAP picture.

**Intra random access point (IRAP) picture:** A coded picture for which all VCL NAL units have the same value of `nal_unit_type` in the range of `IDR_W_RADL` to `CRA_NUT`, inclusive.

**Layer:** A set of VCL NAL units that all have a particular value of `nuh_layer_id` and the associated non-VCL NAL units.

**Network abstraction layer (NAL) unit:** A syntax structure containing an indication of the type of data to follow and bytes containing that data in the form of an RBSP interspersed as necessary with emulation prevention bytes.

**Network abstraction layer (NAL) unit stream:** A sequence of NAL units.

**Output Layer Set (OLS):** A set of layers for which one or more layers are specified as the output layers.

Operation point (OP): A temporal subset of an OLS, identified by an OLS index and a highest value of TemporalId.

Picture parameter set (PPS): A syntax structure containing syntax elements that apply to zero or more entire coded pictures as determined by a syntax element found in each slice header.

Picture unit (PU): A set of NAL units that are associated with each other according to a specified classification rule, are consecutive in decoding order, and contain exactly one coded picture.

Random access: The act of starting the decoding process for a bitstream at a point other than the beginning of the stream.

Sequence parameter set (SPS): A syntax structure containing syntax elements that apply to zero or more entire CLVSSs as determined by the content of a syntax element found in the PPS referred to by a syntax element found in each picture header.

Slice: An integer number of complete tiles or an integer number of consecutive complete CTU rows within a tile of a picture that are exclusively contained in a single NAL unit.

Slice header (SH): A part of a coded slice containing the data elements pertaining to all tiles or CTU rows within a tile represented in the slice.

Sublayer: A temporal scalable layer of a temporal scalable bitstream consisting of VCL NAL units with a particular value of the TemporalId variable, and the associated non-VCL NAL units.

Subpicture: An rectangular region of one or more slices within a picture.

Sublayer representation: A subset of the bitstream consisting of NAL units of a particular sublayer and the lower sublayers.

Tile: A rectangular region of CTUs within a particular tile column and a particular tile row in a picture.

Tile column: A rectangular region of CTUs having a height equal to the height of the picture and a width specified by syntax elements in the picture parameter set.

Tile row: A rectangular region of CTUs having a height specified by syntax elements in the picture parameter set and a width equal to the width of the picture.

Video coding layer (VCL) NAL unit: A collective term for coded slice NAL units and the subset of NAL units that have reserved values of `nal_unit_type` that are classified as VCL NAL units in this Specification.

### 3.1.2. Definitions Specific to This Memo

Media-Aware Network Element (MANE): A network element, such as a middlebox, selective forwarding unit, or application-layer gateway that is capable of parsing certain aspects of the RTP payload headers or the RTP payload and reacting to their contents.

Informative note: The concept of a MANE goes beyond normal routers or gateways in that a MANE has to be aware of the signaling (e.g., to learn about the payload type mappings of the media streams), and in that it has to be trusted when working with Secure RTP (SRTP). The advantage of using MANEs is that they allow packets to be dropped according to the needs of the media coding. For example, if a MANE has to drop packets due to congestion on a certain link, it can identify and remove those packets whose elimination produces the least adverse effect on the user experience. After dropping packets, MANEs must rewrite RTCP packets to match the changes to the RTP stream, as specified in Section 7 of [RFC3550].

NAL unit decoding order: A NAL unit order that conforms to the constraints on NAL unit order given in Section 7.4.2.4 in [VVC], follow the Order of NAL units in the bitstream.

RTP stream (See [RFC7656]): Within the scope of this memo, one RTP stream is utilized to transport a VVC bitstream, which may contain one or more layers, and each layer may contain one or more temporal sublayers.

Transmission order: The order of packets in ascending RTP sequence number order (in modulo arithmetic). Within an aggregation packet, the NAL unit transmission order is the same as the order of appearance of NAL units in the packet.

### 3.2. Abbreviations

AU	Access Unit
AP	Aggregation Packet
APS	Adaptation Parameter Set
CTU	Coding Tree Unit

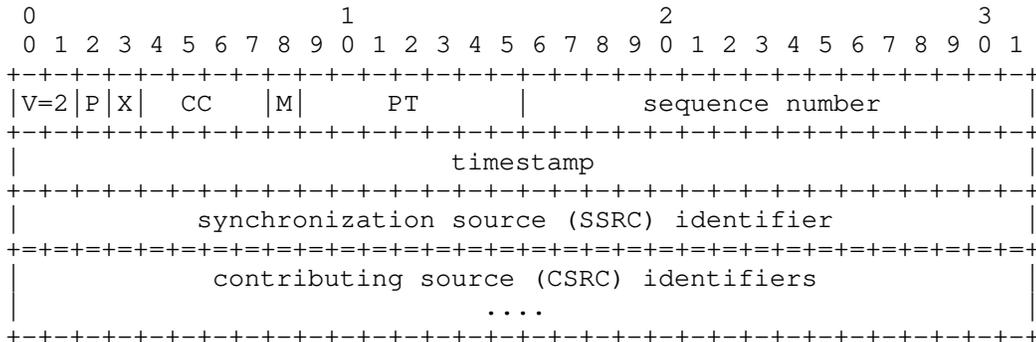
CVS	Coded Video Sequence
DPB	Decoded Picture Buffer
DCI	Decoding Capability Information
DON	Decoding Order Number
FIR	Full Intra Request
FU	Fragmentation Unit
GDR	Gradual Decoding Refresh
HRD	Hypothetical Reference Decoder
IDR	Instantaneous Decoding Refresh
IRAP	Intra Random Access Point
MANE	Media-Aware Network Element
MTU	Maximum Transfer Unit
NAL	Network Abstraction Layer
NALU	Network Abstraction Layer Unit
OLS	Output Layer Set
PLI	Picture Loss Indication
PPS	Picture Parameter Set
RPSI	Reference Picture Selection Indication
SEI	Supplemental Enhancement Information
SLI	Slice Loss Indication
SPS	Sequence Parameter Set
VCL	Video Coding Layer
VPS	Video Parameter Set

#### 4. RTP Payload Format

4.1. RTP Header Usage

The format of the RTP header is specified in [RFC3550] (reprinted as Figure 2 for convenience). This payload format uses the fields of the header in a manner consistent with that specification.

The RTP payload (and the settings for some RTP header bits) for aggregation packets and fragmentation units are specified in Section 4.3.2 and Section 4.3.3, respectively.



RTP Header According to [RFC3550]

Figure 2

The RTP header information to be set according to this RTP payload format is set as follows:

Marker bit (M): 1 bit

Set for the last packet, in transmission order, among each set of packets that contain NAL units of one access unit. This is in line with the normal use of the M bit in video formats to allow an efficient playout buffer handling.

Payload Type (PT): 7 bits

The assignment of an RTP payload type for this new packet format is outside the scope of this document and will not be specified here. The assignment of a payload type has to be performed either through the profile used or in a dynamic way.

Sequence Number (SN): 16 bits

Set and used in accordance with [RFC3550].

Timestamp: 32 bits

The RTP timestamp is set to the sampling timestamp of the content. A 90 kHz clock rate MUST be used. If the NAL unit has no timing properties of its own (e.g., parameter set and SEI NAL units), the RTP timestamp MUST be set to the RTP timestamp of the coded pictures of the access unit in which the NAL unit (according to Section 7.4.2.4 of [VVC]) is included. Receivers MUST use the RTP timestamp for the display process, even when the bitstream contains picture timing SEI messages or decoding unit information SEI messages as specified in [VVC].

Informative note: When picture timing SEI messages are present, the RTP sender is responsible to ensure that the RTP timestamps are consistent with the timing information carried in the picture timing SEI messages.

Synchronization source (SSRC): 32 bits

Used to identify the source of the RTP packets. A single SSRC is used for all parts of a single bitstream.

#### 4.2. Payload Header Usage

The first two bytes of the payload of an RTP packet are referred to as the payload header. The payload header consists of the same fields (F, Z, LayerId, Type, and TID) as the NAL unit header as shown in Section 1.1.4, irrespective of the type of the payload structure.

The TID value indicates (among other things) the relative importance of an RTP packet, for example, because NAL units belonging to higher temporal sublayers are not used for the decoding of lower temporal sublayers. A lower value of TID indicates a higher importance. More-important NAL units MAY be better protected against transmission losses than less-important NAL units.

#### 4.3. Payload Structures

Three different types of RTP packet payload structures are specified. A receiver can identify the type of an RTP packet payload through the Type field in the payload header.

The three different payload structures are as follows:

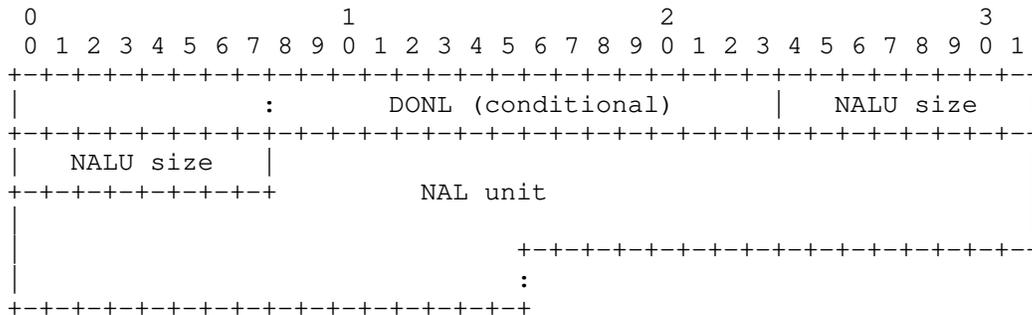
- \* Single NAL unit packet: Contains a single NAL unit in the payload, and the NAL unit header of the NAL unit also serves as the payload header. This payload structure is specified in Section 4.3.1.





An AP MUST carry at least two aggregation units and can carry as many aggregation units as necessary; however, the total amount of data in an AP obviously MUST fit into an IP packet, and the size SHOULD be chosen so that the resulting IP packet is smaller than the MTU size so to avoid IP layer fragmentation. An AP MUST NOT contain FUs specified in Section 4.3.3. APs MUST NOT be nested; i.e., an AP can not contain another AP.

The first aggregation unit in an AP consists of a conditional 16-bit DONL field (in network byte order) followed by a 16-bit unsigned size information (in network byte order) that indicates the size of the NAL unit in bytes (excluding these two octets, but including the NAL unit header), followed by the NAL unit itself, including its NAL unit header, as shown in Figure 5.



The Structure of the First Aggregation Unit in an AP

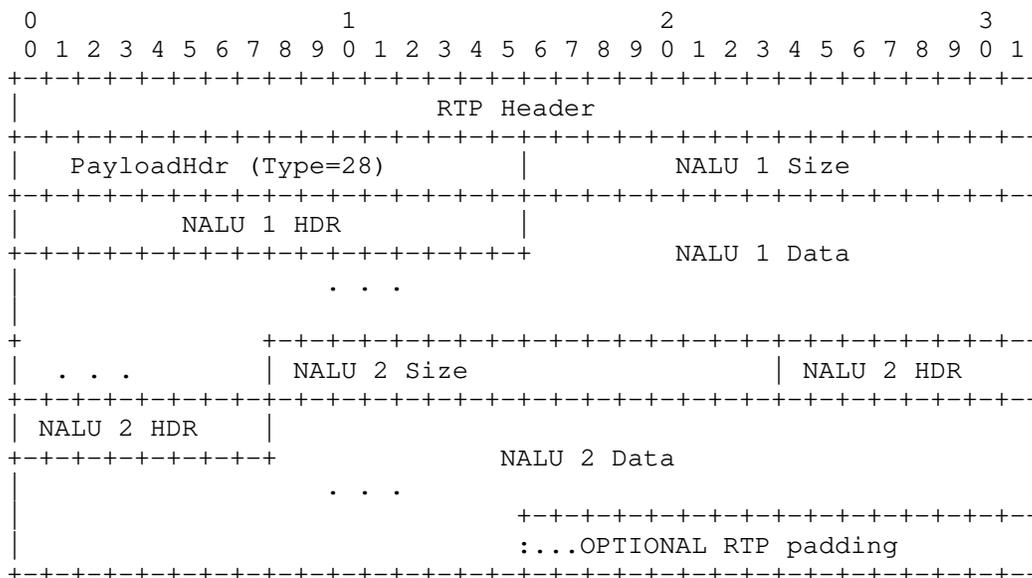
Figure 5

**Informative Note:** The first octet of Figure 5 (indicated by the first colon) belongs to a previous aggregation unit. It is depicted to emphasize that aggregation units are octet-aligned only. Similarly, the NAL unit carried in the aggregation unit can terminate at the octet boundary.

The DONL field, when present, specifies the value of the 16 least significant bits of the decoding order number of the aggregated NAL unit.

If sprop-max-don-diff is greater than 0, the DONL field MUST be present in an aggregation unit that is the first aggregation unit in an AP, and the variable DON for the aggregated NAL unit is derived as equal to the value of the DONL field, and the variable DON for an aggregation unit that is not the first aggregation unit in an AP aggregated NAL unit is derived as equal to the DON of the preceding aggregated NAL unit in the same AP plus 1 modulo 65536. Otherwise

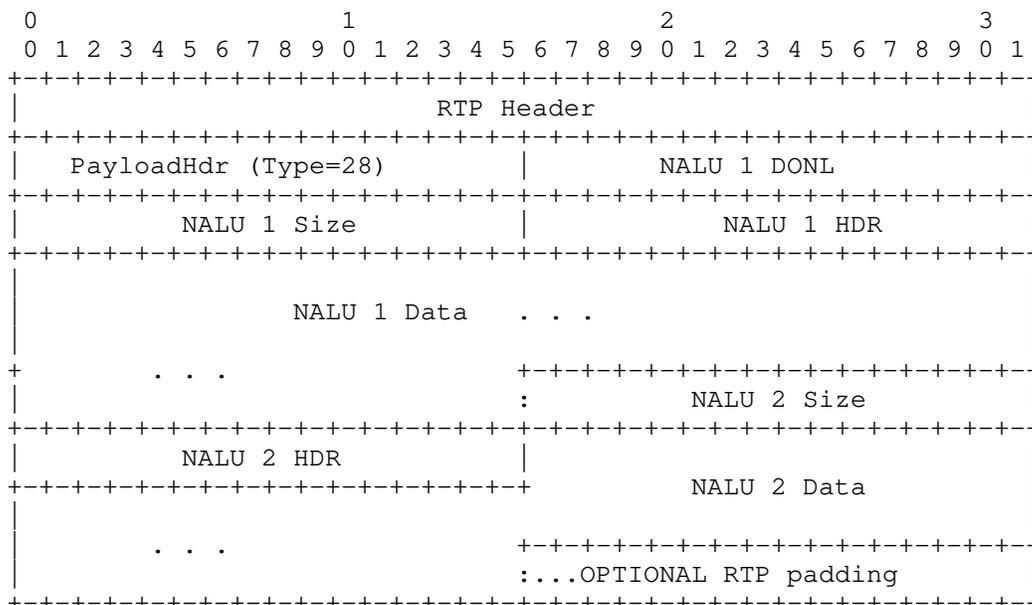




An Example of an AP Packet Containing  
Two Aggregation Units without the DONL Field

Figure 7

Figure 8 presents an example of an AP that contains two aggregation units, labeled as 1 and 2 in the figure, with the DONL field being present.



An Example of an AP Containing  
Two Aggregation Units with the DONL Field

Figure 8

### 4.3.3. Fragmentation Units

Fragmentation Units (FUs) are introduced to enable fragmenting a single NAL unit into multiple RTP packets, possibly without cooperation or knowledge of the [VVC] encoder. A fragment of a NAL unit consists of an integer number of consecutive octets of that NAL unit. Fragments of the same NAL unit MUST be sent in consecutive order with ascending RTP sequence numbers (with no other RTP packets within the same RTP stream being sent between the first and last fragment).

When a NAL unit is fragmented and conveyed within FUs, it is referred to as a fragmented NAL unit. APs MUST NOT be fragmented. FUs MUST NOT be nested; i.e., an FU can not contain a subset of another FU.

The RTP timestamp of an RTP packet carrying an FU is set to the NALU-time of the fragmented NAL unit.



When set to 1, the S bit indicates the start of a fragmented NAL unit, i.e., the first byte of the FU payload is also the first byte of the payload of the fragmented NAL unit. When the FU payload is not the start of the fragmented NAL unit payload, the S bit MUST be set to 0.

E: 1 bit

When set to 1, the E bit indicates the end of a fragmented NAL unit, i.e., the last byte of the payload is also the last byte of the fragmented NAL unit. When the FU payload is not the last fragment of a fragmented NAL unit, the E bit MUST be set to 0.

P: 1 bit

When set to 1, the P bit indicates the last FU of the last VCL NAL unit of a coded picture, i.e., the last byte of the FU payload is also the last byte of the last VCL NAL unit of the coded picture. When the FU payload is not the last fragment of the last VCL NAL unit of a coded picture, the P bit MUST be set to 0.

FuType: 5 bits

The field FuType MUST be equal to the field Type of the fragmented NAL unit.

The DONL field, when present, specifies the value of the 16 least significant bits of the decoding order number of the fragmented NAL unit.

If sprop-max-don-diff is greater than 0, and the S bit is equal to 1, the DONL field MUST be present in the FU, and the variable DON for the fragmented NAL unit is derived as equal to the value of the DONL field. Otherwise (sprop-max-don-diff is equal to 0, or the S bit is equal to 0), the DONL field MUST NOT be present in the FU.

A non-fragmented NAL unit MUST NOT be transmitted in one FU; i.e., the Start bit and End bit must not both be set to 1 in the same FU header.

The FU payload consists of fragments of the payload of the fragmented NAL unit so that if the FU payloads of consecutive FUs, starting with an FU with the S bit equal to 1 and ending with an FU with the E bit equal to 1, are sequentially concatenated, the payload of the fragmented NAL unit can be reconstructed. The NAL unit header of the fragmented NAL unit is not included as such in the FU payload, but rather the information of the NAL unit header of the fragmented NAL

unit is conveyed in F, LayerId, and TID fields of the FU payload headers of the FUs and the FuType field of the FU header of the FUs. An FU payload MUST NOT be empty.

If an FU is lost, the receiver SHOULD discard all following fragmentation units in transmission order corresponding to the same fragmented NAL unit, unless the decoder in the receiver is known to be prepared to gracefully handle incomplete NAL units.

A receiver in an endpoint or in a MANE MAY aggregate the first n-1 fragments of a NAL unit to an (incomplete) NAL unit, even if fragment n of that NAL unit is not received. In this case, the forbidden\_zero\_bit of the NAL unit MUST be set to 1 to indicate a syntax violation.

#### 4.4. Decoding Order Number

For each NAL unit, the variable AbsDon is derived, representing the decoding order number that is indicative of the NAL unit decoding order.

Let NAL unit n be the n-th NAL unit in transmission order within an RTP stream.

If sprop-max-don-diff is equal to 0, AbsDon[n], the value of AbsDon for NAL unit n, is derived as equal to n.

Otherwise (sprop-max-don-diff is greater than 0), AbsDon[n] is derived as follows, where DON[n] is the value of the variable DON for NAL unit n:

- \* If n is equal to 0 (i.e., NAL unit n is the very first NAL unit in transmission order), AbsDon[0] is set equal to DON[0].
- \* Otherwise (n is greater than 0), the following applies for derivation of AbsDon[n]:

```
If DON[n] == DON[n-1],
    AbsDon[n] = AbsDon[n-1]

If (DON[n] > DON[n-1] and DON[n] - DON[n-1] < 32768),
    AbsDon[n] = AbsDon[n-1] + DON[n] - DON[n-1]

If (DON[n] < DON[n-1] and DON[n-1] - DON[n] >= 32768),
    AbsDon[n] = AbsDon[n-1] + 65536 - DON[n-1] + DON[n]

If (DON[n] > DON[n-1] and DON[n] - DON[n-1] >= 32768),
    AbsDon[n] = AbsDon[n-1] - (DON[n-1] + 65536 - DON[n])

If (DON[n] < DON[n-1] and DON[n-1] - DON[n] < 32768),
    AbsDon[n] = AbsDon[n-1] - (DON[n-1] - DON[n])
```

For any two NAL units m and n, the following applies:

- \* AbsDon[n] greater than AbsDon[m] indicates that NAL unit n follows NAL unit m in NAL unit decoding order.
- \* When AbsDon[n] is equal to AbsDon[m], the NAL unit decoding order of the two NAL units can be in either order.
- \* AbsDon[n] less than AbsDon[m] indicates that NAL unit n precedes NAL unit m in decoding order.

Informative note: When two consecutive NAL units in the NAL unit decoding order have different values of AbsDon, the absolute difference between the two AbsDon values may be greater than or equal to 1.

Informative note: There are multiple reasons to allow for the absolute difference of the values of AbsDon for two consecutive NAL units in the NAL unit decoding order to be greater than one. An increment by one is not required, as at the time of associating values of AbsDon to NAL units, it may not be known whether all NAL units are to be delivered to the receiver. For example, a gateway might not forward VCL NAL units of higher sublayers or some SEI NAL units when there is congestion in the network. In another example, the first intra-coded picture of a pre-encoded clip is transmitted in advance to ensure that it is readily available in the receiver, and when transmitting the first intra-coded picture, the originator does not exactly know how many NAL units will be encoded before the first intra-coded picture of the pre-encoded clip follows in decoding order. Thus, the values of AbsDon for the NAL units of the first intra-coded picture of the pre-encoded clip have to be estimated when they are transmitted, and gaps in values of AbsDon may occur.

## 5. Packetization Rules

The following packetization rules apply:

- \* If `sprop-max-don-diff` is greater than 0, the transmission order of NAL units carried in the RTP stream MAY be different than the NAL unit decoding order. Otherwise (`sprop-max-don-diff` is equal to 0), the transmission order of NAL units carried in the RTP stream MUST be the same as the NAL unit decoding order.
- \* A NAL unit of a small size SHOULD be encapsulated in an aggregation packet together with one or more other NAL units in order to avoid the unnecessary packetization overhead for small NAL units. For example, non-VCL NAL units such as access unit delimiters, parameter sets, or SEI NAL units are typically small and can often be aggregated with VCL NAL units without violating MTU size constraints.
- \* Each non-VCL NAL unit SHOULD, when possible from an MTU size match viewpoint, be encapsulated in an aggregation packet together with its associated VCL NAL unit, as typically a non-VCL NAL unit would be meaningless without the associated VCL NAL unit being available.
- \* For carrying exactly one NAL unit in an RTP packet, a single NAL unit packet MUST be used.

## 6. De-packetization Process

The general concept behind de-packetization is to get the NAL units out of the RTP packets in an RTP stream and pass them to the decoder in the NAL unit decoding order.

The de-packetization process is implementation dependent. Therefore, the following description should be seen as an example of a suitable implementation. Other schemes may be used as well, as long as the output for the same input is the same as the process described below. The output is the same when the set of output NAL units and their order are both identical. Optimizations relative to the described algorithms are possible.

All normal RTP mechanisms related to buffer management apply. In particular, duplicated or outdated RTP packets (as indicated by the RTP sequence number and the RTP timestamp) are removed. To determine the exact time for decoding, factors such as a possible intentional delay to allow for proper inter-stream synchronization **MUST** be factored in.

NAL units with NAL unit type values in the range of 0 to 27, inclusive, may be passed to the decoder. NAL-unit-like structures with NAL unit type values in the range of 28 to 31, inclusive, **MUST NOT** be passed to the decoder.

The receiver includes a receiver buffer, which is used to compensate for transmission delay jitter within individual RTP stream, and to reorder NAL units from transmission order to the NAL unit decoding order. In this section, the receiver operation is described under the assumption that there is no transmission delay jitter within an RTP stream. To make a difference from a practical receiver buffer that is also used for compensation of transmission delay jitter, the receiver buffer is hereafter called the de-packetization buffer in this section. Receivers should also prepare for transmission delay jitter; that is, either reserve separate buffers for transmission delay jitter buffering and de-packetization buffering or use a receiver buffer for both transmission delay jitter and de-packetization. Moreover, receivers should take transmission delay jitter into account in the buffering operation, e.g., by additional initial buffering before starting of decoding and playback.

The de-packetization process extracts the NAL units from the RTP packets in an RTP stream as follows. When an RTP packet carries a single NAL unit packet, the payload of the RTP packet is extracted as a single NAL unit, excluding the DONL field, i.e., third and fourth bytes, when `sprop-max-don-diff` is greater than 0. When an RTP packet carries an Aggregation Packet, several NAL units are extracted from

the payload of the RTP packet. In this case, each NAL unit corresponds to the part of the payload of each aggregation unit that follows the NALU size field as described in Section 4.3.2. When an RTP packet carries a Fragmentation Unit (FU), all RTP packets from the first FU (with the S field equal to 1) of the fragmented NAL unit up to the last FU (with the E field equal to 1) of the fragmented NAL unit are collected. The NAL unit is extracted from these RTP packets by concatenating all FU payloads in the same order as the corresponding RTP packets and appending the NAL unit header with the fields F, LayerId, and TID, set to equal to the values of the fields F, LayerId, and TID in the payload header of the FUs respectively, and with the NAL unit type set equal to the value of the field FuType in the FU header of the FUs, as described in Section 4.3.3.

When `sprop-max-don-diff` is equal to 0, the de-packetization buffer size is zero bytes, and the NAL units carried in the single RTP stream are directly passed to the decoder in their transmission order, which is identical to their decoding order.

When `sprop-max-don-diff` is greater than 0, the process described in the remainder of this section applies.

There are two buffering states in the receiver: initial buffering and buffering while playing. Initial buffering starts when the reception is initialized. After initial buffering, decoding and playback are started, and the buffering-while-playing mode is used.

Regardless of the buffering state, the receiver stores incoming NAL units in reception order into the de-packetization buffer. NAL units carried in RTP packets are stored in the de-packetization buffer individually, and the value of `AbsDon` is calculated and stored for each NAL unit.

Initial buffering lasts until the difference between the greatest and smallest `AbsDon` values of the NAL units in the de-packetization buffer is greater than or equal to the value of `sprop-max-don-diff`.

After initial buffering, whenever the difference between the greatest and smallest `AbsDon` values of the NAL units in the de-packetization buffer is greater than or equal to the value of `sprop-max-don-diff`, the following operation is repeatedly applied until this difference is smaller than `sprop-max-don-diff`:

- \* The NAL unit in the de-packetization buffer with the smallest value of `AbsDon` is removed from the de-packetization buffer and passed to the decoder.

When no more NAL units are flowing into the de-packetization buffer, all NAL units remaining in the de-packetization buffer are removed from the buffer and passed to the decoder in the order of increasing AbsDon values.

## 7. Payload Format Parameters

This section specifies the optional parameters. A mapping of the parameters with Session Description Protocol (SDP) [RFC4556] is also provided for applications that use SDP.

Parameters starting with the string "sprop" for stream properties can be used by a sender to provide a receiver with the properties of the stream that is or will be sent. The media sender (and not the receiver) selects whether, and with what values, "sprop" parameters are being sent. This uncommon characteristic of the "sprop" parameters may not be intuitive in the context of some signaling protocol concepts, especially with offer/answer. Please see Section 7.3.2 for guidance specific to the use of sprop parameters in the Offer/Answer case.

### 7.1. Media Type Registration

The receiver MUST ignore any parameter unspecified in this memo.

Type name: video

Subtype name: H266

Required parameters: N/A

Optional parameters:

profile-id, tier-flag, sub-profile-id, interop-constraints, level-id, sprop-sublayer-id, sprop-ols-id, recv-sublayer-id, recv-ols-id, max-recv-level-id, sprop-dci, sprop-vps, sprop-sps, sprop-pps, sprop-sei, max-lsr, max-fps, sprop-max-don-diff, sprop-depack-buf-bytes, depack-buf-cap (Refer to Section 7.2 for definitions).

Encoding considerations:

This type is only defined for transfer via RTP (RFC 3550).

Security considerations:

See Section 9 of RFC XXXX.

Interoperability considerations: N/A

Published specification:

Please refer to RFC XXXX and VVC coding specification [VVC].

Applications that use this media type:

Any application that relies on VVC-based video services over RTP

Fragment identifier considerations: N/A

Additional information: N/A

Person & email address to contact for further information:

Stephan Wenger (stewe@stewe.org)

Intended usage: COMMON

Restrictions on usage: N/A

Author: See Authors' Addresses section of RFC XXXX.

Change controller:

IETF <avtcore@ietf.org>

## 7.2. Optional Parameters Definition

profile-id, tier-flag, sub-profile-id, interop-constraints, and level-id:

These parameters indicate the profile, tier, default level, sub-profile, and some constraints of the bitstream carried by the RTP stream, or a specific set of the profile, tier, default level, sub-profile and some constraints the receiver supports.

The subset of coding tools that may have been used to generate the bitstream or that the receiver supports, as well as some additional constraints are indicated collectively by profile-id, sub-profile-id, and interop-constraints.

Informative note: There are 128 values of profile-id. The subset of coding tools identified by the profile-id can be further constrained with up to 255 instances of sub-profile-id. In addition, 68 bits included in interop-constraints, which can be extended up to 324 bits provide means to further restrict tools from existing profiles. To be able to support this fine-granular signaling of coding tool subsets with profile-id, sub-

profile-id and interop-constraints, it would be safe to require symmetric use of these parameters in SDP offer/answer unless recv-ols-id is included in the SDP answer for choosing one of the layers offered.

The tier is indicated by tier-flag. The default level is indicated by level-id. The tier and the default level specify the limits on values of syntax elements or arithmetic combinations of values of syntax elements that are followed when generating the bitstream or that the receiver supports.

In SDP offer/answer, when the SDP answer does not include the recv-ols-id parameter that is less than the sprop-ols-id parameter in the SDP offer, the following applies:

- The tier-flag, profile-id, sub-profile-id, and interop-constraints parameters MUST be used symmetrically, i.e., the value of each of these parameters in the offer MUST be the same as that in the answer, either explicitly signaled or implicitly inferred.
- The level-id parameter is changeable as long as the highest level indicated by the answer is either equal to or lower than that in the offer. Note that a highest level higher than level-id in the offer for receiving can be included as max-recv-level-id.

In SDP offer/answer, when the SDP answer does include the recv-ols-id parameter that is less than the sprop-ols-id parameter in the SDP offer, the set of tier-flag, profile-id, sub-profile-id, interop-constraints, and level-id parameters included in the answer MUST be consistent with that for the chosen output layer set as indicated in the SDP offer, with the exception that the level-id parameter in the SDP answer is changeable as long as the highest level indicated by the answer is either lower than or equal to that in the offer.

More specifications of these parameters, including how they relate to syntax elements specified in [VVC] are provided below.

profile-id:

When profile-id is not present, a value of 1 (i.e., the Main 10 profile) MUST be inferred.

When used to indicate properties of a bitstream, profile-id is derived from the `general_profile_idc` syntax element that applies to the bitstream in an instance of the `profile_tier_level()` syntax structure.

VVC bitstreams transported over RTP using the technologies of this memo SHOULD contain only a single `profile_tier_level()` structure in the DCI, unless the sender can assure that a receiver can correctly decode the VVC bitstream regardless of which `profile_tier_level()` structure contained in the DCI was used for deriving profile-id and other parameters for the SDP O/A exchange.

As specified in [VVC], a `profile_tier_level()` syntax structure may be contained in an SPS NAL unit, and one or more `profile_tier_level()` syntax structures may be contained in a VPS NAL unit and in a DCI NAL unit. One of the following three cases applies to the container NAL unit of the `profile_tier_level()` syntax structure containing syntax elements used to derive the values of profile-id, tier-flag, level-id, sub-profile-id, or interop-constraints: 1) The container NAL unit is an SPS, the bitstream is a single-layer bitstream, and the `profile_tier_level()` syntax structures in all SPSs referenced by the CVSSs in the bitstream has the same values respectively for those `profile_tier_level()` syntax elements; 2) The container NAL unit is a VPS, the `profile_tier_level()` syntax structure is the one in the VPS that applies to the OLS corresponding to the bitstream, and the `profile_tier_level()` syntax structures applicable to the OLS corresponding to the bitstream in all VPSs referenced by the CVSSs in the bitstream have the same values respectively for those `profile_tier_level()` syntax elements; 3) The container NAL unit is a DCI NAL unit and the `profile_tier_level()` syntax structures in all DCI NAL units in the bitstream has the same values respectively for those `profile_tier_level()` syntax elements.

[VVC] allows for multiple `profile_tier_level()` structures in a DCI NAL unit, which may contain different values for the syntax elements used to derive the values of profile-id, tier-flag, level-id, sub-profile-id, or interop-constraints in the different entries. However, herein defined is only a single profile-id, tier-flag, level-id, sub-profile-id, or interop-constraints. When signaling these parameters and a DCI NAL unit is present with multiple `profile_tier_level()` structures, these values SHOULD be the same as the first `profile_tier_level` structure in the DCI, unless the sender has ensured that the receiver can decode the bitstream when a different value is chosen.

tier-flag, level-id:

The value of tier-flag MUST be in the range of 0 to 1, inclusive. The value of level-id MUST be in the range of 0 to 255, inclusive.

If the tier-flag and level-id parameters are used to indicate properties of a bitstream, they indicate the tier and the highest level the bitstream complies with.

If the tier-flag and level-id parameters are used for capability exchange, the following applies. If max-recv-level-id is not present, the default level defined by level-id indicates the highest level the codec wishes to support. Otherwise, max-recv-level-id indicates the highest level the codec supports for receiving. For either receiving or sending, all levels that are lower than the highest level supported MUST also be supported.

If no tier-flag is present, a value of 0 MUST be inferred; if no level-id is present, a value of 51 (i.e., level 3.1) MUST be inferred.

Informative note: The level values currently defined in the VVC specification are in the form of "majorNum.minorNum", and the value of the level-id for each of the levels is equal to  $\text{majorNum} * 16 + \text{minorNum} * 3$ . It is expected that if any levels are defined in the future, the same convention will be used, but this cannot be guaranteed.

When used to indicate properties of a bitstream, the tier-flag and level-id parameters are derived respectively from the syntax element `general_tier_flag`, and the syntax element `general_level_idc` or `sub_layer_level_idc[j]`, that apply to the bitstream, in an instance of the `profile_tier_level( )` syntax structure.

If the tier-flag and level-id are derived from the `profile_tier_level( )` syntax structure in a DCI NAL unit, the following applies:

- tier-flag = `general_tier_flag`
- level-id = `general_level_idc`

Otherwise, if the tier-flag and level-id are derived from the `profile_tier_level( )` syntax structure in an SPS or VPS NAL unit, and the bitstream contains the highest sublayer representation in the OLS corresponding to the bitstream, the following applies:

- tier-flag = `general_tier_flag`

- level-id = general\_level\_idc

Otherwise, if the tier-flag and level-id are derived from the profile\_tier\_level( ) syntax structure in an SPS or VPS NAL unit, and the bitstream does not contain the highest sublayer representation in the OLS corresponding to the bitstream, the following applies, with j being the value of the sprop-sublayer-id parameter:

- tier-flag = general\_tier\_flag
- level-id = sub\_layer\_level\_idc[j]

#### sub-profile-id:

The value of the parameter is a comma-separated (',' ) list of data using base64 encoding (Section 4 of [RFC4648]) representation without "==" padding.

When used to indicate properties of a bitstream, sub-profile-id is derived from each of the ptl\_num\_sub\_profiles general\_sub\_profile\_idc[i] syntax elements that apply to the bitstream in a profile\_tier\_level( ) syntax structure.

#### interop-constraints:

A base64 encoding (Section 4 of [RFC4648]) representation of the data that includes the syntax elements ptl\_frame\_only\_constraint\_flag and ptl\_multilayer\_enabled\_flag and the general\_constraints\_info( ) syntax structure that apply to the bitstream in an instance of the profile\_tier\_level( ) syntax structure.

If the interop-constraints parameter is not present, the following MUST be inferred:

- ptl\_frame\_only\_constraint\_flag = 1
- ptl\_multilayer\_enabled\_flag = 0
- gci\_present\_flag in the general\_constraints\_info( ) syntax structure = 0

Using interop-constraints for capability exchange results in a requirement on any bitstream to be compliant with the interop-constraints.

#### sprop-sublayer-id:

This parameter MAY be used to indicate the highest allowed value of TID in the bitstream. When not present, the value of sprop-sublayer-id is inferred to be equal to 6.

The value of sprop-sublayer-id MUST be in the range of 0 to 6, inclusive.

#### sprop-ols-id:

This parameter MAY be used to indicate the OLS that the bitstream applies to. When not present, the value of sprop-ols-id is inferred to be equal to TargetOlsIdx as specified in 8.1.1 in [VVC]. If this optional parameter is present, sprop-vps MUST also be present or its content MUST be known a priori at the receiver.

The value of sprop-ols-id MUST be in the range of 0 to 256, inclusive.

Informative note: VVC allows having up to 257 output layer sets indicated in the VPS as the number of output layer sets minus 2 is indicated with a field of 8 bits.

#### recv-sublayer-id:

This parameter MAY be used to signal a receiver's choice of the offered or declared sublayer representations in the sprop-vps and sprop-sps. The value of recv-sublayer-id indicates the TID of the highest sublayer that a receiver supports. When not present, the value of recv-sublayer-id is inferred to be equal to the value of the sprop-sublayer-id parameter in the SDP offer.

The value of recv-sublayer-id MUST be in the range of 0 to 6, inclusive.

#### recv-ols-id:

This parameter MAY be used to signal a receiver's choice of the offered or declared output layer sets in the sprop-vps. The value of recv-ols-id indicates the OLS index of the bitstream that a receiver supports. When not present, the value of recv-ols-id is inferred to be equal to value of the sprop-ols-id parameter inferred from or indicated in the SDP offer. When present, the value of recv-ols-id must be included only when sprop-ols-id was received and must refer to an output layer set in the VPS that includes no layers other than all or a subset of the layers of the OLS referred to by sprop-ols-id. If this optional parameter is present, sprop-vps must have been received or its content must be known a priori at the receiver.

The value of `recv-ols-id` MUST be in the range of 0 to 256, inclusive.

`max-recv-level-id`:

This parameter MAY be used to indicate the highest level a receiver supports.

The value of `max-recv-level-id` MUST be in the range of 0 to 255, inclusive.

When `max-recv-level-id` is not present, the value is inferred to be equal to `level-id`.

`max-recv-level-id` MUST NOT be present when the highest level the receiver supports is not higher than the default level.

`sprop-dci`:

This parameter MAY be used to convey a decoding capability information NAL unit of the bitstream for out-of-band transmission. The parameter MAY also be used for capability exchange. The value of the parameter is a base64 encoding (Section 4 of [RFC4648]) representations of the decoding capability information NAL unit as specified in Section 7.3.2.1 of [VVC].

`sprop-vps`:

This parameter MAY be used to convey any video parameter set NAL unit of the bitstream for out-of-band transmission of video parameter sets. The parameter MAY also be used for capability exchange and to indicate sub-stream characteristics (i.e., properties of output layer sets and sublayer representations as defined in [VVC]). The value of the parameter is a comma-separated (' , ') list of base64 encoding (Section 4 of [RFC4648]) representations of the video parameter set NAL units as specified in Section 7.3.2.3 of [VVC].

The `sprop-vps` parameter MAY contain one or more than one video parameter set NAL units. However, all other video parameter sets contained in the `sprop-vps` parameter MUST be consistent with the first video parameter set in the `sprop-vps` parameter. A video parameter set `vpsB` is said to be consistent with another video parameter set `vpsA` if the number of OLSs in `vpsA` and `vpsB` is the same and any decoder that conforms to the profile, tier, level, and constraints indicated by the data starting from the syntax element `general_profile_idc` to the syntax structure `general_constraints_info()`, inclusive, in the `profile_tier_level()`

) syntax structure corresponding to any OLS with index `olsIdx` in `vpsA` can decode any CVS(s) referencing `vpsB` when `TargetOlsIdx` is equal to `olsIdx` that conforms to the profile, tier, level, and constraints indicated by the data starting from the syntax element `general_profile_idc` to the syntax structure `general_constraints_info()`, inclusive, in the `profile_tier_level( )` syntax structure corresponding to the OLS with index `TargetOlsIdx` in `vpsB`.

#### `sprop-sps:`

This parameter MAY be used to convey sequence parameter set NAL units of the bitstream for out-of-band transmission of sequence parameter sets. The value of the parameter is a comma-separated (' , ') list of base64 encoding (Section 4 of [RFC4648]) representations of the sequence parameter set NAL units as specified in Section 7.3.2.4 of [VVC].

A sequence parameter set `spsB` is said to be consistent with another sequence parameter set `spsA` if any decoder that conforms to the profile, tier, level, and constraints indicated by the data starting from the syntax element `general_profile_idc` to the syntax structure `general_constraints_info()`, inclusive, in the `profile_tier_level( )` syntax structure in `spsA` can decode any CLVS(s) referencing `spsB` that conforms to the profile, tier, level, and constraints indicated by the data starting from the syntax element `general_profile_idc` to the syntax structure `general_constraints_info()`, inclusive, in the `profile_tier_level( )` syntax structure in `spsB`.

#### `sprop-pps:`

This parameter MAY be used to convey picture parameter set NAL units of the bitstream for out-of-band transmission of picture parameter sets. The value of the parameter is a comma-separated (' , ') list of base64 encoding (Section 4 of [RFC4648]) representations of the picture parameter set NAL units as specified in Section 7.3.2.5 of [VVC].

#### `sprop-sei:`

This parameter MAY be used to convey one or more SEI messages that describe bitstream characteristics. When present, a decoder can rely on the bitstream characteristics that are described in the SEI messages for the entire duration of the session, independently from the persistence scopes of the SEI messages as specified in [VSEI].

The value of the parameter is a comma-separated (',' ) list of base64 encoding (Section 4 of [RFC4648]) representations of SEI NAL units as specified in [VSEI].

Informative note: Intentionally, no list of applicable or inapplicable SEI messages is specified here. Conveying certain SEI messages in sprop-sei may be sensible in some application scenarios and meaningless in others. However, a few examples are described below:

- 1) In an environment where the bitstream was created from film-based source material, and no splicing is going to occur during the lifetime of the session, the film grain characteristics SEI message is likely meaningful, and sending it in sprop-sei rather than in the bitstream at each entry point may help with saving bits and allows one to configure the renderer only once, avoiding unwanted artifacts.
- 2) Examples for SEI messages that would be meaningless to be conveyed in sprop-sei include the decoded picture hash SEI message (it is close to impossible that all decoded pictures have the same hashtag) or the filler payload SEI message (as there is no point in just having more bits in SDP).

max-lsr:

The max-lsr MAY be used to signal the capabilities of a receiver implementation and MUST NOT be used for any other purpose. The value of max-lsr is an integer indicating the maximum processing rate in units of luma samples per second. The max-lsr parameter signals that the receiver is capable of decoding video at a higher rate than is required by the highest level.

Informative note: When the OPTIONAL media type parameters are used to signal the properties of a bitstream, and max-lsr is not present, the values of tier-flag, profile-id, sub-profile-id interop-constraints, and level-id must always be such that the bitstream complies fully with the specified profile, tier, and level.

When max-lsr is signaled, the receiver MUST be able to decode bitstreams that conform to the highest level, with the exception that the MaxLumaSr value in Table 136 of [VVC] for the highest level is replaced with the value of max-lsr. Senders MAY use this knowledge to send pictures of a given size at a higher picture rate than is indicated in the highest level.

When not present, the value of max-lsr is inferred to be equal to the value of MaxLumaSr given in Table 136 of [VVC] for the highest level.

The value of max-lsr MUST be in the range of MaxLumaSr to  $16 * \text{MaxLumaSr}$ , inclusive, where MaxLumaSr is given in Table 136 of [VVC] for the highest level.

#### max-fps:

The value of max-fps is an integer indicating the maximum picture rate in units of pictures per 100 seconds that can be effectively processed by the receiver. The max-fps parameter MAY be used to signal that the receiver has a constraint in that it is not capable of processing video effectively at the full picture rate that is implied by the highest level and, when present, max-lsr.

The value of max-fps is not necessarily the picture rate at which the maximum picture size can be sent, it constitutes a constraint on maximum picture rate for all resolutions.

Informative note: The max-fps parameter is semantically different from max-lsr in that max-fps is used to signal a constraint, lowering the maximum picture rate from what is implied by other parameters.

The encoder MUST use a picture rate equal to or less than this value. In cases where the max-fps parameter is absent, the encoder is free to choose any picture rate according to the highest level and any signaled optional parameters.

The value of max-fps MUST be smaller than or equal to the full picture rate that is implied by the highest level and, when present, max-lsr.

#### sprop-max-don-diff:

If there is no NAL unit naluA that is followed in transmission order by any NAL unit preceding naluA in decoding order (i.e., the transmission order of the NAL units is the same as the decoding order), the value of this parameter MUST be equal to 0.

Otherwise, this parameter specifies the maximum absolute difference between the decoding order number (i.e., AbsDon) values of any two NAL units naluA and naluB, where naluA follows naluB in decoding order and precedes naluB in transmission order.

The value of `sprop-max-don-diff` MUST be an integer in the range of 0 to 32767, inclusive.

When not present, the value of `sprop-max-don-diff` is inferred to be equal to 0.

`sprop-depack-buf-bytes`:

This parameter signals the required size of the de-packetization buffer in units of bytes. The value of the parameter MUST be greater than or equal to the maximum buffer occupancy (in units of bytes) of the de-packetization buffer as specified in Section 6.

The value of `sprop-depack-buf-bytes` MUST be an integer in the range of 0 to 4294967295, inclusive.

When `sprop-max-don-diff` is present and greater than 0, this parameter MUST be present and the value MUST be greater than 0. When not present, the value of `sprop-depack-buf-bytes` is inferred to be equal to 0.

Informative note: The value of `sprop-depack-buf-bytes` indicates the required size of the de-packetization buffer only. When network jitter can occur, an appropriately sized jitter buffer has to be available as well.

`depack-buf-cap`:

This parameter signals the capabilities of a receiver implementation and indicates the amount of de-packetization buffer space in units of bytes that the receiver has available for reconstructing the NAL unit decoding order from NAL units carried in the RTP stream. A receiver is able to handle any RTP stream for which the value of the `sprop-depack-buf-bytes` parameter is smaller than or equal to this parameter.

When not present, the value of `depack-buf-cap` is inferred to be equal to 4294967295. The value of `depack-buf-cap` MUST be an integer in the range of 1 to 4294967295, inclusive.

Informative note: `depack-buf-cap` indicates the maximum possible size of the de-packetization buffer of the receiver only, without allowing for network jitter.

### 7.3. SDP Parameters

The receiver MUST ignore any parameter unspecified in this memo.

### 7.3.1. Mapping of Payload Type Parameters to SDP

The media type video/H266 string is mapped to fields in the Session Description Protocol (SDP) [RFC8866] as follows:

- \* The media name in the "m=" line of SDP MUST be video.
- \* The encoding name in the "a=rtpmap" line of SDP MUST be H266 (the media subtype).
- \* The clock rate in the "a=rtpmap" line MUST be 90000.
- \* The OPTIONAL parameters profile-id, tier-flag, sub-profile-id, interop-constraints, level-id, sprop-sublayer-id, sprop-ols-id, recv-sublayer-id, recv-ols-id, max-recv-level-id, max-lsr, max-fps, sprop-max-don-diff, sprop-depack-buf-bytes and depack-buf-cap, when present, MUST be included in the "a=fmtp" line of SDP. The fmtp line is expressed as a media type string, in the form of a semicolon-separated list of parameter=value pairs.
- \* The OPTIONAL parameter sprop-vps, sprop-sps, sprop-pps, sprop-sei, and sprop-dci, when present, MUST be included in the "a=fmtp" line of SDP or conveyed using the "fmtp" source attribute as specified in Section 6.3 of [RFC5576]. For a particular media format (i.e., RTP payload type), sprop-vps, sprop-sps, sprop-pps, sprop-sei, or sprop-dci MUST NOT be both included in the "a=fmtp" line of SDP and conveyed using the "fmtp" source attribute. When included in the "a=fmtp" line of SDP, those parameters are expressed as a media type string, in the form of a semicolon-separated list of parameter=value pairs. When conveyed in the "a=fmtp" line of SDP for a particular payload type, the parameters sprop-vps, sprop-sps, sprop-pps, sprop-sei, and sprop-dci MUST be applied to each SSRC with the payload type. When conveyed using the "fmtp" source attribute, these parameters are only associated with the given source and payload type as parts of the "fmtp" source attribute.

Informative note: Conveyance of sprop-vps, sprop-sps, and sprop-pps using the "fmtp" source attribute allows for out-of-band transport of parameter sets in topologies like Topo-Video-switch-MCU as specified in [RFC7667]

An general usage of media representation in SDP is as follows:

```
m=video 49170 RTP/AVP 98
a=rtpmap:98 H266/90000
a=fmtp:98 profile-id=1;
  sprop-vps=<video parameter sets data>;
  sprop-sps=<sequence parameter set data>;
  sprop-pps=<picture parameter set data>;
```

A SIP Offer/Answer exchange wherein both parties are expected to both send and receive could look like the following. Only the media codec-specific parts of the SDP are shown. Some lines are wrapped due to text constraints.

Offerer->Answerer:

```
m=video 49170 RTP/AVP 98
a=rtpmap:98 H266/90000
a=fmtp:98 profile-id=1; level_id=83;
```

The above represents an offer for symmetric video communication using [VVC] and it's payload specification, at the main profile and level 5.1 (and, as the levels are downgradable, all lower levels. Informally speaking, this offer tells the receiver of the offer that the sender is willing to receive up to 4Kp60 resolution at the maximum bitrates specified in [VVC]. At the same time, if this offer were accepted "as is", the offer can expect that the answerer would be able to receive and properly decode H.266 media up to and including level 5.1.

Answerer->Offerer:

```
m=video 49170 RTP/AVP 98
a=rtpmap:98 H266/90000
a=fmtp:98 profile-id=1; level_id=67
```

With this answer to the offer above, the system receiving the offer advises the offerer that it is incapable of handling H.266 at level 5.1 but is capable of decoding 1080p60. As H.266 video codecs must support decoding at all levels below the maximum level they implement, the resulting user experience would likely be that both systems send video at 1080p60. However, nothing prevents an encoder from further downgrading its sending to, for example 720p30 if it were short of cycles, bandwidth, or for other reasons.

### 7.3.2. Usage with SDP Offer/Answer Model

This section describes the negotiation of unicast messages using the offer-answer model as described in [RFC3264] and its updates. The section is split into subsections, covering a) media format configurations not involving non-temporal scalability; b) scalable media format configurations; c) the description of the use of those parameters not involving the media configuration itself but rather the parameters of the payload format design; and d) multicast.

#### 7.3.2.1. Non-scalable media format configuration

A non-scalable VVC media configuration is such a configuration where no non-temporal scalability mechanisms are allowed. In [VVC] version 1, that implies that `general_profile_idc` indicates one of the following profiles: Main10, Main10 Still Picture, Main 10 4:4:4, Main10 4:4:4 Still Picture, with `general_profile_idc` values of 1, 65, 33, and 97, respectively. Note that non-scalable media configurations includes temporal scalability, inline with VVC's design philosophy and profile structure.

The following limitations and rules pertaining to the media configuration apply:

- \* The parameters identifying a media format configuration for VVC are `profile-id`, `tier-flag`, `sub-profile-id`, `level-id`, and `interop-constraints`. These media configuration parameters, except `level-id`, MUST be used symmetrically.

The answerer MUST structure its answer in according to one of the following three options:

1) maintain all configuration parameters with the values remaining the same as in the offer for the media format (payload type), with the exception that the value of `level-id` is changeable as long as the highest level indicated by the answer is not higher than that indicated by the offer;

2) include in the answer the `recv-sublayer-id` parameter, with a value less than the `sprop-sublayer-id` parameter in the offer, for the media format (payload type), and maintain all configuration parameters with the values remaining the same as in the offer for the media format (payload type), with the exception that the value of `level-id` is changeable as long as the highest level indicated by the answer is not higher than the level indicated by the `sprop-sps` or `sprop-vps` in offer for the chosen sublayer representation; or

3) remove the media format (payload type) completely (when one or more of the parameter values are not supported).

Informative note: The above requirement for symmetric use does not apply for level-id, and does not apply for the other bitstream or RTP stream properties and capability parameters as described in Section 7.3.2.3 below.

- \* To simplify handling and matching of these configurations, the same RTP payload type number used in the offer SHOULD also be used in the answer, as specified in [RFC3264].
- \* The same RTP payload type number used in the offer for the media subtype H266 MUST be used in the answer when the answer includes recv-sublayer-id. When the answer does not include recv-sublayer-id, the answer MUST NOT contain a payload type number used in the offer for the media subtype H266 unless the configuration is exactly the same as in the offer or the configuration in the answer only differs from that in the offer with a different value of level-id. The answer MAY contain the recv-sublayer-id parameter if an VVC bitstream contains multiple operation points (using temporal scalability and sublayers) and sprop-sps or sprop-vps is included in the offer where information of sublayers are present in the first sequence parameter set or video parameter set contained in sprop-sps or sprop-vps respectively. If the sprop-sps or sprop-vps is provided in an offer, an answerer MAY select a particular operation point indicated in the first sequence parameter set or video parameter set contained in sprop-sps or sprop-vps respectively. When the answer includes a recv-sublayer-id that is less than a sprop-sublayer-id in the offer, the following applies:
  - 1) When sprop-sps parameter is present, all sequence parameter sets contained in the sprop-sps parameter in the SDP answer and all sequence parameter sets sent in-band for either the offerer-to-answerer direction or the answerer-to-offerer direction MUST be consistent with the first sequence parameter set in the sprop-sps parameter of the offer (see the semantics of sprop-sps in Section 7.1 of this document on one sequence parameter set being consistent with another sequence parameter set).

2) When sprop-vps parameter is present, all video parameter sets contained in the sprop-vps parameter in the SDP answer and all video parameter sets sent in-band for either the offerer-to-answerer direction or the answerer-to-offerer direction MUST be consistent with the first video parameter set in the sprop-vps parameter of the offer (see the semantics of sprop-vps in Section 7.1 of this document on one video parameter set being consistent with another video parameter set).

3) The bitstream sent in either direction MUST conform to the profile, tier, level, and constraints of the chosen sublayer representation as indicated by the profile\_tier\_level( ) syntax structure in the first sequence parameter set in the sprop-sps parameter or by the first profile\_tier\_level( ) syntax structure in the first video parameter set in the sprop-vps parameter of the offer.

Informative note: When an offerer receives an answer that does not include recv-sublayer-id, it has to compare payload types not declared in the offer based on the media type (i.e., video/H266) and the above media configuration parameters with any payload types it has already declared. This will enable it to determine whether the configuration in question is new or if it is equivalent to configuration already offered, since a different payload type number may be used in the answer. The ability to perform operation point selection enables a receiver to utilize the temporal scalable nature of an VVC bitstream.

#### 7.3.2.2. Scalable media format configuration

A scalable VVC media configuration is such a configuration where non-temporal scalability mechanisms are allowed. In [VVC] version 1, that implies that general\_profile\_idc indicates one of the following profiles: Multilayer Main 10, and Multilayer Main 10 4:4:4, with general\_profile\_idc values of 17 and 49, respectively.

The following limitations and rules pertaining to the media configuration apply. They are listed in an order that would be logical for an implementation to follow:

- \* The parameters identifying a media format configuration for scalable VVC are profile-id, tier-flag, sub-profile-id, level-id, interop-constraints, and sprop-vps. These media configuration parameters, except level-id, MUST be used symmetrically, except as noted below.

- \* The answerer MAY include a level-id that MUST be lower than or equal to the level-id indicated in the offer (either expressed by level-id in the offer, or implied by the default level as specific in Section 7.1).
- \* When sprop-ols-id is present in an offer, sprop-vps MUST also be present in the same offer and including at least one valid VPS, so to allow the answerer to meaningfully interpret sprop-ols-id and select recv-ols-id (see below).
- \* The answerer MUST NOT include recv-ols-id unless the offer includes sprop-ols-id. When present, recv-ols-id MUST indicate a supported output layer set in the VPS that includes no layers other than all or a subset of the layers of the OLS referred to by sprop-ols-id. If unable, the answerer MUST remove the media format.

Informative note: if an offerer wants to offer more than one output layer set, it can do so by offering multiple VVC media with different payload types.

- \* The offerer MAY include sprop-sublayer-id which indicates the highest allowed value of TID in the bitstream. The answerer MAY include recv-sublayer-id which can be used to reduce the number of sublayers from the value of sprop-sublayer-id.
- \* When the answerer includes recv-ols-id and configuration parameters profile-id, tier-flag, sub-profile-id, level-id, and interop-constraints, it MUST use the configuration parameter values as signaled in the sprop-vps for the operating point with the largest number of sublayers for the chosen output layer set, with the exception that the value of level-id is changeable as long as the highest level indicated by the answer is not higher than the level indicated by the sprop-vps in offer for the operating point with the largest number of sublayers for the chosen output layer set.

#### 7.3.2.3. Payload format configuration

The following limitations and rules pertain to the configuration of the payload format buffer management mostly and apply to both scalable and non-scalable VVC.

- \* The parameters sprop-max-don-diff, and sprop-depack-buf-bytes describe the properties of an RTP stream that the offerer or the answerer is sending for the media format configuration. This differs from the normal usage of the offer/answer parameters: normally such parameters declare the properties of the bitstream

or RTP stream that the offerer or the answerer is able to receive. When dealing with VVC, the offerer assumes that the answerer will be able to receive media encoded using the configuration being offered.

Informative note: The above parameters apply for any RTP stream, when present, sent by a declaring entity with the same configuration. In other words, the applicability of the above parameters to RTP streams depends on the source endpoint. Rather than being bound to the payload type, the values may have to be applied to another payload type when being sent, as they apply for the configuration.

- \* The capability parameter `max-lsr` MAY be used to declare further capabilities of the offerer or answerer for receiving. It MUST NOT be present when the direction attribute is `sendonly`.
- \* The capability parameter `max-fps` MAY be used to declare lower capabilities of the offerer or answerer for receiving. It MUST NOT be present when the direction attribute is `sendonly`.
- \* When an offerer offers an interleaved stream, indicated by the presence of `sprop-max-don-diff` with a value larger than zero, the offerer MUST include the size of the de-packetization buffer `sprop-depack-buf-bytes`.
- \* To enable the offerer and answerer to inform each other about their capabilities for de-packetization buffering in receiving RTP streams, both parties are RECOMMENDED to include `depack-buf-cap`.
- \* The `sprop-dci`, `sprop-vps`, `sprop-sps`, or `sprop-pps`, when present (included in the `"a=fmtp"` line of SDP or conveyed using the `"fmtp"` source attribute as specified in Section 6.3 of [RFC5576]), are used for out-of-band transport of the parameter sets (DCI, VPS, SPS, or PPS, respectively).
- \* The answerer MAY use either out-of-band or in-band transport of parameter sets for the bitstream it is sending, regardless of whether out-of-band parameter sets transport has been used in the offerer-to-answerer direction. Parameter sets included in an answer are independent of those parameter sets included in the offer, as they are used for decoding two different bitstreams, one from the answerer to the offerer and the other in the opposite direction. In case some RTP packets are sent before the SDP offer/answer settles down, in-band parameter sets MUST be used for those RTP stream parts sent before the SDP offer/answer.

- \* The following rules apply to transport of parameter set in the offerer-to-answerer direction.
  - An offer MAY include sprop-dci, sprop-vps, sprop-sps, and/or sprop-pps. If none of these parameters is present in the offer, then only in-band transport of parameter sets is used.
  - If the level to use in the offerer-to-answerer direction is equal to the default level in the offer, the answerer MUST be prepared to use the parameter sets included in sprop-vps, sprop-sps, and sprop-pps (either included in the "a=fmtp" line of SDP or conveyed using the "fmtp" source attribute) for decoding the incoming bitstream, e.g., by passing these parameter set NAL units to the video decoder before passing any NAL units carried in the RTP streams. Otherwise, the answerer MUST ignore sprop-vps, sprop-sps, and sprop-pps (either included in the "a=fmtp" line of SDP or conveyed using the "fmtp" source attribute) and the offerer MUST transmit parameter sets in-band.
- \* The following rules apply to transport of parameter set in the answerer-to-offerer direction.
  - An answer MAY include sprop-dci, sprop-vps, sprop-sps, and/or sprop-pps. If none of these parameters is present in the answer, then only in-band transport of parameter sets is used.
  - The offerer MUST be prepared to use the parameter sets included in sprop-vps, sprop-sps, and sprop-pps (either included in the "a=fmtp" line of SDP or conveyed using the "fmtp" source attribute) for decoding the incoming bitstream, e.g., by passing these parameter set NAL units to the video decoder before passing any NAL units carried in the RTP streams.
- \* When sprop-dci, sprop-vps, sprop-sps, and/or sprop-pps are conveyed using the "fmtp" source attribute as specified in Section 6.3 of [RFC5576], the receiver of the parameters MUST store the parameter sets included in sprop-dci, sprop-vps, sprop-sps, and/or sprop-pps and associate them with the source given as part of the "fmtp" source attribute. Parameter sets associated with one source (given as part of the "fmtp" source attribute) MUST only be used to decode NAL units conveyed in RTP packets from the same source (given as part of the "fmtp" source attribute). When this mechanism is in use, SSRC collision detection and resolution MUST be performed as specified in [RFC5576].

Table 1 lists the interpretation of all the parameters that MAY be used for the various combinations of offer, answer, and direction attributes. Note that the two columns wherein the `recv-ols-id` parameter is used only apply to answers, whereas the other columns apply to both offers and answers.

	sendonly --+				
	answer: recvonly, recv-ols-id --+				
	recvonly w/o recv-ols-id --+				
	answer: sendrecv, recv-ols-id --+				
	sendrecv w/o recv-ols-id --+				
profile-id	C	D	C	D	P
tier-flag	C	D	C	D	P
level-id	D	D	D	D	P
sub-profile-id	C	D	C	D	P
interop-constraints	C	D	C	D	P
max-recv-level-id	R	R	R	R	-
sprop-max-don-diff	P	P	-	-	P
sprop-depack-buf-bytes	P	P	-	-	P
depack-buf-cap	R	R	R	R	-
max-lsr	R	R	R	R	-
max-fps	R	R	R	R	-
sprop-dci	P	P	-	-	P
sprop-sei	P	P	-	-	P
sprop-vps	P	P	-	-	P
sprop-sps	P	P	-	-	P
sprop-pps	P	P	-	-	P
sprop-sublayer-id	P	P	-	-	P
recv-sublayer-id	O	O	O	O	-
sprop-ols-id	P	P	-	-	P
recv-ols-id	X	O	X	O	-

Table 1. Interpretation of parameters for various combinations of offers, answers, direction attributes, with and without recv-ols-id. Columns that do not indicate offer or answer apply to both.

Legend:

- C: configuration for sending and receiving bitstreams
- D: changeable configuration, same as C except possible to answer with a different but consistent value (see the semantics of the six parameters related to profile, tier, and level on these parameters being consistent)
- P: properties of the bitstream to be sent
- R: receiver capabilities
- O: operation point selection
- X: MUST NOT be present
- : not usable, when present MUST be ignored

Parameters used for declaring receiver capabilities are, in general, downgradable; i.e., they express the upper limit for a sender's possible behavior. Thus, a sender MAY select to set its encoder using only lower/lesser or equal values of these parameters.

When the answer does not include a `recv-ols-id` that is less than the `sprop-ols-id` in the offer, parameters declaring a configuration point are not changeable, with the exception of the `level-id` parameter for unicast usage, and these parameters express values a receiver expects to be used and MUST be used verbatim in the answer as in the offer.

When a sender's capabilities are declared with the configuration parameters, these parameters express a configuration that is acceptable for the sender to receive bitstreams. In order to achieve high interoperability levels, it is often advisable to offer multiple alternative configurations. It is impossible to offer multiple configurations in a single payload type. Thus, when multiple configuration offers are made, each offer requires its own RTP payload type associated with the offer. However, it is possible to offer multiple operation points using one configuration in a single payload type by including `sprop-vps` in the offer and `recv-ols-id` in the answer.

An implementation SHOULD be able to understand all media type parameters (including all optional media type parameters), even if it doesn't support the functionality related to the parameter. This, in conjunction with proper application logic in the implementation allows the implementation, after having received an offer, to create an answer by potentially downgrading one or more of the optional parameters to the point where the implementation can cope, leading to higher chances of interoperability beyond the most basic interop points (for which, as described above, no optional parameters are necessary).

Informative note: in implementations of previous H.26x payload formats it was occasionally observed that implementations were incapable of parsing most (or all) of the optional parameters. As a result, the offer-answer exchange resulted in a baseline performance (using the default values for the optional parameters) with the resulting suboptimal user experience. However, there are valid reasons to forego the implementation complexity of implementing the parsing of some or all of the optional parameters, for example, when there is pre-determined knowledge, not negotiated by an SDP-based offer/answer process, of the capabilities of the involved systems (walled gardens, baseline requirements defined in application standards higher up in the stack, and similar).

An answerer MAY extend the offer with additional media format configurations. However, to enable their usage, in most cases a second offer is required from the offerer to provide the bitstream property parameters that the media sender will use. This also has the effect that the offerer has to be able to receive this media format configuration, not only to send it.

### 7.3.3. Multicast

For bitstreams being delivered over multicast, the following rules apply:

- \* The media format configuration is identified by profile-id, tier-flag, sub-profile-id, level-id, and interop-constraints. These media format configuration parameters, including level-id, MUST be used symmetrically; that is, the answerer MUST either maintain all configuration parameters or remove the media format (payload type) completely. Note that this implies that the level-id for offer/answer in multicast is not changeable.
- \* To simplify the handling and matching of these configurations, the same RTP payload type number used in the offer SHOULD also be used in the answer, as specified in [RFC3264]. An answer MUST NOT contain a payload type number used in the offer unless the configuration is the same as in the offer.
- \* Parameter sets received MUST be associated with the originating source and MUST only be used in decoding the incoming bitstream from the same source.
- \* The rules for other parameters are the same as above for unicast as long as the three above rules are obeyed.

### 7.3.4. Usage in Declarative Session Descriptions

When VVC over RTP is offered with SDP in a declarative style, as in Real Time Streaming Protocol (RTSP) [RFC7826] or Session Announcement Protocol (SAP) [RFC2974], the following considerations are necessary.

- \* All parameters capable of indicating both bitstream properties and receiver capabilities are used to indicate only bitstream properties. For example, in this case, the parameter profile-id, tier-id, level-id declares the values used by the bitstream, not the capabilities for receiving bitstreams. As a result, the following interpretation of the parameters MUST be used:
  - Declaring actual configuration or bitstream properties:

- o profile-id
  - o tier-flag
  - o level-id
  - o interop-constraints
  - o sub-profile-id
  - o sprop-dci
  - o sprop-vps
  - o sprop-sps
  - o sprop-pps
  - o sprop-max-don-diff
  - o sprop-depack-buf-bytes
  - o sprop-sublayer-id
  - o sprop-ols-id
  - o sprop-sei
- Not usable (when present, they MUST be ignored):
- o max-lsr
  - o max-fps
  - o max-recv-level-id
  - o depack-buf-cap
  - o recv-sublayer-id
  - o recv-ols-id
- A receiver of the SDP is required to support all parameters and values of the parameters provided; otherwise, the receiver MUST reject (RTSP) or not participate in (SAP) the session. It falls on the creator of the session to use values that are expected to be supported by the receiving application.

### 7.3.5. Considerations for Parameter Sets

When out-of-band transport of parameter sets is used, parameter sets MAY still be additionally transported in-band unless explicitly disallowed by an application, and some of these additional parameter sets may update some of the out-of-band transported parameter sets. Update of a parameter set refers to the sending of a parameter set of the same type using the same parameter set ID but with different values for at least one other parameter of the parameter set.

## 8. Use with Feedback Messages

The following subsections define the use of the Picture Loss Indication (PLI) and Full Intra Request (FIR) feedback messages with [VVC]. The PLI is defined in [RFC4585], and the FIR message is defined in [RFC5104]. In accordance with this memo, unlike [HEVC], a sender MUST NOT send Slice Loss Indication (SLI) or Reference Picture Selection Indication (RPSI), and a receiver SHOULD ignore RPSI and treat a received SLI as a PLI.

### 8.1. Picture Loss Indication (PLI)

As specified in RFC 4585, Section 6.3.1, the reception of a PLI by a media sender indicates "the loss of an undefined amount of coded video data belonging to one or more pictures". Without having any specific knowledge of the setup of the bitstream (such as use and location of in-band parameter sets, non-IRAP decoder refresh points, picture structures, and so forth), a reaction to the reception of an PLI by a VVC sender SHOULD be to send an IRAP picture and relevant parameter sets; potentially with sufficient redundancy so to ensure correct reception. However, sometimes information about the bitstream structure is known. For example, state could have been established outside of the mechanisms defined in this document that parameter sets are conveyed out of band only, and stay static for the duration of the session. In that case, it is obviously unnecessary to send them in-band as a result of the reception of a PLI. Other examples could be devised based on a priori knowledge of different aspects of the bitstream structure. In all cases, the timing and congestion control mechanisms of RFC 4585 MUST be observed.

### 8.2. Full Intra Request (FIR)

The purpose of the FIR message is to force an encoder to send an independent decoder refresh point as soon as possible, while observing applicable congestion-control-related constraints, such as those set out in [RFC8082]).

Upon reception of a FIR, a sender MUST send an IDR picture. Parameter sets MUST also be sent, except when there is a priori knowledge that the parameter sets have been correctly established. A typical example for that is an understanding between sender and receiver, established by means outside this document, that parameter sets are exclusively sent out-of-band.

## 9. Security Considerations

The scope of this Security Considerations section is limited to the payload format itself and to one feature of [VVC] that may pose a particularly serious security risk if implemented naively. The payload format, in isolation, does not form a complete system. Implementers are advised to read and understand relevant security-related documents, especially those pertaining to RTP (see the Security Considerations section in [RFC3550]), and the security of the call-control stack chosen (that may make use of the media type registration of this memo). Implementers should also consider known security vulnerabilities of video coding and decoding implementations in general and avoid those.

Within this RTP payload format, and with the exception of the user data SEI message as described below, no security threats other than those common to RTP payload formats are known. In other words, neither the various media-plane-based mechanisms, nor the signaling part of this memo, seems to pose a security risk beyond those common to all RTP-based systems.

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. However, as "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format's responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. The rest of this section discusses the security impacting properties of the payload format itself.

Because the data compression used with this payload format is applied end-to-end, any encryption needs to be performed after compression. A potential denial-of-service threat exists for data encodings using compression techniques that have non-uniform receiver-end computational load. The attacker can inject pathological datagrams

into the bitstream that are complex to decode and that cause the receiver to be overloaded. [VVC] is particularly vulnerable to such attacks, as it is extremely simple to generate datagrams containing NAL units that affect the decoding process of many future NAL units. Therefore, the usage of data origin authentication and data integrity protection of at least the RTP packet is RECOMMENDED but NOT REQUIRED, based on the thoughts of [RFC7202]

Like HEVC [RFC7798], [VVC] includes a user data Supplemental Enhancement Information (SEI) message. This SEI message allows inclusion of an arbitrary bitstring into the video bitstream. Such a bitstring could include JavaScript, machine code, and other active content. [VVC] leaves the handling of this SEI message to the receiving system. In order to avoid harmful side effects of the user data SEI message, decoder implementations cannot naively trust its content. For example, it would be a bad and insecure implementation practice to forward any JavaScript a decoder implementation detects to a web browser. The safest way to deal with user data SEI messages is to simply discard them, but that can have negative side effects on the quality of experience by the user.

End-to-end security with authentication, integrity, or confidentiality protection will prevent a MANE from performing media-aware operations other than discarding complete packets. In the case of confidentiality protection, it will even be prevented from discarding packets in a media-aware way. To be allowed to perform such operations, a MANE is required to be a trusted entity that is included in the security context establishment. This on-path inclusion of the MANE forgoes end-to-end security guarantees for the end points.

## 10. Congestion Control

Congestion control for RTP SHALL be used in accordance with RTP [RFC3550] and with any applicable RTP profile, e.g., AVP [RFC3551] or AVPF [RFC4585]. If best-effort service is being used, an additional requirement is that users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within an acceptable range. Packet loss is considered acceptable if a TCP flow across the same network path, and experiencing the same network conditions, would achieve an average throughput, measured on a reasonable timescale, that is not less than all RTP streams combined are achieved. This condition can be satisfied by implementing congestion-control mechanisms to adapt the transmission rate, the number of layers subscribed for a layered multicast session, or by arranging for a receiver to leave the session if the loss rate is unacceptably high.

The bitrate adaptation necessary for obeying the congestion control principle is easily achievable when real-time encoding is used, for example, by adequately tuning the quantization parameter. However, when pre-encoded content is being transmitted, bandwidth adaptation requires the pre-coded bitstream to be tailored for such adaptivity. The key mechanisms available in [VVC] are temporal scalability, and spatial/SNR scalability. A media sender can remove NAL units belonging to higher temporal sublayers (i.e., those NAL units with a high value of TID) or higher spatio-SNR layers until the sending bitrate drops to an acceptable range.

The mechanisms mentioned above generally work within a defined profile and level and, therefore, no renegotiation of the channel is required. Only when non-downgradable parameters (such as profile) are required to be changed does it become necessary to terminate and restart the RTP stream(s). This may be accomplished by using different RTP payload types.

MANEs MAY remove certain unusable packets from the RTP stream when that RTP stream was damaged due to previous packet losses. This can help reduce the network load in certain special cases. For example, MANEs can remove those FUs where the leading FUs belonging to the same NAL unit have been lost or those dependent slice segments when the leading slice segments belonging to the same slice have been lost, because the trailing FUs or dependent slice segments are meaningless to most decoders. MANE can also remove higher temporal scalable layers if the outbound transmission (from the MANE's viewpoint) experiences congestion.

## 11. IANA Considerations

A new media type, as specified in Section 7.1 of this memo, has been registered with IANA.

## 12. Acknowledgements

Dr. Byeongdo Choi is thanked for the video codec related technical discussion and other aspects in this memo. Xin Zhao and Dr. Xiang Li are thanked for their contributions on [VVC] specification descriptive content. Spencer Dawkins is thanked for his valuable review comments that led to great improvements of this memo. Some parts of this specification share text with the RTP payload format for HEVC [RFC7798]. We thank the authors of that specification for their excellent work.

## 13. References

### 13.1. Normative References

- [ISO23090-3] ISO/IEC 23090-3, "Information technology - Coded representation of immersive media Part 3 Versatile Video Coding", 2021, <<https://www.iso.org/standard/73022.html>>.
- [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>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.
- [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>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC4556] Zhu, L. and B. Tung, "Public Key Cryptography for Initial Authentication in Kerberos (PKINIT)", RFC 4556, DOI 10.17487/RFC4556, June 2006, <<https://www.rfc-editor.org/info/rfc4556>>.
- [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>>.
- [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <<https://www.rfc-editor.org/info/rfc4648>>.

- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.
- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/info/rfc5124>>.
- [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>>.
- [RFC8082] Wenger, S., Lennox, J., Burman, B., and M. Westerlund, "Using Codec Control Messages in the RTP Audio-Visual Profile with Feedback with Layered Codecs", RFC 8082, DOI 10.17487/RFC8082, March 2017, <<https://www.rfc-editor.org/info/rfc8082>>.
- [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>>.
- [RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <<https://www.rfc-editor.org/info/rfc8866>>.
- [VSEI] "Versatile supplemental enhancement information messages for coded video bitstreams", 2020, <<https://www.itu.int/rec/T-REC-H.274>>.
- [VVC] "Versatile Video Coding, ITU-T Recommendation H.266", 2020, <<http://www.itu.int/rec/T-REC-H.266>>.

### 13.2. Informative References

- [CABAC] and et al, "Transform coefficient coding in HEVC, IEEE Transactions on Circuits and Systems for Video Technology", DOI 10.1109/TCSVT.2012.2223055, December 2012, <<https://doi.org/10.1109/TCSVT.2012.2223055>>.
- [HEVC] "High efficiency video coding, ITU-T Recommendation H.265", 2019, <<https://www.itu.int/rec/T-REC-H.265>>.

- [MPEG2S] ISO/IEC, "Information technology - Generic coding of moving pictures and associated audio information - Part 1: Systems, ISO International Standard 13818-1", 2013.
- [RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, DOI 10.17487/RFC2974, October 2000, <<https://www.rfc-editor.org/info/rfc2974>>.
- [RFC6184] Wang, Y.-K., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <<https://www.rfc-editor.org/info/rfc6184>>.
- [RFC6190] Wenger, S., Wang, Y.-K., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <<https://www.rfc-editor.org/info/rfc6190>>.
- [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>>.
- [RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <<https://www.rfc-editor.org/info/rfc7667>>.
- [RFC7798] Wang, Y.-K., Sanchez, Y., Schierl, T., Wenger, S., and M. M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <<https://www.rfc-editor.org/info/rfc7798>>.
- [RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <<https://www.rfc-editor.org/info/rfc7826>>.

Appendix A. Change History

To RFC Editor: PLEASE REMOVE THIS SECTION BEFORE PUBLICATION

- draft-zhao-payload-rtp-vvc-00 ..... initial version
- draft-zhao-payload-rtp-vvc-01 ..... editorial clarifications and corrections
- draft-ietf-payload-rtp-vvc-00 ..... initial WG draft
- draft-ietf-payload-rtp-vvc-01 ..... VVC specification update
- draft-ietf-payload-rtp-vvc-02 ..... VVC specification update
- draft-ietf-payload-rtp-vvc-03 ..... VVC coding tool introduction update
- draft-ietf-payload-rtp-vvc-04 ..... VVC coding tool introduction update
- draft-ietf-payload-rtp-vvc-05 ..... reference update and adding placement for open issues
- draft-ietf-payload-rtp-vvc-06 ..... address editor's note
- draft-ietf-payload-rtp-vvc-07 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-08 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-09 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-10 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-11 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-12 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-13 ..... address editor's notes
- draft-ietf-payload-rtp-vvc-14 ..... address 2nd WGLC comments

Authors' Addresses

Shuai Zhao  
Intel  
2200 Mission College Blvd  
Santa Clara, 95054  
United States of America  
Email: shuai.zhao@ieee.org

Stephan Wenger  
Tencent  
2747 Park Blvd  
Palo Alto, 94588  
United States of America  
Email: stewe@stewe.org

Yago Sanchez  
Fraunhofer HHI  
Einsteinufer 37  
10587 Berlin  
Germany  
Email: yago.sanchez@hhi.fraunhofer.de

Ye-Kui Wang  
Bytedance Inc.  
8910 University Center Lane  
San Diego, 92122  
United States of America  
Email: yekui.wang@bytedance.com

Miska M. Hannuksela  
Nokia Technologies  
Hatanpään valtatie 30  
FI-33100 Tampere  
Finland  
Email: miska.hannuksela@nokia.com

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: May 24, 2020

M. Zanaty  
E. Berger  
S. Nandakumar  
Cisco Systems  
November 21, 2019

Frame Marking RTP Header Extension  
draft-ietf-avtext-framemarking-10

Abstract

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

Status of This Memo

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 <https://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 24, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Key Words for Normative Requirements . . . . .	4
3. Frame Marking RTP Header Extension . . . . .	4
3.1. Short Extension for Non-Scalable Streams . . . . .	4
3.2. Long Extension for Scalable Streams . . . . .	5
3.2.1. Layer ID Mappings for Scalable Streams . . . . .	7
3.2.1.1. H265 LID Mapping . . . . .	7
3.2.1.2. H264-SVC LID Mapping . . . . .	8
3.2.1.3. H264 (AVC) LID Mapping . . . . .	8
3.2.1.4. VP8 LID Mapping . . . . .	8
3.2.1.5. Future Codec LID Mapping . . . . .	8
3.3. Signaling Information . . . . .	9
3.4. Usage Considerations . . . . .	9
3.4.1. Relation to Layer Refresh Request (LRR) . . . . .	9
3.4.2. Scalability Structures . . . . .	10
4. Security Considerations . . . . .	10
5. Acknowledgements . . . . .	10
6. IANA Considerations . . . . .	10
7. References . . . . .	10
7.1. Normative References . . . . .	10
7.2. Informative References . . . . .	11
Authors' Addresses . . . . .	12

## 1. Introduction

Many widely deployed RTP [RFC3550] topologies [RFC7667] used in modern voice and video conferencing systems include a centralized component that acts as an RTP switch. It receives voice and video streams from each participant, which may be encrypted using SRTP [RFC3711], or extensions that provide participants with private media [I-D.ietf-perc-private-media-framework] via end-to-end encryption where the switch has no access to media decryption keys. The goal is to provide a set of streams back to the participants which enable them to render the right media content. In a simple video configuration, for example, the goal will be that each participant sees and hears just the active speaker. In that case, the goal of the switch is to receive the voice and video streams from each participant, determine the active speaker based on energy in the voice packets, possibly using the client-to-mixer audio level RTP header extension [RFC6464], and select the corresponding video stream for transmission to participants; see Figure 1.

In this document, an "RTP switch" is used as a common short term for the terms "switching RTP mixer", "source projecting middlebox", "source forwarding unit/middlebox" and "video switching MCU" as discussed in [RFC7667].

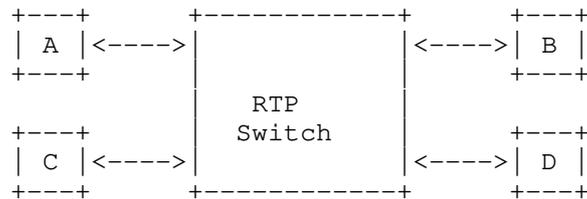


Figure 1: RTP switch

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

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

By providing meta-information about the RTP streams outside the encrypted media payload, an RTP switch can do codec-agnostic selective forwarding without decrypting the payload. This document specifies the necessary meta-information in an RTP header extension.

## 2. Key Words for Normative Requirements

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. Frame Marking RTP Header Extension

This specification uses RTP header extensions as defined in [RFC8285]. A subset of meta-information from the video stream is provided as an RTP header extension to allow an RTP switch to do generic selective forwarding of video streams encoded with potentially different video codecs.

The Frame Marking RTP header extension is encoded using the one-byte header or two-byte header as described in [RFC8285]. The one-byte header format is used for examples in this memo. The two-byte header format is used when other two-byte header extensions are present in the same RTP packet, since mixing one-byte and two-byte extensions is not possible in the same RTP packet.

This extension is only specified for Source (not Redundancy) RTP Streams [RFC7656] that carry video payloads. It is not specified for audio payloads, nor is it specified for Redundancy RTP Streams. The (separate) specifications for Redundancy RTP Streams often include provisions for recovering any header extensions that were part of the original source packet. Such provisions SHALL be followed to recover the Frame Marking RTP header extension of the original source packet. Source packet frame markings may be useful when generating Redundancy RTP Streams; for example, the I and D bits can be used to generate extra or no redundancy, respectively, and redundancy schemes with source blocks can align source block boundaries with Independent frame boundaries as marked by the I bit.

A frame, in the context of this specification, is the set of RTP packets with the same RTP timestamp from a specific RTP synchronization source (SSRC).

### 3.1. Short Extension for Non-Scalable Streams

The following RTP header extension is RECOMMENDED for non-scalable streams. It MAY also be used for scalable streams if the sender has limited or no information about stream scalability. The ID is assigned per [RFC8285], and the length is encoded as L=0 which indicates 1 octet of data.

```

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

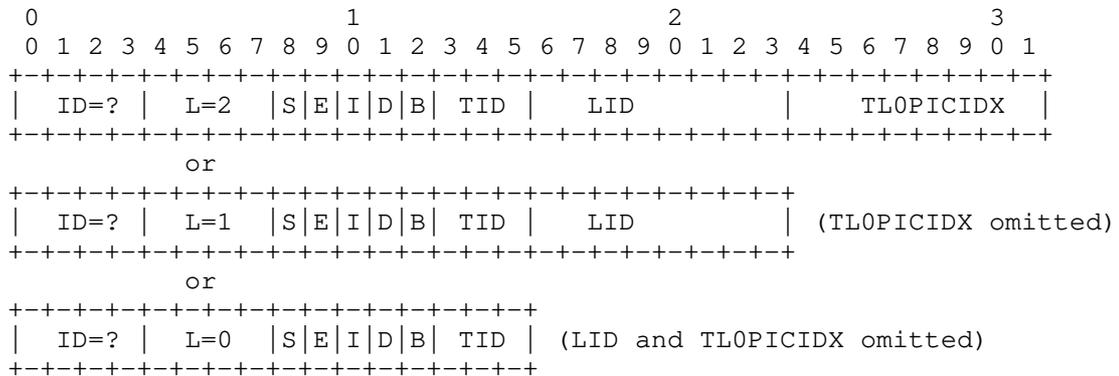
```

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

- o S: Start of Frame (1 bit) - MUST be 1 in the first packet in a frame; otherwise MUST be 0.
- o E: End of Frame (1 bit) - MUST be 1 in the last packet in a frame; otherwise MUST be 0. SHOULD match the RTP header marker bit in payload formats with such semantics for marking end of frame.
- o I: Independent Frame (1 bit) - MUST be 1 for frames that can be decoded independent of temporally prior frames, e.g. intra-frame, VPX keyframe, H.264 IDR [RFC6184], H.265 IDR/CRA/BLA/RAP [RFC7798]; otherwise MUST be 0.
- o D: Discardable Frame (1 bit) - MUST be 1 for frames the sender knows can be discarded, and still provide a decodable media stream; otherwise MUST be 0.
- o The remaining (4 bits) - are reserved/fixed values and not used for non-scalable streams; they MUST be set to 0 upon transmission and ignored upon reception.

### 3.2. Long Extension for Scalable Streams

The following RTP header extension is RECOMMENDED for scalable streams. It MAY also be used for non-scalable streams, in which case TID, LID and TL0PICIDX MUST be 0 or omitted. The ID is assigned per [RFC8285], and the length is encoded as L=2 which indicates 3 octets of data when nothing is omitted, or L=1 for 2 octets when TL0PICIDX is omitted, or L=0 for 1 octet when both LID and TL0PICIDX are omitted.



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

- o S: Start of Frame (1 bit) - MUST be 1 in the first packet in a frame within a layer; otherwise MUST be 0.
- o E: End of Frame (1 bit) - MUST be 1 in the last packet in a frame within a layer; otherwise MUST be 0. Note that the RTP header marker bit MAY be used to infer the last packet of the highest enhancement layer, in payload formats with such semantics.
- o I: Independent Frame (1 bit) - MUST be 1 for frames that can be decoded independent of temporally prior frames, e.g. intra-frame, VPX keyframe, H.264 IDR [RFC6184], H.265 IDR/CRA/BLA/RAP [RFC7798]; otherwise MUST be 0. Note that this bit only signals temporal independence, so it can be 1 in spatial or quality enhancement layers that depend on temporally co-located layers but not temporally prior frames.
- o D: Discardable Frame (1 bit) - MUST be 1 for frames the sender knows can be discarded, and still provide a decodable media stream; otherwise MUST be 0.
- o B: Base Layer Sync (1 bit) - When TID is not 0, this MUST be 1 if the sender knows this frame only depends on the base temporal layer; otherwise MUST be 0. When TID is 0 or if no scalability is used, this MUST be 0.
- o TID: Temporal ID (3 bits) - The base temporal layer starts with 0, and increases with 1 for each higher temporal layer/sub-layer. If no scalability is used, this MUST be 0. It is implicitly 0 in the short extension format.
- o LID: Layer ID (8 bits) - Identifies the spatial and quality layer encoded, starting with 0 and increasing with higher fidelity. If no scalability is used, this MUST be 0 or omitted to reduce length. When omitted, TL0PICIDX MUST also be omitted. It is implicitly 0 in the short extension format or when omitted in the long extension format.

- o TLOPICIDX: Temporal Layer 0 Picture Index (8 bits) - When TID is 0 and LID is 0, this is a cyclic counter labeling base layer frames. When TID is not 0 or LID is not 0, this indicates a dependency on the given index, such that this frame in this layer depends on the frame with this label in the layer with TID 0 and LID 0. If no scalability is used, or the cyclic counter is unknown, this MUST be omitted to reduce length. Note that 0 is a valid index value for TLOPICIDX.

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

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

### 3.2.1. Layer ID Mappings for Scalable Streams

This section maps the specific Layer ID information contained in specific scalable codecs to the generic LID and TID fields.

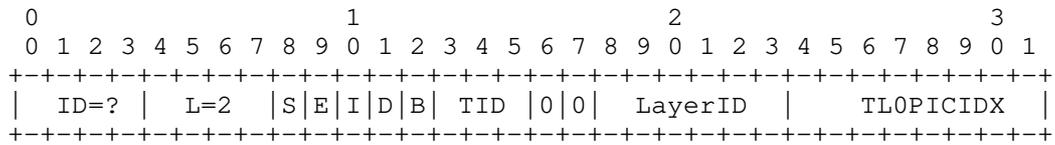
Note that non-scalable streams have no Layer ID information and thus no mappings.

#### 3.2.1.1. H265 LID Mapping

The following shows the H265 [RFC7798] LayerID (6 bits) and TID (3 bits) from the NAL unit header mapped to the generic LID and TID fields.

The I bit MUST be 1 when the NAL unit type is 16-23 (inclusive), otherwise it MUST be 0.

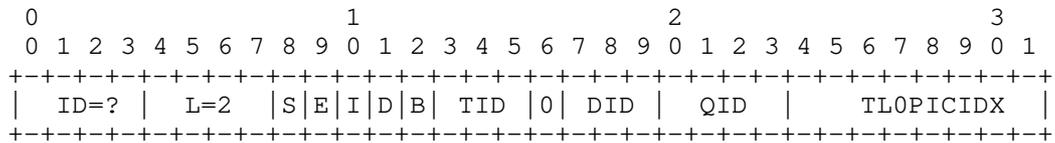
The S and E bits MUST match the corresponding bits in PACI:PHES:TSCI payload structures.



3.2.1.2. H264-SVC LID Mapping

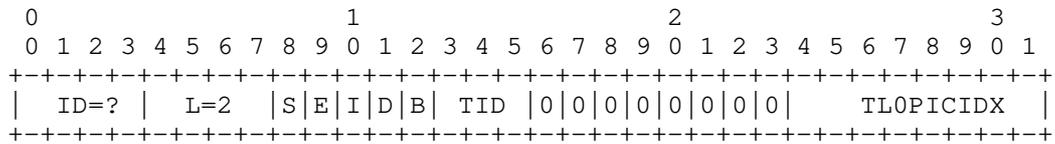
The following shows H264-SVC [RFC6190] Layer encoding information (3 bits for spatial/dependency layer, 4 bits for quality layer and 3 bits for temporal layer) mapped to the generic LID and TID fields.

The S, E, I and D bits MUST match the corresponding bits in PACSI payload structures.



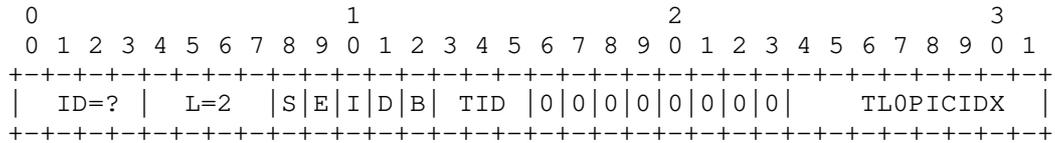
3.2.1.3. H264 (AVC) LID Mapping

The following shows the header extension for H264 (AVC) [RFC6184] that contains only temporal layer information.



3.2.1.4. VP8 LID Mapping

The following shows the header extension for VP8 [RFC7741] that contains only temporal layer information.



3.2.1.5. Future Codec LID Mapping

The RTP payload format specification for future video codecs SHOULD include a section describing the LID mapping and TID mapping for the

codec. For example, the LID/TID mapping for the VP9 codec is described in the VP9 RTP Payload Format [I-D.ietf-payload-vp9].

### 3.3. Signaling Information

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

An example attribute line in SDP:

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

### 3.4. Usage Considerations

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

When an RTP switch needs to discard a received video frame due to congestion control considerations, it is RECOMMENDED that it preferably drop frames marked with the D (Discardable) bit set, or the highest values of TID and LID, which indicate the highest temporal and spatial/quality enhancement layers, since those typically have fewer dependencies on them than lower layers.

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

#### 3.4.1. Relation to Layer Refresh Request (LRR)

Receivers can use the Layer Refresh Request (LRR) [I-D.ietf-avtext-lrr] RTCP feedback message to upgrade to a higher layer in scalable encodings. The TID/LID values and formats used in LRR messages MUST correspond to the same values and formats specified in Section 3.2.

Because frame marking can only be used with temporally-nested streams, temporal-layer LRR refreshes are unnecessary for frame-marked streams. Other refreshes can be detected based on the I bit being set for the specific spatial layers.

### 3.4.2. Scalability Structures

The LID and TID information is most useful for fixed scalability structures, such as nested hierarchical temporal layering structures, where each temporal layer only references lower temporal layers or the base temporal layer. The LID and TID information is less useful, or even not useful at all, for complex, irregular scalability structures that do not conform to common, fixed patterns of inter-layer dependencies and referencing structures. Therefore it is RECOMMENDED to use LID and TID information for RTP switch forwarding decisions only in the case of temporally nested scalability structures, and it is NOT RECOMMENDED for other (more complex or irregular) scalability structures.

## 4. Security Considerations

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

## 5. Acknowledgements

Many thanks to Bernard Aboba, Jonathan Lennox, Stephan Wenger, and Dale Worley for their inputs.

## 6. IANA Considerations

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

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

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

## 7. References

### 7.1. 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>>.
- [RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <<https://www.rfc-editor.org/info/rfc6184>>.
- [RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <<https://www.rfc-editor.org/info/rfc6190>>.
- [RFC7741] Westin, P., Lundin, H., Glover, M., Uberti, J., and F. Galligan, "RTP Payload Format for VP8 Video", RFC 7741, DOI 10.17487/RFC7741, March 2016, <<https://www.rfc-editor.org/info/rfc7741>>.
- [RFC7798] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S., and M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <<https://www.rfc-editor.org/info/rfc7798>>.
- [RFC8285] Singer, D., Desineni, H., and R. Even, Ed., "A General Mechanism for RTP Header Extensions", RFC 8285, DOI 10.17487/RFC8285, October 2017, <<https://www.rfc-editor.org/info/rfc8285>>.

## 7.2. Informative References

- [I-D.ietf-avtext-lrr]  
Lennox, J., Hong, D., Uberti, J., Holmer, S., and M. Flodman, "The Layer Refresh Request (LRR) RTCP Feedback Message", draft-ietf-avtext-lrr-07 (work in progress), July 2017.
- [I-D.ietf-payload-vp9]  
Uberti, J., Holmer, S., Flodman, M., Lennox, J., and D. Hong, "RTP Payload Format for VP9 Video", draft-ietf-payload-vp9-07 (work in progress), July 2019.
- [I-D.ietf-perc-private-media-framework]  
Jones, P., Benham, D., and C. Groves, "A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing (PERC)", draft-ietf-perc-private-media-framework-12 (work in progress), June 2019.

- [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>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.
- [RFC6464] Lennox, J., Ed., Ivov, E., and E. Marocco, "A Real-time Transport Protocol (RTP) Header Extension for Client-to-Mixer Audio Level Indication", RFC 6464, DOI 10.17487/RFC6464, December 2011, <<https://www.rfc-editor.org/info/rfc6464>>.
- [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>>.
- [RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <<https://www.rfc-editor.org/info/rfc7667>>.

## Authors' Addresses

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

Email: [mzanaty@cisco.com](mailto:mzanaty@cisco.com)

Espen Berger  
Cisco Systems

Phone: +47 98228179  
Email: [espeberg@cisco.com](mailto:espeberg@cisco.com)

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

Email: [snandaku@cisco.com](mailto:snandaku@cisco.com)

Payload Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: December 31, 2017

J. Lennox  
D. Hong  
Vidyo  
J. Uberti  
S. Holmer  
M. Flodman  
Google  
June 29, 2017

The Layer Refresh Request (LRR) RTCP Feedback Message  
draft-ietf-avtext-lrr-07

Abstract

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

Status of This Memo

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 December 31, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

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

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Conventions, Definitions and Acronyms . . . . .	2
2.1. Terminology . . . . .	3
3. Layer Refresh Request . . . . .	5
3.1. Message Format . . . . .	6
3.2. Semantics . . . . .	7
4. Usage with specific codecs . . . . .	8
4.1. H264 SVC . . . . .	8
4.2. VP8 . . . . .	9
4.3. H265 . . . . .	10
5. Usage with different scalability transmission mechanisms . . . . .	11
6. SDP Definitions . . . . .	11
7. Security Considerations . . . . .	12
8. IANA Considerations . . . . .	12
9. References . . . . .	12
9.1. Normative References . . . . .	12
9.2. Informative References . . . . .	13
Authors' Addresses . . . . .	14

## 1. Introduction

This memo describes an RTCP [RFC3550] Payload-Specific Feedback Message [RFC4585] "Layer Refresh Request" (LRR). It is designed to allow a receiver of a layered media stream to request that one or more of its substreams be refreshed, such that it can then be decoded by an endpoint which previously was not receiving those layers, without requiring that the entire stream be refreshed (as it would be if the receiver sent a Full Intra Request (FIR); [RFC5104] see also [RFC8082]).

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

## 2. Conventions, Definitions and Acronyms

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

## 2.1. Terminology

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

For spatial (or quality) layers, in normal encoding, a subpicture can depend both on earlier pictures of that spatial layer and also on lower-layer pictures of the current picture. A layer refresh, however, typically requires that a spatial layer picture be encoded in a way that references only the lower-layer subpictures of the current picture, not any earlier pictures of that spatial layer. Additionally, the encoder must promise that no earlier pictures of that spatial layer will be used as reference in the future.

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

An illustration of spatial layer refresh of an enhancement layer is shown below. <-- indicates a coding dependency.

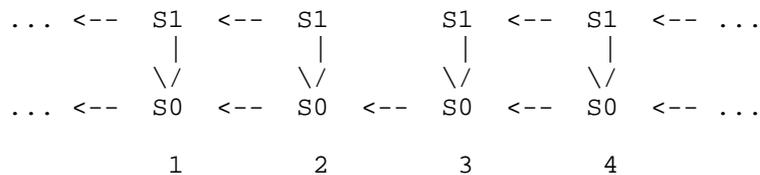


Figure 1

In Figure 1, frame 3 is a layer refresh point for spatial layer S1; a decoder which had previously only been decoding spatial layer S0 would be able to decode layer S1 starting at frame 3.

An illustration of spatial layer refresh of a base layer is shown below. <-- indicates a coding dependency.

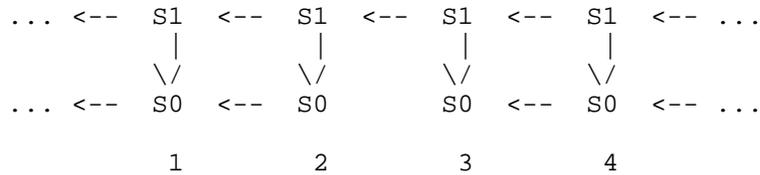


Figure 2

In Figure 2, frame 3 is a layer refresh point for spatial layer S0; a decoder which had previously not been decoding the stream at all could decode layer S0 starting at frame 3.

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

An illustration of temporal layer refresh is shown below. <-- indicates a coding dependency.

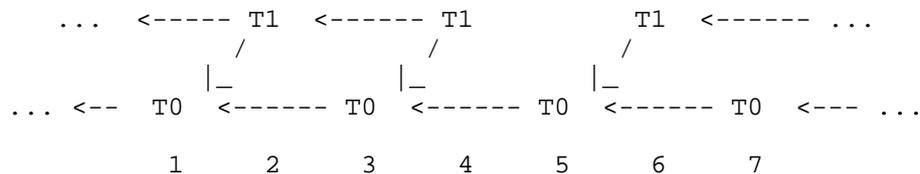


Figure 3

In Figure 3, frame 6 is a layer refresh point for temporal layer T1; a decoder which had previously only been decoding temporal layer T0 would be able to decode layer T1 starting at frame 6.

An illustration of an inherently temporally nested stream is shown below. <-- indicates a coding dependency.

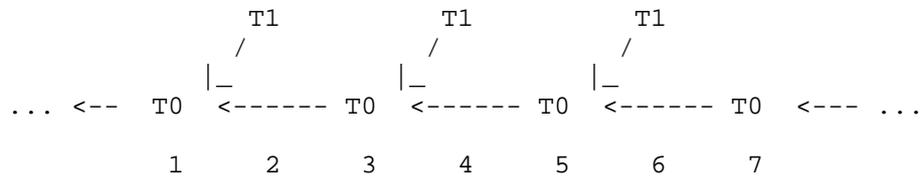


Figure 4

In Figure 4, the stream is temporally nested in its ordinary structure; a decoder receiving layer T0 can begin decoding layer T1 at any point.

A "Layer Index" is a numeric label for a specific spatial and temporal layer of a scalable stream. It consists of the pair of a "temporal ID" identifying the temporal layer, and a "layer ID" identifying the spatial or quality layer. The details of how layers of a scalable stream are labeled are codec-specific. Details for several codecs are defined in Section 4.

### 3. Layer Refresh Request

A layer refresh frame can be requested by sending a Layer Refresh Request (LRR), which is an RTP Control Protocol (RTCP) [RFC3550] payload-specific feedback message [RFC4585] asking the encoder to encode a frame which makes it possible to upgrade to a higher layer. The LRR contains one or two tuples, indicating the temporal and spatial layer the decoder wants to upgrade to, and (optionally) the currently highest temporal and spatial layer the decoder can decode.

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

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

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

[NOTE TO RFC Editor: Please replace "TBD" with the IANA-assigned payload-specific feedback number.]



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

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

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

When C is 1, TTID MUST NOT be less than CTID, and TLID MUST NOT be less than CLID; at least one of TTID or TLID MUST be greater than CTID or CLID respectively. That is to say, the target layer index <TTID, TLID> MUST be a layer upgrade from the current layer index <CTID, CLID>. A sender MAY request an upgrade in both temporal and spatial/quality layers simultaneously.

A receiver receiving an LRR feedback packet which does not satisfy the requirements of the previous paragraph, i.e. one where the C bit is present but TTID is less than CTID or TLID is less than CLID, MUST discard the request.

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

### 3.2. Semantics

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

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

The sender MUST respect bandwidth limits provided by the application of congestion control, as described in Section 5 of [RFC5104]. As layer refresh points will often be larger than non-refreshing frames,

this may restrict a sender's ability to send a layer refresh point quickly.

LRR MUST NOT be sent as a reaction to picture losses due to packet loss or corruption -- it is RECOMMENDED to use PLI [RFC4585] instead. LRR SHOULD be used only in situations where there is an explicit change in decoders' behavior, for example when a receiver will start decoding a layer which it previously had been discarding.

#### 4. Usage with specific codecs

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

##### 4.1. H264 SVC

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

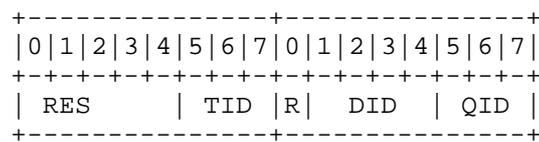


Figure 6

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

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

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

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

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

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

#### 4.2. VP8

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

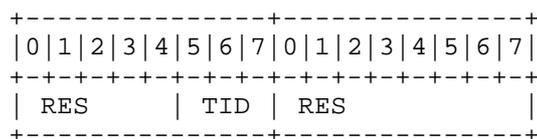


Figure 7

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

A VP8 layer refresh point can be identified by the presence of the "Y" bit in the VP8 payload header. When this bit is set, this and all subsequent frames depend only on the current base temporal layer.

On receipt of an LRR for a VP8 stream, A sender which supports LRR MUST encode the stream so it can set the Y bit in a packet whose temporal layer is at or below the target layer index.

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

#### 4.3. H265

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

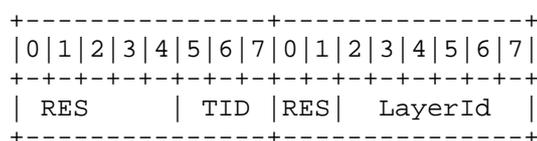


Figure 8

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

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

If a stream's `sps_temporal_id_nesting_flag` is not set, the NAL unit types 2 to 5 inclusively identify temporal layer switching points. A layer refresh to any higher target temporal layer is satisfied when a NAL unit type of 4 or 5 with TID equal to 1 more than current TID is seen. Alternatively, layer refresh to a target temporal layer can be incrementally satisfied with NAL unit type of 2 or 3. In this case, given current TID = T0 and target TID = TN, layer refresh to TN is satisfied when NAL unit type of 2 or 3 is seen for TID = T1, then TID = T2, all the way up to TID = TN. During this incremental process, layer refresh to TN can be completely satisfied as soon as a NAL unit type of 2 or 3 is seen.

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

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

#### 5. Usage with different scalability transmission mechanisms

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

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

#### 6. SDP Definitions

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

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

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

Figure 9: Syntax of the "lrr" ccm

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

## 7. Security Considerations

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

## 8. IANA Considerations

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

Value name: lrr

Long name: Layer Refresh Request Command

Usable with: ccm

Mux: IDENTICAL-PER-PT

Reference: RFC XXXX

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

Name: LRR

Long Name: Layer Refresh Request Command

Value: TBD

Reference: RFC XXXX

## 9. References

### 9.1. Normative References

[I-D.ietf-avtext-framemarking]

Berger, E., Nandakumar, S., and M. Zanaty, "Frame Marking RTP Header Extension", draft-ietf-avtext-framemarking-04 (work in progress), March 2017.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<http://www.rfc-editor.org/info/rfc3550>>.
- [RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <<http://www.rfc-editor.org/info/rfc4585>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<http://www.rfc-editor.org/info/rfc5104>>.
- [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <<http://www.rfc-editor.org/info/rfc5234>>.
- [RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <<http://www.rfc-editor.org/info/rfc6190>>.
- [RFC7741] Westin, P., Lundin, H., Glover, M., Uberti, J., and F. Galligan, "RTP Payload Format for VP8 Video", RFC 7741, DOI 10.17487/RFC7741, March 2016, <<http://www.rfc-editor.org/info/rfc7741>>.
- [RFC7798] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S., and M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <<http://www.rfc-editor.org/info/rfc7798>>.

## 9.2. Informative References

- [I-D.ietf-payload-vp9] Uberti, J., Holmer, S., Flodman, M., Lennox, J., and D. Hong, "RTP Payload Format for VP9 Video", draft-ietf-payload-vp9-03 (work in progress), March 2017.

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

[RFC8082] Wenger, S., Lennox, J., Burman, B., and M. Westerlund, "Using Codec Control Messages in the RTP Audio-Visual Profile with Feedback with Layered Codecs", RFC 8082, DOI 10.17487/RFC8082, March 2017, <<http://www.rfc-editor.org/info/rfc8082>>.

#### Authors' Addresses

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

Email: [jonathan@vidyo.com](mailto:jonathan@vidyo.com)

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

Email: [danny@vidyo.com](mailto:danny@vidyo.com)

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

Email: [justin@uberti.name](mailto:justin@uberti.name)

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

Email: holmer@google.com

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

Email: mflodman@google.com

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 9, 2017

A. Roach  
Mozilla  
S. Nandakumar  
Cisco Systems  
P. Thatcher  
Google  
October 06, 2016

RTP Stream Identifier Source Description (SDES)  
draft-ietf-avtext-rid-09

Abstract

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

Status of This Memo

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 9, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

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

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
3. Usage of RtpStreamId and RepairedRtpStreamId in RTP and RTCP	3
3.1. RTCP 'RtpStreamId' SDES Extension . . . . .	5
3.2. RTCP 'RepairedRtpStreamId' SDES Extension . . . . .	5
3.3. RTP 'RtpStreamId' and 'RepairedRtpStreamId' Header Extensions . . . . .	5
4. IANA Considerations . . . . .	6
4.1. New RtpStreamId SDES item . . . . .	6
4.2. New RepairRtpStreamId SDES item . . . . .	6
4.3. New RtpStreamId Header Extension URI . . . . .	7
4.4. New RepairRtpStreamId Header Extension URI . . . . .	7
5. Security Considerations . . . . .	7
6. Acknowledgements . . . . .	8
7. References . . . . .	8
7.1. Normative References . . . . .	8
7.2. Informative References . . . . .	9
Authors' Addresses . . . . .	9

## 1. Introduction

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

To address this situation, we define a new RTCP Stream Identifier Source Description (SDES) identifier, `RtpStreamId`, that uniquely identifies a single RTP stream. A key motivator for defining this identifier is the ability to differentiate among different encodings of a single Source Stream that are sent simultaneously (i.e., simulcast). This need for unique identification extends to dependent

streams (e.g., where layers used by a layered codec are transmitted on separate streams).

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

## 2. Terminology

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

The following acronyms are also used:

- o CNAME: Canonical End-Point Identifier, defined in [RFC3550]
- o MID: Media Identification, defined in [I-D.ietf-mmusic-sdp-bundle-negotiation]
- o MSID: Media Stream Identifier, defined in [I-D.ietf-mmusic-msid]
- o RTCP: Real-time Transport Control Protocol, defined in [RFC3550]
- o RTP: Real-time Transport Protocol, defined in [RFC3550]
- o SDES: Source Description, defined in [RFC3550]
- o SSRC: Synchronization Source, defined in [RFC3550]

## 3. Usage of `RtpStreamId` and `RepairedRtpStreamId` in RTP and RTCP

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

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

redundancy RTP streams to indicate the RTP stream repaired by a redundancy RTP stream.

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

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

RtpStreamId and RepairedRtpStreamId values are scoped by source identifier (e.g., CNAME) and by media session. When the media is multiplexed using the BUNDLE extension [I-D.ietf-mmusic-sdp-bundle-negotiation], these values are further scoped by their associated MID values. For example: an RtpStreamId of "1" may be present in the stream identified with a CNAME of "1234@example.com", and may also be present in a stream with a CNAME of "5678@example.org", and these would refer to different streams. Similarly, an RtpStreamId of "1" may be present with an MID of "A", and again with a MID of "B", and also refer to two different streams.

Note that the RepairedRtpStreamId mechanism is limited to indicating one repaired stream per redundancy stream. If systems require correlation for schemes in which a redundancy stream contains information used to repair more than one stream, they will have to use a more complex mechanism than the one defined in this specification.

As with all SDES items, RtpStreamId and RepairedRtpStreamId are limited to a total of 255 octets in length. RtpStreamId and RepairedStreamId are constrained to contain only alphanumeric characters. For avoidance of doubt, the only allowed byte values for these IDs are decimal 48 through 57, 65 through 90, and 97 through 122.

3.1. RTCP 'RtpStreamId' SDES Extension

```

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|RtpStreamId=TBD|      length      | RtpStreamId                      ...
+-----+-----+-----+-----+-----+-----+-----+-----+

```

The RtpStreamId payload is ASCII encoded and is not null-terminated.

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

3.2. RTCP 'RepairedRtpStreamId' SDES Extension

```

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|Repaired...=TBD|      length      | RepairRtpStreamId                      ...
+-----+-----+-----+-----+-----+-----+-----+-----+

```

The RepairedRtpStreamId payload is ASCII encoded and is not null-terminated.

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

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

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

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

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

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

#### 4. IANA Considerations

##### 4.1. New RtpStreamId SDES item

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

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

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

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

##### 4.2. New RepairRtpStreamId SDES item

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

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

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

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

#### 4.3. New RtpStreamId Header Extension URI

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

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

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

#### 4.4. New RepairRtpStreamId Header Extension URI

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

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

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

### 5. Security Considerations

Although the identifiers defined in this document are limited to be strictly alphanumeric, SDES items have the potential to carry any string. As a consequence, there exists a risk that it might carry privacy-sensitive information. Implementations need to take care when generating identifiers so that they do not contain information that can identify the user or allow for long term tracking of the device. Following the generation recommendations in Section 3.3 will result in non-instance-specific labels, with only minor fingerprinting possibilities in the total number of used RtpStreamIds and RepairedRtpStreamIds.

Even if the SDES items are generated to convey as little information as possible, implementors are strongly encouraged to encrypt SDES items - both in RTCP and RTP header extensions - so as to preserve privacy against third parties.

As the SDES items are used for identification of the RTP streams for different application purposes, it is important that the intended values are received. An attacker, either a third party or malicious RTP middlebox, that removes, or changes the values for these SDES

items, can severely impact the application. The impact can include failure to decode or display the media content of the RTP stream. It can also result in incorrectly attributing media content to identifiers of the media source, such as incorrectly identifying the speaker. To prevent this from occurring due to third party attacks, integrity and source authentication is needed.

Options for Securing RTP Sessions [RFC7201] discusses options for how encryption, integrity and source authentication can be accomplished.

## 6. Acknowledgements

Many thanks for review and input from Cullen Jennings, Magnus Westerlund, Colin Perkins, Jonathan Lennox, and Paul Kyzivat. Magnus Westerlund provided substantially all of the Security Considerations section.

## 7. References

### 7.1. Normative References

- [I-D.ietf-avtext-sdes-hdr-ext]  
Westerlund, M., Burman, B., Even, R., and M. Zanaty, "RTP Header Extension for RTCP Source Description Items", draft-ietf-avtext-sdes-hdr-ext-07 (work in progress), June 2016.
- [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-32 (work in progress), August 2016.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<http://www.rfc-editor.org/info/rfc3550>>.
- [RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <<http://www.rfc-editor.org/info/rfc5285>>.
- [RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <<http://www.rfc-editor.org/info/rfc7656>>.

## 7.2. Informative References

[I-D.ietf-mmusic-msid]

Alvestrand, H., "WebRTC MediaStream Identification in the Session Description Protocol", draft-ietf-mmusic-msid-15 (work in progress), July 2016.

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

## Authors' Addresses

Adam Roach  
Mozilla

Email: [adam@nostrum.com](mailto:adam@nostrum.com)

Suhas Nandakumar  
Cisco Systems

Email: [snandaku@cisco.com](mailto:snandaku@cisco.com)

Peter Thatcher  
Google

Email: [pthatcher@google.com](mailto:pthatcher@google.com)

avtcore  
Internet-Draft  
Intended status: Standards Track  
Expires: October 10, 2020

S. Lujan  
A. Descampe  
C. Damman  
intoPIX  
T. Richter  
IIS  
A. Willeme  
UCL/ICTEAM  
April 8, 2020

RTP Payload Format for ISO/IEC 21122 (JPEG XS)  
draft-ietf-payload-rtp-jpegxs-03

Abstract

This document specifies a Real-Time Transport Protocol (RTP) payload format to be used for transporting JPEG XS (ISO/IEC 21122) encoded video. JPEG XS is a low-latency, lightweight image coding system. Compared to an uncompressed video use case, it allows higher resolutions and frame rates, while offering visually lossless quality, reduced power consumption, and end-to-end latency confined to a fraction of a frame.

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 <https://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 October 10, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1. Introduction	2
2. Conventions, Definitions, and Abbreviations	3
3. Media Format Description	4
3.1. Image Data Structures	4
3.2. Codestream	5
3.3. Video support box and colour specification box	5
3.4. JPEG XS Frame	5
4. Payload Format	6
4.1. RTP packetization	6
4.2. Payload Header	8
4.3. Payload Data	11
4.4. Traffic Shaping and Delivery Timing	14
5. Congestion Control Considerations	14
6. Payload Format Parameters	14
6.1. Media Type Definition	14
6.2. Mapping to SDP	17
6.2.1. General	17
6.2.2. Media type and subtype	18
6.2.3. Traffic shaping	18
6.2.4. Offer/Answer Considerations	18
7. IANA Considerations	19
8. Security Considerations	19
9. RFC Editor Considerations	20
10. References	20
10.1. Normative References	20
10.2. Informative References	22
10.3. URIs	22
Authors' Addresses	22

## 1. Introduction

This document specifies a payload format for packetization of JPEG XS encoded video signals into the Real-time Transport Protocol (RTP) [RFC3550].

The JPEG XS coding system offers compression and recompression of image sequences with very moderate computational resources while remaining robust under multiple compression and decompression cycles

and mixing of content sources, e.g. embedding of subtitles, overlays or logos. Typical target compression ratios ensuring visually lossless quality are in the range of 2:1 to 10:1, depending on the nature of the source material. The end-to-end latency can be confined to a fraction of a frame, typically between a small number of lines down to below a single line.

## 2. Conventions, Definitions, and Abbreviations

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].

### Application Data Unit (ADU)

The unit of source data provided as payload to the transport layer, and corresponding, in this RTP payload definition, to a single JPEG XS frame.

### Colour specification box (CS box)

A ISO colour specification box defined in ISO/IEC 21122-3 [ISO21122-3] that includes colour-related metadata required to correctly display JPEG XS frames, such as colour primaries, transfer characteristics and matrix coefficients.

### EOC marker

A marker that consists of the two bytes 0xff11 indicating the end of a JPEG XS codestream.

### JPEG XS codestream

A sequence of bytes representing a compressed image formatted according to JPEG XS Part-1 [ISO21122-1].

### JPEG XS codestream header

A sequence of bytes, starting with a SOC marker, at the beginning of each JPEG XS codestream encoded in multiple markers and marker segments that does not carry entropy coded data, but metadata such as the frame dimension and component precision.

### JPEG XS frame

The concatenation of a video support box, as defined in ISO/IEC 21122-3 [ISO21122-3], a colour specification box, as defined in ISO/IEC 21122-3 [ISO21122-3] as well, and either one JPEG XS codestream in the case of a progressive frame or two JPEG XS codestreams in the case of an interlaced frame.

### JPEG XS header segment

The concatenation of a video support box, as defined in ISO/IEC 21122-3 [ISO21122-3], a colour specification box, as defined in

ISO/IEC 21122-3 as well [ISO21122-3] and a JPEG XS codestream header.

JPEG XS stream

A sequence of JPEG XS frames.

Marker

A two-byte functional sequence that is part of a JPEG XS codestream starting with a 0xff byte and a subsequent byte defining its function.

Marker segment

A marker along with a 16-bit marker size and payload data following the size.

Packetization unit

A portion of a Application Data Unit whose boundaries shall coincide with boundaries of RTP packet payloads, i.e. the first (resp. last) byte of a packetization unit shall be the first (resp. last) byte of a RTP packet payload.

Slice

The smallest independently decodable unit of a JPEG XS codestream, bearing in mind that it decodes to wavelet coefficients which still require inverse wavelet filtering to give an image.

SOC marker

A marker that consists of the two bytes 0xff10 indicating the start of a JPEG XS codestream.

Video support box (VS box)

A ISO video support box defined in ISO/IEC 21122-3 [ISO21122-3] that includes metadata required to play back a JPEG XS stream, such as its maximum bitrate, its subsampling structure, its buffer model and its frame rate.

### 3. Media Format Description

#### 3.1. Image Data Structures

JPEG XS is a low-latency lightweight image coding system for coding continuous-tone grayscale or continuous-tone colour digital images.

This coding system provides an efficient representation of image signals through the mathematical tool of wavelet analysis. The wavelet filter process separates each component into multiple bands, where each band consists of multiple coefficients describing the image signal of a given component within a frequency domain specific

to the wavelet filter type, i.e. the particular filter corresponding to the band.

Wavelet coefficients are grouped into precincts, where each precinct includes all coefficients over all bands that contribute to a spatial region of the image.

One or multiple precincts are furthermore combined into slices consisting of an integer number of precincts. Precincts do not cross slice boundaries, and wavelet coefficients in precincts that are part of different slices can be decoded independently from each other. Note, however, that the wavelet transformation runs across slice boundaries. A slice always extends over the full width of the image, but may only cover parts of its height.

### 3.2. Codestream

A JPEG XS codestream header, followed by several slices, and terminated by an EOC marker form a JPEG XS codestream.

The overall codestream format, including the definition of all markers, is further defined in ISO/IEC 21122-1 [ISO21122-1]. It represents sample values of a single image, bare any interpretation relative to a colour space.

### 3.3. Video support box and colour specification box

While the information defined in the codestream is sufficient to reconstruct the sample values of one image, the interpretation of the samples remains undefined by the codestream itself. This interpretation is given by the video support box and the colour specification box which contain significant information to correctly play the JPEG XS stream. The layout and syntax of these boxes, together with their content, are defined in ISO/IEC 21122-3 [ISO21122-3]. The video support box provides information on the maximum bitrate, the frame rate, the subsampling image format, the timecode of the current JPEG XS frame, the profile, level and sublevel used (as defined in ISO/IEC 21122-2 [ISO21122-2]), and optionally on the buffer model and the mastering display metadata. The colour specification box indicates the colour primaries, transfer characteristics, matrix coefficients and video full range flag needed to specify the colour space of the video stream.

### 3.4. JPEG XS Frame

The concatenation of a video support box, a colour specification box and one or two JPEG XS codestreams forms a JPEG XS frame. In the case of a video stream made of progressive frames, only one

codestream follows the boxes. In the case of a video stream made of interlaced frames, two codestreams follow the boxes, each corresponding to a field of the interlaced frame. The video information box included in the video support box contains a field indicating if the frame is progressive or interlaced (see ISO/IEC 21122-3 [ISO21122-3]). This information can also be found in each RTP packet header (see Section 4.2).

#### 4. Payload Format

This section specifies the payload format for JPEG XS streams over the Real-time Transport Protocol (RTP) [RFC3550].

In order to be transported over RTP, each JPEG XS stream is transported in a distinct RTP stream, identified by a distinct SSRC.

A JPEG XS stream is divided into Application Data Units (ADUs), each ADU corresponding to a single JPEG XS frame.

##### 4.1. RTP packetization

An ADU is made of several packetization units. If a packetization unit is bigger than the maximum size of a RTP packet payload, the unit is split into multiple RTP packet payloads, as illustrated in Figure 1. As seen there, each packet shall contain (part of) one and only one packetization unit. A packetization unit may extend over multiple packets. The payload of every packet shall have the same size (based e.g. on the Maximum Transfer Unit of the network), except (possibly) the last packet of a packetization unit. The boundaries of a packetization unit shall coincide with the boundaries of the payload of a packet, i.e. the first (resp. last) byte of the packetization unit shall be the first (resp. last) byte of the payload.

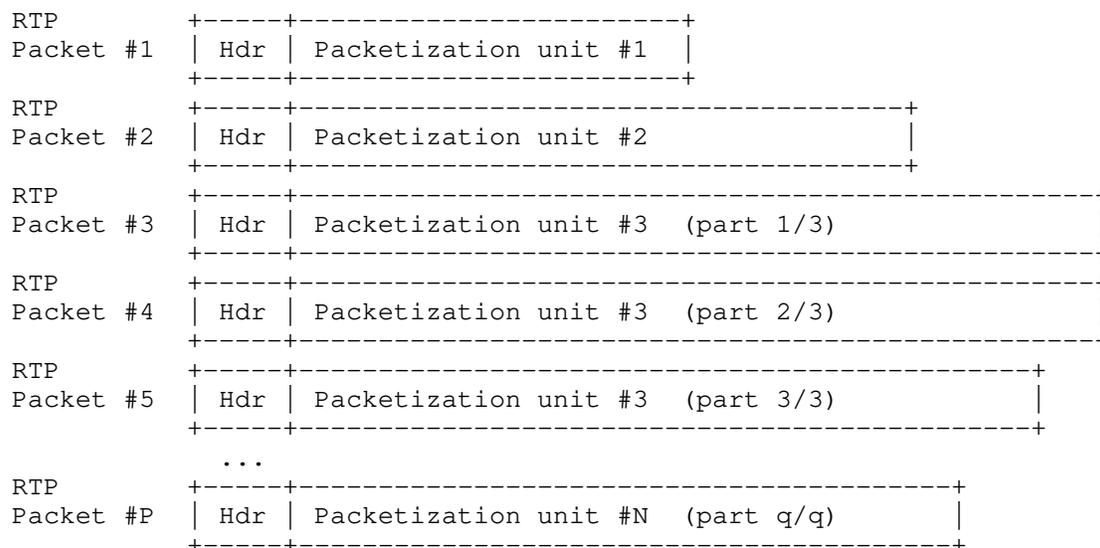


Figure 1: Example of ADU packetization

There are two different packetization modes defined for this RTP payload format.

1. Codestream packetization mode: in this mode, the packetization unit shall be the entire codestream, preceeded by boxes, if any. This means that a progressive frame will have a single packetization unit, while an interlaced frame will have two. The progressive case is illustrated in Figure 2.
2. Slice packetization mode: in this mode, the packetization unit shall be the slice, i.e. there shall be data from no more than one slice per RTP packet. The first packetization unit shall be made of the JPEG XS header segment (i.e. the concatenation of the VS box, the CS box and the JPEG XS codestream header). This first unit is then followed by successive units, each containing one and only one slice. The packetization unit containing the last slice of a JPEG XS codestream shall also contain the EOC marker immediately following this last slice. This is illustrated in Figure 3. In the case of interlaced frame, the JPEG XS codestream header of the second field shall be in its own packetization unit.

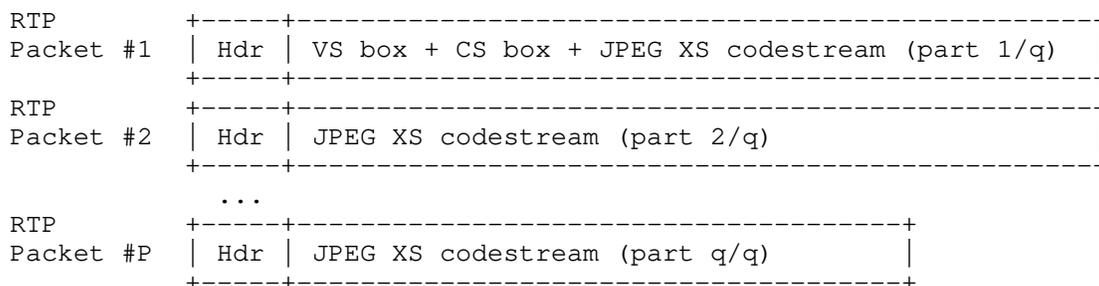


Figure 2: Example of codestream packetization mode

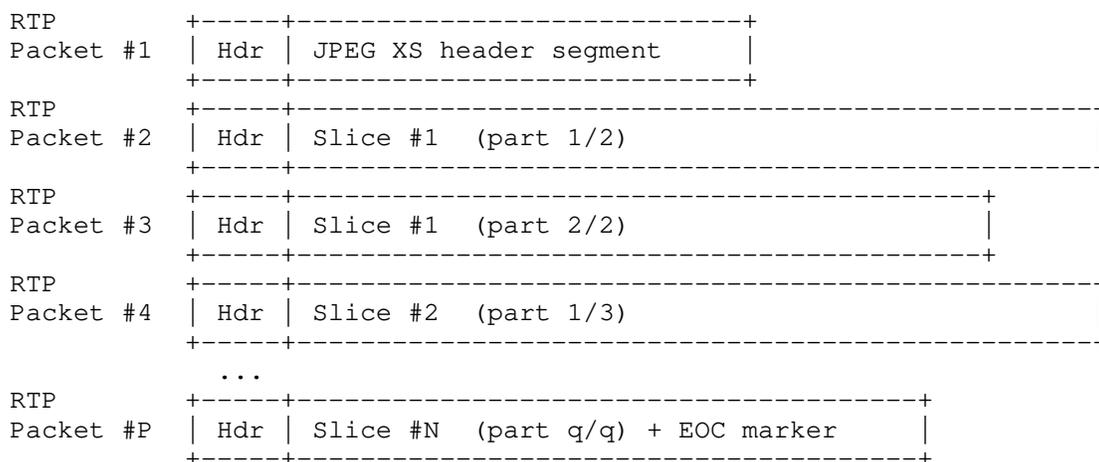


Figure 3: Example of slice packetization mode

Thanks to the constant bit-rate of JPEG XS, the codestream packetization mode guarantees that a JPEG XS RTP stream will produce a constant number of bytes per frame, and a constant number of RTP packets per frame. To reach the same guarantee with the slice packetization mode, an additional constraint needs to be set at the rate allocation stage in the JPEG XS encoder. For instance, one option would be to impose a constant bit-rate at the slice level.

#### 4.2. Payload Header

Figure 4 illustrates the RTP payload header used in order to transport a JPEG XS stream.

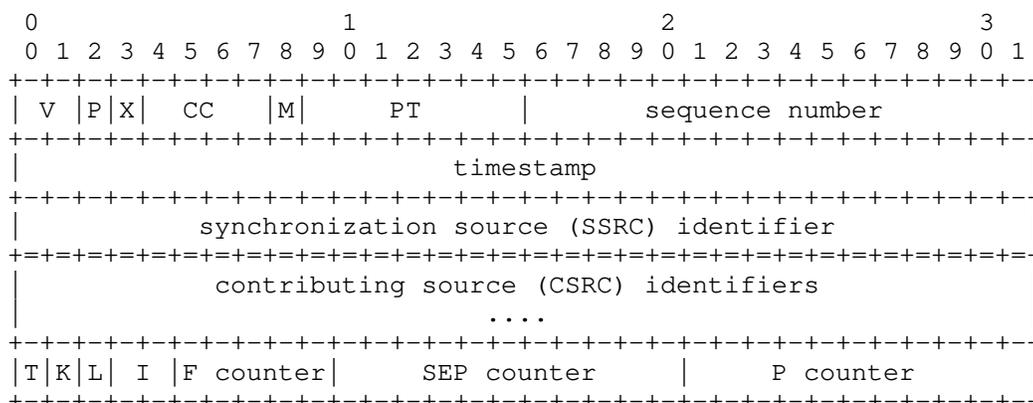


Figure 4: RTP and payload headers

The version (V), padding (P), extension (X), CSRC count (CC), sequence number, synchronization source (SSRC) and contributing source (CSRC) fields follow their respective definitions in RFC 3550 [RFC3550].

The timestamp SHOULD be based on a 90 kHz clock reference.

As per specified in RFC 3550 [RFC3550] and RFC 4175 [RFC4175], the RTP timestamp designates the sampling instant of the first octet of the frame to which the RTP packet belongs. Packets shall not include data from multiple frames, and all packets belonging to the same frame shall have the same timestamp. Several successive RTP packets will consequently have equal timestamps if they belong to the same frame (that is until the marker bit is set to 1, marking the last packet of the frame), and the timestamp is only increased when a new frame begins.

If the sampling instant does not correspond to an integer value of the clock, the value shall be truncated to the next lowest integer, with no ambiguity.

The remaining fields are defined as follows:

Marker (M) [1 bit]:

The M bit is used to indicate the last packet of a frame. This enables a decoder to finish decoding the frame.

Payload Type (PT) [7 bits]:

A dynamically allocated payload type field that designates the payload as JPEG XS video.

**Transmission mode (T) [1 bit]:**

The T bit is set to indicate that packets are sent sequentially by the transmitter. A receiver could use this information to dimension its input buffer(s) accordingly. If T=0, nothing can be assumed about the transmission order and packets may be sent out-of-order by the transmitter. If T=1, packets must be sent sequentially by the transmitter.

**pacKetization mode (K) [1 bit]:**

The K bit is set to indicate which packetization mode is used. K=0 indicates codestream packetization mode, while K=1 indicates slice packetization mode. If Transmission mode (T) is set to 0, slice packetization mode must be used and K must be set to 1.

**Last (L) [1 bit]:**

The L bit is set to indicate the last packet of a packetization unit. As the end of the frame also ends the packet containing the last unit of the frame, the L bit is set whenever the M bit is set. In the case of a progressive frame using the codestream packetization mode, the L bit and M bit are equivalent.

**Interlaced mode (I) [2 bit]:**

These 2 bits are used to indicate how the JPEG XS frame is scanned (progressive or interlaced).

00: The payload is progressively scanned.

01: Reserved for future use.

10: The payload is part of the first field of an interlaced video frame. The height specified in the JPEG XS picture header is half of the height of the entire displayed image.

11: The payload is part of the second field of an interlaced video frame. The height specified in the JPEG XS picture header is half of the height of the entire displayed image.

**F counter [5 bits]:**

The frame (F) counter identifies the frame number modulo 32 to which a packet belongs. Frame numbers are incremented by 1 for each frame transmitted. The frame number, in addition to the time stamp, may help the decoder manage its input buffer and bring packets back into their natural order.

**SEP counter [11 bits]:**

The Slice and Extended Packet (SEP) counter is used differently depending on the packetization mode.

- \* In the case of codestream packetization mode (K=0), this counter resets whenever the Packet counter resets (see hereunder), and increments by 1 whenever the Packet counter overruns.
- \* In the case of slice packetization mode (K=1), this counter identifies the slice modulo 2047 to which the packet contributes. If the data belongs to the JPEG XS header segment, this field shall have its maximal value, namely 2047 = 0x07ff. Otherwise, it is the slice index modulo 2047. Slice indices are counted from 0 (corresponding to the top of the frame).

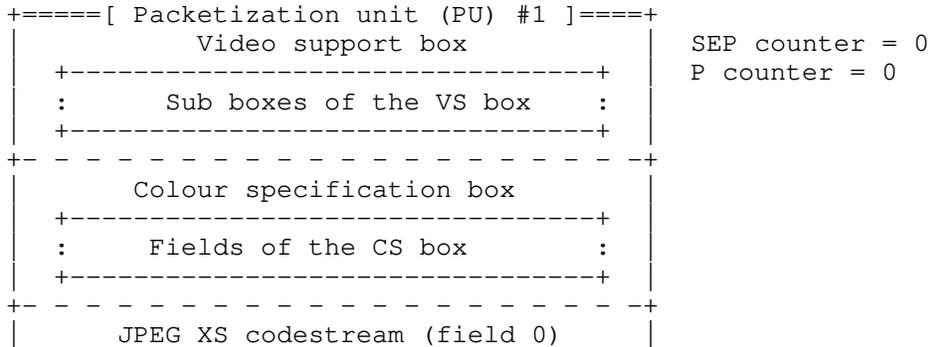
P counter [11 bits]:

The packet (P) counter identifies the packet number modulo 2048 within the current packetization unit. It is set to 0 at the start of the packetization unit and incremented by 1 for every subsequent packet (if any) belonging to the same unit. Practically, if codestream packetization mode is enabled, this field counts the packets within a codestream and is extended by the SEP counter when it overruns. If slice packetization mode is enabled, this field counts the packets within a slice or within the JPEG XS header segment.

4.3. Payload Data

The payload data of a JPEG XS RTP stream consists of a concatenation of multiple JPEG XS frames.

Each JPEG XS frame is the concatenation of one or more packetization unit(s), as explained in Section 4.1. Figure 5 depicts this layout for an interlaced frame in the codestream packetization mode and Figure 6 depicts this layout for a progressive frame in the slice packetization mode. The Frame counter value is not indicated because the value is constant for all packetization units of a given frame.



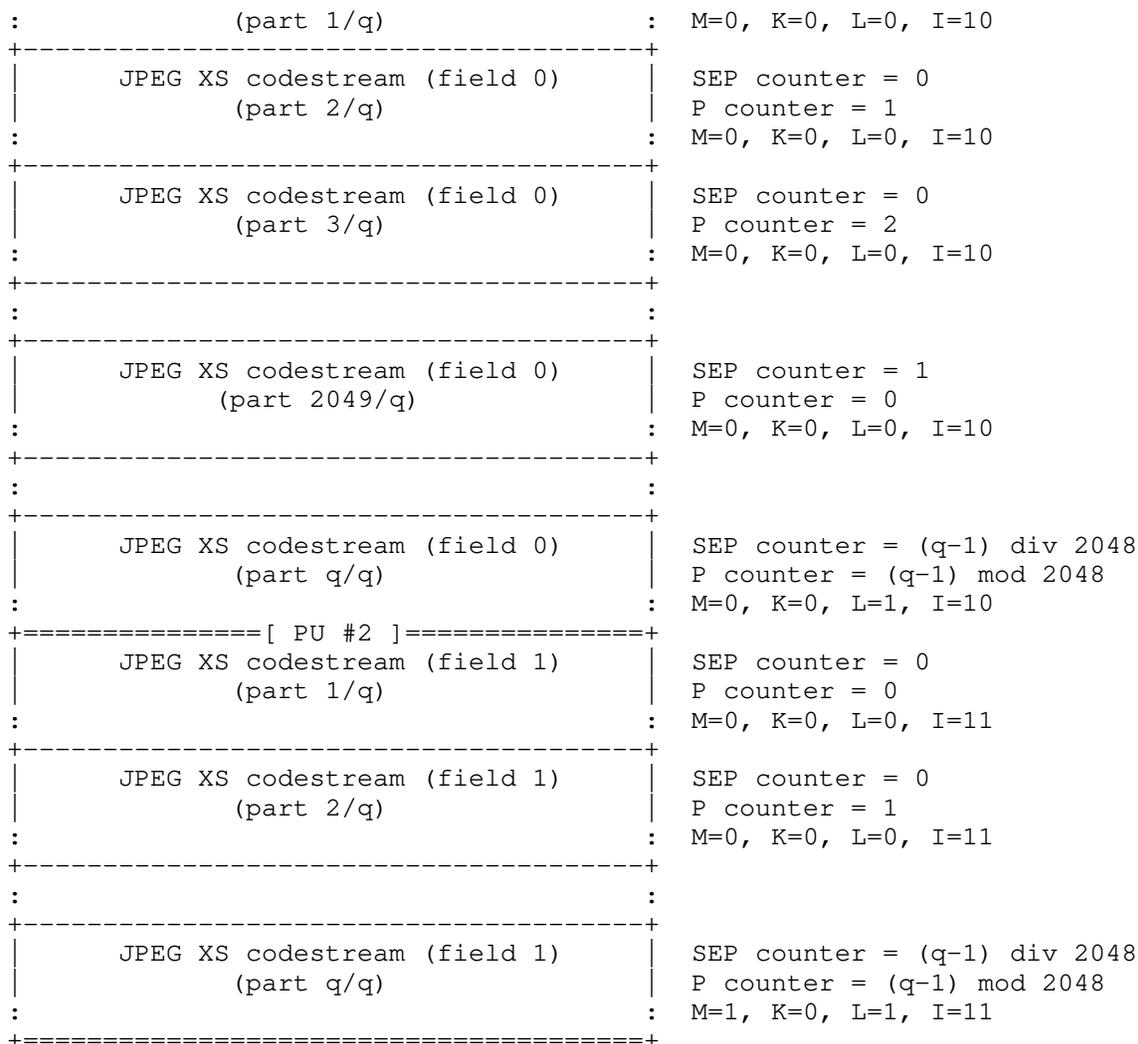
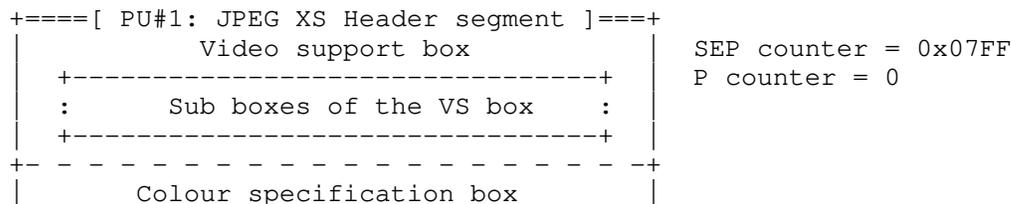


Figure 5: Example of JPEG XS Payload Data (codestream packetization mode, interlaced frame)



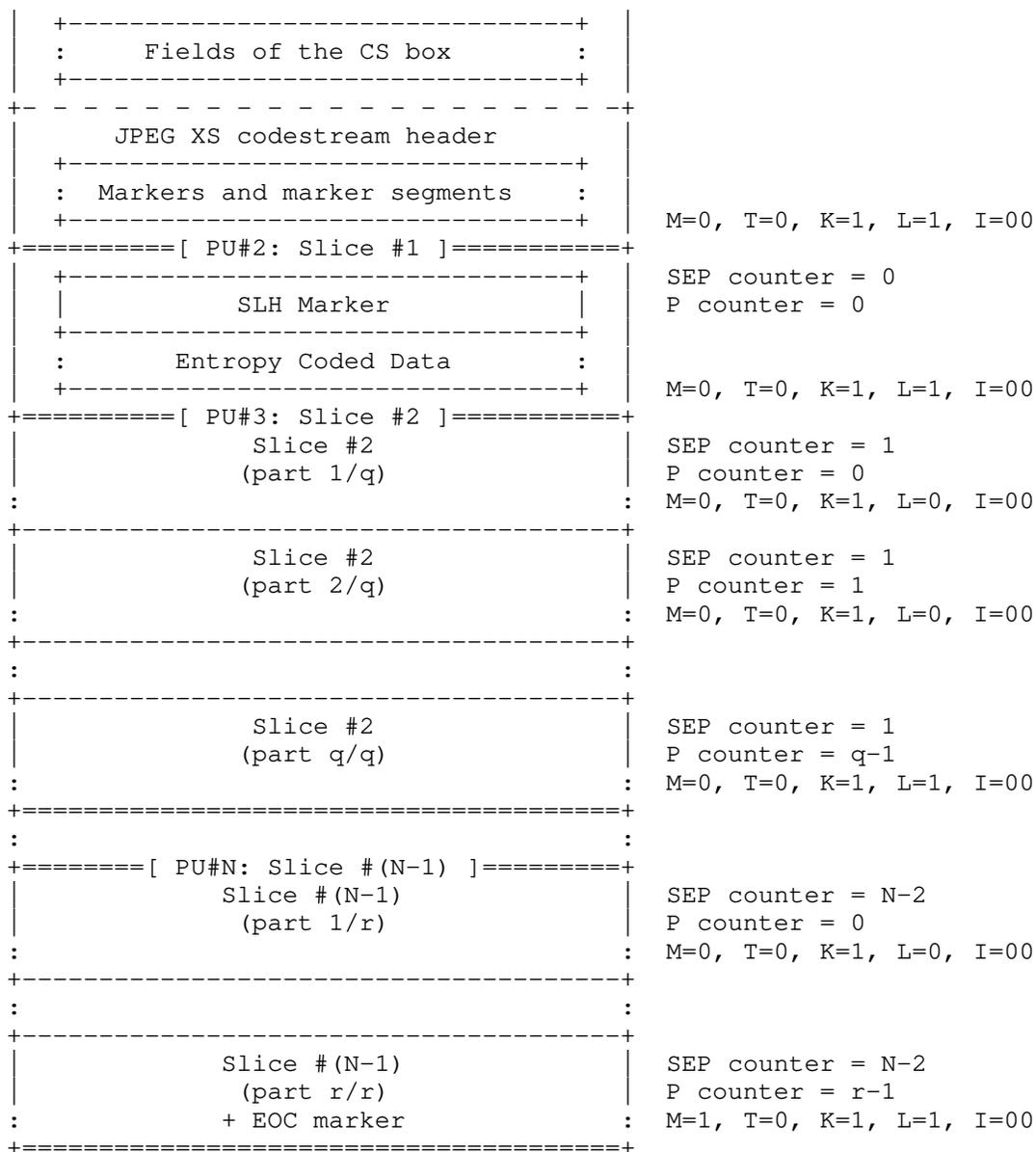


Figure 6: Example of JPEG XS Payload Data (slice packetization mode, progressive frame)

#### 4.4. Traffic Shaping and Delivery Timing

The traffic shaping and delivery timing shall be in accordance with the Network Compatibility Model compliance definitions specified in SMPTE ST 2110-21 [SMPTE-ST2110-21] for either Narrow Linear Senders (Type NL) or Wide Senders (Type W). The session description shall include a format-specific parameter of either TP=2110TPNL or TP=2110TPW to indicate compliance with Type NL or Type W respectively.

NOTE: The Virtual Receiver Buffer Model compliance definitions of ST 2110-21 do not apply.

### 5. Congestion Control Considerations

Congestion control for RTP SHALL be used in accordance with RFC 3550 [RFC3550], and with any applicable RTP profile: e.g., RFC 3551 [RFC3551]. An additional requirement if best-effort service is being used is users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within acceptable parameters. Circuit Breakers [RFC8083] is an update to RTP [RFC3550] that defines criteria for when one is required to stop sending RTP Packet Streams and applications implementing this standard MUST comply with it. RFC 8085 [RFC8085] provides additional information on the best practices for applying congestion control to UDP streams.

### 6. Payload Format Parameters

#### 6.1. Media Type Definition

Type name: video

Subtype name: jxsv

Required parameters:

rate: The RTP timestamp clock rate. Applications using this payload format SHOULD use a value of 90000.

transmission mode: Indicates if packets are sent sequentially by the transmitter. A receiver could use this information to dimension its input buffer(s) accordingly. If set to 0, nothing can be assumed about the transmission order and packets may be sent out-of-order. If value is 1, packets must be sent sequentially by the transmitter.

Optional parameters:

profile: The JPEG XS profile in use, as defined in ISO/IEC 21122-2 (JPEG XS Part 2) [ISO21122-2].

level: The JPEG XS level in use, as defined in ISO/IEC 21122-2 (JPEG XS Part 2) [ISO21122-2].

sublevel: The JPEG XS sublevel in use, as defined in ISO/IEC 21122-2 (JPEG XS Part 2) [ISO21122-2].

sampling: Signals the colour difference signal sub-sampling structure.

Signals utilizing the non-constant luminance Y'C'B C'R signal format of Recommendation ITU-R BT.601-7, Recommendation ITU-R BT.709-6, Recommendation ITU-R BT.2020-2, or Recommendation ITU-R BT.2100 shall use the appropriate one of the following values for the Media Type Parameter "sampling":

- YCbCr-4:4:4 (4:4:4 sampling)
- YCbCr-4:2:2 (4:2:2 sampling)
- YCbCr-4:2:0 (4:2:0 sampling)

Signals utilizing the Constant Luminance Y'C C'BC C'RC signal format of Recommendation ITU-R BT.2020-2 shall use the appropriate one of the following values for the Media Type Parameter "sampling":

- CLYCbCr-4:4:4 (4:4:4 sampling)
- CLYCbCr-4:2:2 (4:2:2 sampling)
- CLYCbCr-4:2:0 (4:2:0 sampling)

Signals utilizing the constant intensity I CT CP signal format of Recommendation ITU-R BT.2100 shall use the appropriate one of the following values for the Media Type Parameter "sampling":

- ICtCp-4:4:4 (4:4:4 sampling)
- ICtCp-4:2:2 (4:2:2 sampling)
- ICtCp-4:2:0 (4:2:0 sampling)

Signals utilizing the 4:4:4 R' G' B' or RGB signal format (such as that of Recommendation ITU-R BT.601, Recommendation ITU-R BT.709, Recommendation ITU-R BT.2020, Recommendation ITU-R BT.2100, SMPTE ST 2065-1 or ST 2065-3) shall use the following value for the Media Type Parameter sampling.

RGB     RGB or R' G' B' samples

Signals utilizing the 4:4:4 X' Y' Z' signal format (such as defined in SMPTE ST 428-1) shall use the following value for the Media Type Parameter sampling.

XYZ X' Y' Z' samples

Key signals as defined in SMPTE RP 157 shall use the value key for the Media Type Parameter sampling. The Key signal is represented as a single component.

KEY samples of the key signal

depth: Determines the number of bits per sample. This is an integer with typical values including 8, 10, 12, and 16.

width: Determines the number of pixels per line. This is an integer between 1 and 32767.

height: Determines the number of lines per frame. This is an integer between 1 and 32767.

exactframerate: Signals the frame rate in frames per second. Integer frame rates shall be signaled as a single decimal number (e.g. "25") whilst non-integer frame rates shall be signaled as a ratio of two integer decimal numbers separated by a "forward-slash" character (e.g. "30000/1001"), utilizing the numerically smallest numerator value possible.

colorimetry: Specifies the system colorimetry used by the image samples. Valid values and their specification are:

BT601-5	ITU Recommendation BT.601-5
BT709-2	ITU Recommendation BT.709-2
SMPTE240M	SMPTE standard 240M
BT601	as specified in Recommendation ITU-R BT.601-7
BT709	as specified in Recommendation ITU-R BT.709-6
BT2020	as specified in Recommendation ITU-R BT.2020-2
BT2100	as specified in Recommendation ITU-R BT.2100 Table 2 titled "System colorimetry"
ST2065-1	as specified in SMPTE ST 2065-1 Academy Color Encoding Specification (ACES)
ST2065-3	as specified for Academy Density Exchange Encoding (ADX) in SMPTE ST 2065-3
XYZ	as specified in ISO 11664-1 section titled "1931 Observer"

Signals utilizing the Recommendation ITU-R BT.2100 colorimetry should also signal the representational range using the optional parameter RANGE defined below.

interlace: If this OPTIONAL parameter name is present, it indicates that the video is interlaced. If this parameter name is not present, the progressive video format shall be assumed.

TCS: Transfer Characteristic System. This parameter specifies the transfer characteristic system of the image samples. Valid values and their specification are:

- SDR (Standard Dynamic Range) Video streams of standard dynamic range, that utilize the OETF of Recommendation ITU-R BT.709 or Recommendation ITU-R BT.2020. Such streams shall be assumed to target the EOTF specified in ITU-R BT.1886.
- PQ Video streams of high dynamic range video that utilize the Perceptual Quantization system of Recommendation ITU-R BT.2100
- HLG Video streams of high dynamic range video that utilize the Hybrid Log-Gamma system of Recommendation ITU-R BT.2100

RANGE: This parameter should be used to signal the encoding range of the sample values within the stream. When paired with ITU Rec BT.2100 colorimetry, this parameter has two allowed values NARROW and FULL, corresponding to the ranges specified in table 9 of ITU Rec BT.2100. In any other context, this parameter has three allowed values: NARROW, FULLPROTECT, and FULL, which correspond to the ranges specified in SMPTE RP 2077. In the absence of this parameter, NARROW shall be the assumed value in either case.

Encoding considerations:

This media type is framed and binary; see Section 4.8 in RFC 6838 [RFC6838].

Security considerations:

Please see the Security Considerations section in RFC XXXX

## 6.2. Mapping to SDP

### 6.2.1. General

A Session Description Protocol (SDP) object shall be created for each RTP stream and it shall be in accordance with the provisions of SMPTE ST 2110-10 [SMPTE-ST2110-10].

The information carried in the media type specification has a specific mapping to fields in the Session Description Protocol (SDP), which is commonly used to describe RTP sessions.

#### 6.2.2. Media type and subtype

The media type ("video") goes in SDP "m=" as the media name.

The media subtype ("jxsv") goes in SDP "a=rtpmap" as the encoding name, followed by a slash ("/") and the required parameter "rate" corresponding to the RTP timestamp clock rate (which for the payload format defined in this document SHOULD be 90000), followed by a slash ("/") and the required parameter "transmission mode" set to 1 if packets are sent sequentially by the transmitter, or 0 if transmission order is not constrained. The optional parameters go in the SDP "a=fmtp" attribute by copying them directly from the MIME media type string as a semicolon-separated list of parameter=value pairs.

A sample SDP mapping for JPEG XS video is as follows:

```
m=video 30000 RTP/AVP 112
a=rtpmap:112 jxsv/90000/1
a=fmtp:112 sampling=YCbCr-4:2:2; width=1920; height=1080;
          depth=10; colorimetry=BT709; TCS=SDR;
          RANGE=FULL; TP=2110TPNL
```

In this example, a JPEG XS RTP stream is being sent to UDP destination port 30000, with an RTP dynamic payload type of 112 and a media clock rate of 90000 Hz. Note that the "a=fmtp:" line has been wrapped to fit this page, and will be a single long line in the SDP file.

#### 6.2.3. Traffic shaping

The SDP object shall include the TP parameter (either 2110TPNL or 2110TPW as specified in Section 4.4) and may include the CMAX parameter as specified in SMPTE ST 2110-21 [SMPTE-ST2110-21].

#### 6.2.4. Offer/Answer Considerations

The following considerations apply when using SDP offer/answer procedures [RFC3264] to negotiate the use of the JPEG XS payload in RTP:

- o The "encode" parameter can be used for sendrecv, sendonly, and recvonly streams. Each encode type MUST use a separate payload type number.

- o Any unknown parameter in an offer MUST be ignored by the receiver and MUST NOT be included in the answer.

## 7. IANA Considerations

This memo requests that IANA registers video/jxsv as specified in Section 6.1. The media type is also requested to be added to the IANA registry for "RTP Payload Format MIME types" [1].

## 8. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550] and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. This implies that confidentiality of the media streams is achieved by encryption.

However, as "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format's responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity, and source authenticity for RTP in general. This responsibility lies on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms.

This payload format and the JPEG XS encoding do not exhibit any substantial non-uniformity, either in output or in complexity to perform the decoding operation and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological datagrams.

It is important to note that HD or UHDTV JPEG XS-encoded video can have significant bandwidth requirements (typically more than 1 Gbps for ultra high-definition video, especially if using high framerate). This is sufficient to cause potential for denial-of-service if transmitted onto most currently available Internet paths.

Accordingly, if best-effort service is being used, users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within acceptable parameters. Packet loss is considered acceptable if a TCP flow across the same network path, and experiencing the same network conditions, would achieve an average throughput, measured on a reasonable timescale, that is not less than the RTP flow is achieving. This condition can be satisfied by

implementing congestion control mechanisms to adapt the transmission rate (or the number of layers subscribed for a layered multicast session), or by arranging for a receiver to leave the session if the loss rate is unacceptably high.

This payload format may also be used in networks that provide quality-of-service guarantees. If enhanced service is being used, receivers SHOULD monitor packet loss to ensure that the service that was requested is actually being delivered. If it is not, then they SHOULD assume that they are receiving best-effort service and behave accordingly.

## 9. RFC Editor Considerations

Note to RFC Editor: This section may be removed after carrying out all the instructions of this section.

RFC XXXX is to be replaced by the RFC number this specification receives when published.

## 10. References

### 10.1. Normative References

[ISO21122-1]

International Organization for Standardization (ISO) - International Electrotechnical Commission (IEC), "Information technology - JPEG XS low-latency lightweight image coding system - Part 1: Core coding system", ISO/IEC PRF 21122-1, under development, <<https://www.iso.org/standard/74535.html>>.

[ISO21122-2]

International Organization for Standardization (ISO) - International Electrotechnical Commission (IEC), "Information technology - JPEG XS low-latency lightweight image coding system - Part 2: Profiles and buffer models", ISO/IEC PRF 21122-2, under development, <<https://www.iso.org/standard/74536.html>>.

[ISO21122-3]

International Organization for Standardization (ISO) - International Electrotechnical Commission (IEC), "Information technology - JPEG XS low-latency lightweight image coding system - Part 3: Transport and container formats", ISO/IEC FDIS 21122-3, under development, <<https://www.iso.org/standard/74537.html>>.

- [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>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.
- [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>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/info/rfc6838>>.
- [RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", RFC 8083, DOI 10.17487/RFC8083, March 2017, <<https://www.rfc-editor.org/info/rfc8083>>.
- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <<https://www.rfc-editor.org/info/rfc8085>>.
- [SMPTE-ST2110-10] Society of Motion Picture and Television Engineers, "SMPTE Standard - Professional Media Over Managed IP Networks: System Timing and Definitions", SMPTE ST 2110-10:2017, 2017, <<https://doi.org/10.5594/SMPTE.ST2110-10.2017>>.

[SMPTE-ST2110-21]

Society of Motion Picture and Television Engineers, "SMPTE Standard - Professional Media Over Managed IP Networks: Traffic Shaping and Delivery Timing for Video", SMPTE ST 2110-21:2017, 2017, <<https://doi.org/10.5594/SMPTE.ST2110-21.2017>>.

## 10.2. Informative References

[RFC4175] Gharai, L. and C. Perkins, "RTP Payload Format for Uncompressed Video", RFC 4175, DOI 10.17487/RFC4175, September 2005, <<https://www.rfc-editor.org/info/rfc4175>>.

[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>>.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/info/rfc5124>>.

[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>>.

## 10.3. URIs

[1] <http://www.iana.org/assignments/rtp-parameters>

## Authors' Addresses

Sebastien Lugan  
intoPIX S.A.  
Rue Emile Francqui, 9  
1435 Mont-Saint-Guibert  
Belgium

Phone: +32 10 23 84 70  
Email: [rtp@intopix.com](mailto:rtp@intopix.com)  
URI: <http://www.intopix.com>

Antonin Descampe  
intoPIX S.A.  
Rue Emile Francqui, 9  
1435 Mont-Saint-Guibert  
Belgium

Phone: +32 10 23 84 70  
Email: a.descampe@intopix.com  
URI: <http://www.intopix.com>

Corentin Damman  
intoPIX S.A.  
Rue Emile Francqui, 9  
1435 Mont-Saint-Guibert  
Belgium

Phone: +32 10 23 84 70  
Email: c.damman@intopix.com  
URI: <http://www.intopix.com>

Thomas Richter  
Fraunhofer IIS  
Am Wolfsmantel 33  
91048 Erlangen  
Germany

Phone: +49 9131 776 5126  
Email: thomas.richter@iis.fraunhofer.de  
URI: <https://www.iis.fraunhofer.de/>

Alexandre Willeme  
Universite catholique de Louvain  
Place du Levant, 2 - bte L5.04.04  
1348 Louvain-la-Neuve  
Belgium

Phone: +32 10 47 80 82  
Email: alexandre.willeme@uclouvain.be  
URI: <https://uclouvain.be/en/icteam>

payload  
Internet-Draft  
Intended status: Standards Track  
Expires: January 27, 2020

Reisenbauer  
Frequentis  
Brandhuber  
eurofunk  
Hagedorn  
Hagedorn  
Hoehnsch  
T-Systems  
Wenk  
Frequentis  
July 26, 2019

RTP Payload Format for the TETRA Audio Codec  
draft-ietf-payload-tetra-03

Abstract

This document specifies a Real-time Transport Protocol (RTP) payload format for TETRA encoded speech signals. The payload format is designed to be able to interoperate with existing TETRA transport formats on non-IP networks. A media type registration is included, specifying the use of the RTP payload format and the storage format.

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 <https://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 January 27, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1. Introduction . . . . .	2
2. Conventions Used In This Document . . . . .	3
3. Media Format Background . . . . .	3
4. Payload format . . . . .	4
4.1. RTP Header Usage . . . . .	4
4.2. Payload layout . . . . .	4
4.3. Payload Header . . . . .	5
4.3.1. I bit: Frame Indicator . . . . .	5
4.3.2. F bit: Frame Type . . . . .	6
4.3.3. CTRL: Control bit(5 bits) . . . . .	6
4.3.4. C bit: Failed Crypto operation indication . . . . .	6
4.3.5. FRAME_NR: FN (5 bits) . . . . .	7
4.3.6. R: Audio Signal Relevance (3 bits) . . . . .	7
4.3.7. S: Spare (7 bits) . . . . .	7
4.4. Payload Data . . . . .	8
5. Payload example . . . . .	8
6. Congestion Control Considerations . . . . .	8
7. Payload Format Parameters . . . . .	9
7.1. Media Type Definition . . . . .	9
8. Mapping to SDP . . . . .	10
8.1. Offer/Answer Considerations . . . . .	11
8.2. Declarative SDP Considerations . . . . .	11
9. IANA Considerations . . . . .	12
10. Security Considerations . . . . .	12
11. References . . . . .	12
11.1. Normative References . . . . .	12
11.2. Informative References . . . . .	13
Authors' Addresses . . . . .	14

## 1. Introduction

This document specifies the payload format for packetization of Terrestrial Trunked Radio (TETRA) encoded speech signals [ETSI-TETRA-Codec] into the Real-time Transport Protocol (RTP) [RFC3550]. The payload format supports transmission of multiple frames per payload, robustness against packet loss, and interoperation with existing TETRA transport formats on non-IP networks, as described in Section Section 3.

The payload format itself is specified in Section Section 4.

## 2. Conventions Used In This Document

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] when they appear in ALL CAPS. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

The following acronyms are used in this document:

- o ETSI: European Telecommunications Standards Institute
- o TETRA: TERrestrial Trunked RADio

The byte order used in this document is network byte order, i.e., the most significant byte first. The bit order is also the most significant bit first. This is presented in all figures as having the most significant bit leftmost on a line and with the lowest number. Some bit fields may wrap over multiple lines in which cases the bits on the first line are more significant than the bits on the next line.

Best current practices for writing an RTP payload format specification were followed [RFC2736] updated with [RFC8088].

## 3. Media Format Background

The TETRA codec is used as vocoder for TETRA systems. The TETRA codec is designed for compressing 30ms of audio speech data into 137 bits. The TETRA codec is designed in such a way that on the air interface two of these 30ms samples are transported together (sub-block 1 and sub-block 2). The codec allows that data of the first 30ms voice frame can be stolen and used for other purposes, e.g. for the exchange of dynamically updated key-material in end-to-end encrypted voice sessions. Codec payload serialisation is specified for TDM lines with 2048 kBit/s within traditional circuit mode based TETRA system. For this purpose two optional formats are defined [ETSI-TETRA-Codec], the first format is called FSTE (First Speech Transport Encoding Format), the other format is called OSTE (Optimized Speech Transport Encoding Format). These two formats differ mainly insofar that the OSTE format transports an additional 5 bit frame number, which provides timing information from the air interface to the receiving side in order to save the need for buffering due to different transports speed on air and in 64 kbit/s circuit switched networks. The RTP payload format is defined such that the value of this frame number can be transported.





## 4.3.2. F bit: Frame Type

Value	Frame contains
0	FSTE encoded data
1	OSTE encoded data

## 4.3.3. CTRL: Control bit(5 bits)

Ctrl 1..3 derived from the information propagated according table 5.7 of [ETSI-TETRA-ISI].

Value	Sub block 1	Sub block 2
000	normal	normal
001	C stolen	normal
010	U stolen	normal
011	C stolen	C stolen
100	C stolen	U stolen
101	U stolen	C stolen
110	U stolen	U stolen
111	O&M ISI block	

Ctrl 4..5 derived from the information propagated according table 5.7 of [ETSI-TETRA-ISI].

Value	Sub block 1	Sub block 2
00	no bad frame indicator	no bad frame indicator
01	no bad frame indicator	bad frame indicator(s)
10	bad frame indicator(s)	no bad frame indicator
11	bad frame indicator(s)	bad frame indicator(s)

NOTE: The interpretation of C4 and C5 is outside the scope of the present document

## 4.3.4. C bit: Failed Crypto operation indication

This bit may be set to "1" if a decryption (encrypted audio along the circuit switched mobile network, decryption at the RTP sender forwarding this audio) operation could not be performed successfully for the specific half-block. Consequently, the encryption status of

the half-block audio data is unknown. Implementation of an RTP receiver has to take into account "C bit" when forwarding such TETRA audio data (either to a decoder directly or via TETRA infrastructure to a TETRA mobile unit), the contained audio might be scrambled - depending if the audio originally was generated as a plain-override half-block or as an encrypted half-block.

#### 4.3.5. FRAME\_NR: FN (5 bits)

The frame number bits contain an uplink frame number as defined in table 5.3 of [ETSI-TETRA-ISI]. If no frame number is available the FRAME\_NR value SHALL be set to 00000.

#### 4.3.6. R: Audio Signal Relevance (3 bits)

The Audio Signal Relevance bits contain information about the Relevance of the voice packet contained here.

R 1

0: no audio signal relevance propagated (R2 and R3 do not contain any valid information)

1: audio signal relevance propagated in R2 and R3

R 2..3 According to table 1 of [BDBOS-BIP20]

value	relevance
00	no audio signal relevance (level $\leq$ -72 dBm0)
01	low audio signal relevance (-52dBm0 $\leq$ level $<$ -72dBm0)
10	medium audio signal relevance (-32dBm0 $\leq$ level $<$ -52dBm0)
11	high audio signal relevance (0dBm0 $\leq$ level $<$ -32dBm0)

NOTE: Receiver SHOULD consider stolen or erroneous blocks as not available for audio decoding as indicated by control bits independent of audio signal relevance bits.

#### 4.3.7. S: Spare (7 bits)

The S bits bits are reserved for future use and set to "0" currently.



codec control layer, and the lower-level transport interface, as well as components dedicated to congestion control functions.

Congestion control for RTP SHALL be used in accordance with RFC 3550 [RFC3550], and with any applicable RTP profile; e.g., RFC 3551 [RFC3551]. An additional requirement if best-effort service is being used is: users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within acceptable parameters.

## 7. Payload Format Parameters

This RTP payload format is identified using one media subtype (audio/TETRA) which is registered in accordance with RFC 4855 [RFC4855] and per media type registration template from RFC 6838 [RFC6838].

### 7.1. Media Type Definition

The media type for the TETRA codec is expected to be allocated from the IETF tree once this draft turns into an RFC. This media type registration covers both real-time transfer via RTP and non-real-time transfers via stored files.

Type name:

audio

Subtype name:

TETRA

Required parameters:

none

Optional parameters:

These parameters apply to RTP transfer only.

- \* maxptime: The maximum amount of media which can be encapsulated in a payload packet, expressed as time in milliseconds. The time is calculated as the sum of the time that the media present in the packet represents. The time SHOULD be an integer multiple of the frame size. If this parameter is not present, the sender MAY encapsulate any number of speech frames into one RTP packet.

- \* ptime: see RFC 4566 [RFC4566].

Encoding considerations:

This media type is framed and binary according Section 4.8 of RFC 6838 [RFC6838].

Security considerations:

See Section Section 10 of RFC XXXX. [RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Interoperability considerations: N/A

## Published specification:

RFC XXXX [RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

## Applications that use this media type:

This media type is used in applications needing transport or storage of encoded voice. Some examples include; Voice over IP, streaming media, voice messaging, and voice recording on recording systems.

## Additional Information:

- Deprecated alias names for this type: N/A
- Magic number(s): N/A
- File extension(s): N/A
- Macintosh file type code(s): N/A

## Person &amp; email address to contact for further information:

Andreas Reisenbauer <mailto:andreas.reisenbauer@frequentis.com>  
IETF Payload Working Group <mailto:payload@ietf.org>

## Intended usage:

COMMON

## Restrictions on usage:

This media subtype depends on RTP framing and hence is only defined for transfer via RTP RFC 3550 [RFC3550]. Transport within other framing protocols is not defined at this time.

## Author:

Andreas Reisenbauer <mailto:andreas.reisenbauer@frequentis.com>

## Change controller:

The IETF PAYLOAD Working Group, or other party as designated by the IESG.

## 8. Mapping to SDP

The information carried in the media type specification has a specific mapping to fields in the Session Description Protocol [RFC4566], which is commonly used to describe RTP sessions. When SDP is used to specify sessions employing the TETRA codec, the mapping is as follows:

## Media Type name:

audio

## Media subtype name:

TETRA

## Required parameters:

none

## Optional parameters:

none

#### Mapping Parameters into SDP

The information carried in the media type specification has a specific mapping to fields in the Session Description Protocol [RFC4566], which is commonly used to describe RTP sessions. When SDP is used to specify sessions employing the TETRA codec, the mapping is as follows:

- \* The media type ("audio") goes in SDP "m=" as the media name.
- \* The media subtype (payload format name) goes in SDP "a=rtpmap" as the encoding name. The RTP clock rate in "a=rtpmap" MUST be 8000.
- \* The parameters "ptime" and "maxptime" go in the SDP "a=ptime" and "a=maxptime" attributes, respectively.
- \* Any remaining parameters go in the SDP "a=fmtp" attribute by copying them directly from the media type parameter string as a semicolon-separated list of parameter=value pairs.

Here is an example SDP session of usage of TETRA:

```
m=audio 49120 RTP/AVP 99
a=rtpmap:99 TETRA/8000
a=maxptime:60
a=ptime:60
```

#### 8.1. Offer/Answer Considerations

The following considerations apply when using SDP Offer-Answer procedures to negotiate the use of TETRA payload in RTP:

- o In most cases, the parameters "maxptime" and "ptime" will not affect interoperability; however, the setting of the parameters can affect the performance of the application. The SDP offer-answer handling of the "ptime" and "maxptime" parameter is described in RFC3264 [RFC3264].
- o Integer multiples of 30ms SHALL be used for ptime. It is recommended to use packet size of 60ms. There is no need that ptime and maxptime parameters are negotiated symmetrically.
- o Any unknown parameter in an offer SHALL be removed in the answer.

#### 8.2. Declarative SDP Considerations

For declarative media, the "ptime" and "maxptime" parameter specify the possible variants used by the sender.

## 9. IANA Considerations

This memo requests that IANA registers [audio/TETRA] from section Section 7.1. The media type is also requested to be added to the IANA registry for "RTP Payload Format MIME types" (<<http://www.iana.org/assignments/rtp-parameters>>).

## 10. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550] , and in any applicable RTP profile. The main security considerations for the RTP packet carrying the RTP payload format defined within this memo are confidentiality, integrity and source authenticity. Confidentiality is achieved by encryption of the RTP payload. Integrity of the RTP packets through suitable cryptographic integrity protection mechanism. Cryptographic systems may also allow the authentication of the source of the payload. A suitable security mechanism for this RTP payload format should provide confidentiality, integrity protection and at least source authentication capable of determining if an RTP packet is from a member of the RTP session or not.

Note that the appropriate mechanism to provide security to RTP and payloads following this memo may vary. It is dependent on the application, the transport, and the signaling protocol employed. Therefore a single mechanism is not sufficient, although if suitable the usage of SRTP [RFC3711] is recommended. Other mechanism that may be used are IPsec [RFC4301] and TLS [RFC5246] (RTP over TCP), but also other alternatives may exist.

## 11. References

### 11.1. Normative References

#### [BDBOS-BIP20]

BDBOS, "BIP 20 QOS Dienstguete-Parameter BOS- Interoperabilitaetsprofil fuer Endgeraete zur Nutzung im Digitalfunk BOS; Version 2014-04 - Revision 2", 2014.

#### [ETSI-TETRA-Codec]

ETSI, "EN 300 395-2; Terrestrial Trunked Radio (TETRA); Speech codec for full-rate traffic channel; Part 2: TETRA codec V1.3.1", 2005, <[http://www.etsi.org/deliver/etsi\\_en/300300\\_300399/30039502/01.03.01\\_60/en\\_30039502v010301p.pdf](http://www.etsi.org/deliver/etsi_en/300300_300399/30039502/01.03.01_60/en_30039502v010301p.pdf)>.

## [ETSI-TETRA-ISI]

ETSI, "TS 100 392-3-8; Terrestrial Trunked Radio (TETRA); Voice plus Data (V+D); Part 3: Interworking at the Inter-System Interface (ISI); Sub-part 8: Generic Speech Format Implementation V1.3.1", 2018, <[https://www.etsi.org/deliver/etsi\\_ts/100300\\_100399/1003920308/01.03.01\\_60/ts\\_1003920308v010301p.pdf](https://www.etsi.org/deliver/etsi_ts/100300_100399/1003920308/01.03.01_60/ts_1003920308v010301p.pdf)>.

- [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>>.
- [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>>.
- [RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", RFC 8083, DOI 10.17487/RFC8083, March 2017, <<https://www.rfc-editor.org/info/rfc8083>>.
- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <<https://www.rfc-editor.org/info/rfc8085>>.

## 11.2. Informative References

- [RFC2736] Handley, M. and C. Perkins, "Guidelines for Writers of RTP Payload Format Specifications", BCP 36, RFC 2736, DOI 10.17487/RFC2736, December 1999, <<https://www.rfc-editor.org/info/rfc2736>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.

- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<https://www.rfc-editor.org/info/rfc4301>>.
- [RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <<https://www.rfc-editor.org/info/rfc4855>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/info/rfc6838>>.
- [RFC8088] Westerlund, M., "How to Write an RTP Payload Format", RFC 8088, DOI 10.17487/RFC8088, May 2017, <<https://www.rfc-editor.org/info/rfc8088>>.
- [RMCAT] IETF, "RTP Media Congestion Avoidance Techniques (rmcat) Working Group", 2018, <<https://datatracker.ietf.org/wg/rmcat/about/>>.

## Authors' Addresses

Andreas Reisenbauer  
Frequentis AG  
Innovationsstr. 1  
Vienna 1100  
Austria

Email: [andreas.reisenbauer@frequentis.com](mailto:andreas.reisenbauer@frequentis.com)

Udo Brandhuber  
eurofunk Kappacher GmbH  
Germany

Email: [ubrandhuber@eurofunk.com](mailto:ubrandhuber@eurofunk.com)

Joachim Hagedorn  
Hagedorn Informationssysteme GmbH  
Germany

Email: joachim@hagedorn-infosysteme.de

Klaus-Peter Hoehnsch  
T-Systems International GmbH  
Germany

Email: klaus-peter.hoehnsch@t-systems.com

Stefan Wenk  
Frequentis AG  
Innovationsstr. 1  
Vienna 1100  
Austria

Email: stefan.wenk@frequentis.com

Payload Working Group  
Internet-Draft  
Intended Status: Standards Track  
Expires: October 8, 2020

Victor Demjanenko  
John Punaro  
David Satterlee  
VOCAL Technologies, Ltd.  
April 10, 2020

RTP Payload Format for TSV CIS Codec  
draft-ietf-payload-tsvcis-05

Status of This Memo

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

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

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

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."

Abstract

This document describes the RTP payload format for the Tactical Secure Voice Cryptographic Interoperability Specification (TSVCIS) speech coder. TSV CIS is a scalable narrowband voice coder supporting varying encoder data rates and fallbacks. It is implemented as an augmentation to the Mixed Excitation Linear Prediction Enhanced (MELPe) speech coder by conveying additional speech coder parameters for enhancing voice quality. TSV CIS augmented speech data is

processed in conjunction with its temporal matched MELP 2400 speech data. The RTP packetization of TSV CIS and MELPe speech coder data is described in detail.

Table of Contents

1. Introduction . . . . . 2  
 1.1. Conventions . . . . . 3  
 2. Background . . . . . 3  
 3. Payload Format . . . . . 4  
 3.1. MELPe Bitstream Definitions . . . . . 5  
 3.1.1. 2400 bps Bitstream Structure . . . . . 6  
 3.1.2. 1200 bps Bitstream Structure . . . . . 6  
 3.1.3. 600 bps Bitstream Structure . . . . . 7  
 3.1.4. Comfort Noise Bitstream Definition . . . . . 8  
 3.2. TSV CIS Bitstream Definition . . . . . 8  
 3.3. Multiple TSV CIS Frames in an RTP Packet . . . . . 10  
 3.4. Congestion Control Considerations . . . . . 11  
 4. Payload Format Parameters . . . . . 11  
 4.1. Media Type Definitions . . . . . 11  
 4.2. Mapping to SDP . . . . . 13  
 4.3. Declarative SDP Considerations . . . . . 15  
 4.4. Offer/Answer SDP Considerations . . . . . 15  
 5. Discontinuous Transmissions . . . . . 16  
 6. Packet Loss Concealment . . . . . 16  
 7. IANA Considerations . . . . . 16  
 8. Security Considerations . . . . . 16  
 10. References . . . . . 17  
 10.1. Normative References . . . . . 17  
 10.2. Informative References . . . . . 19  
 Authors' Addresses . . . . . 19

1. Introduction

This document describes how compressed Tactical Secure Voice Cryptographic Interoperability Specification (TSV CIS) speech as produced by the TSV CIS codec [TSV CIS] [NRLVDR] may be formatted for use as an RTP payload. The TSV CIS speech coder (or TSV CIS speech aware communications equipment on any intervening transport link) may adjust to restricted bandwidth conditions by reducing the amount of augmented speech data and relying on the underlying MELPe speech coder for the most constrained bandwidth links.

Details are provided for packetizing the TSV CIS augmented speech data along with MELPe 2400 bps speech parameters in a RTP packet. The sender may send one or more codec data frames per packet, depending on the application scenario or based on transport network conditions,

bandwidth restrictions, delay requirements, and packet loss tolerance.

## 1.1. Conventions

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.

Best current practices for writing an RTP payload format specification were followed [RFC2736] [RFC8088].

## 2. Background

The MELP speech coder was developed by the US military as an upgrade from the LPC-based CELP standard vocoder for low-bitrate communications [MELP]. ("LPC" stands for "Linear-Predictive Coding", and "CELP" stands for "Code-Excited Linear Prediction".) MELP was further enhanced and subsequently adopted by NATO as MELPe for use by its members and Partnership for Peace countries for military and other governmental communications as international NATO Standard STANAG 4591 [MELPE].

The Tactical Secure Voice Cryptographic Interoperability Specification (TSVCIS) is a specification written by the Tactical Secure Voice Working Group (TSVWG) for enabling all modern tactical secure voice devices to be interoperable across the Department of Defense [TSVCIS]. One of the most important aspects is that the voice modes defined in TSVCIS are based on specific fixed rates of Naval Research Lab's (NRL's) Variable Data Rate (VDR) Vocoder which uses the MELPe standard as its base [NRLVDR]. A complete TSVCIS speech frame consists of MELPe speech parameters and corresponding TSVCIS augmented speech data.

In addition to the augmented speech data, the TSVCIS specification identifies which speech coder and framing bits are to be encrypted, and how they are protected by forward error correction (FEC) techniques (using block codes). At the RTP transport layer, only the speech-coder-related bits need to be considered and are conveyed in unencrypted form. In most IP-based network deployments, standard link encryption methods (SRTP, VPNs, FIPS 140 link encryptors or Type 1 Ethernet encryptors) would be used to secure the RTP speech contents.

TSVCIS augmented speech data is derived from the signal processing and data already performed by the MELPe speech coder. For the



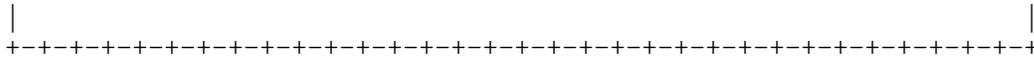


Figure 1: Packet Format Diagram

The RTP header of the packetized encoded TSVCIS speech has the expected values as described in [RFC3550]. The usage of the M bit SHOULD be as specified in the applicable RTP profile -- for example, [RFC3551], where [RFC3551] specifies that if the sender does not suppress silence (i.e., sends a frame on every frame interval), the M bit will always be zero. When more than one codec data frame is present in a single RTP packet, the timestamp specified is that of the oldest data frame represented in the RTP packet.

The assignment of an RTP payload type for this new packet format is outside the scope of this document and will not be specified here. It is expected that the RTP profile for a particular class of applications will assign a payload type for this encoding, or if that is not done, then a payload type in the dynamic range shall be chosen by the sender.

3.1. MELPe Bitstream Definitions

The TSVCIS speech coder includes all three MELPe coder rates used as base speech parameters or as speech coders for bandwidth restricted links. RTP packetization of MELPe follows RFC 8130 and is repeated here for all three MELPe rates [RFC8130] with its recommendations now regarded as requirements. The bits previously labeled as RSVA, RSVB, and RSVC in RFC 8130 SHOULD be filled with rate coding, CODA, CODB, and CODC, as shown in Table 1 (compatible with Table 7 in Section 3.3 of [RFC8130]).

Coder Bitrate	CODA	CODB	CODC	Length
2400 bps	0	0	N/A	7
1200 bps	1	0	0	11
600 bps	0	1	N/A	7
Comfort Noise	1	0	1	2
TSVCIS data	1	1	N/A	var.

Table 1: TSVCIS/MELPe Frame Bitrate Indicators and Frame Length

The total number of bits used to describe one MELPe frame of 2400 bps speech is 54, which fits in 7 octets (with two rate code bits). For MELPe 1200 bps speech, the total number of bits used is 81, which fits in 11 octets (with three rate code bits and four unused bits). For MELPe 600 bps speech, the total number of bits used is 54, which fits in 7 octets (with two rate code bits). The comfort noise frame consists of 13 bits, which fits in 2 octets (with three rate code bits). TSVCIIS packed parameters will use the last code combination in a trailing byte as discussed in Section 3.2.

It should be noted that CODB for MELPe 600 bps mode MAY deviate from the value in Table 1 when bit 55 is used as an alternating 1/0 end-to-end framing bit. Frame decoding would remain distinct as CODA being zero on its own would indicate a 7-byte frame for either 2400 or 600 bps rate and the use of 600 bps speech coding could be deduced from the RTP timestamp (and anticipated by the SDP negotiations).

3.1.1. 2400 bps Bitstream Structure

The 2400 bps MELPe RTP payload is constructed as per Figure 2. Note that CODA MUST be filled with 0 and CODB SHOULD be filled with 0 as per Section 3.1. CODB MAY contain an end-to-end framing bit if required by the endpoints.

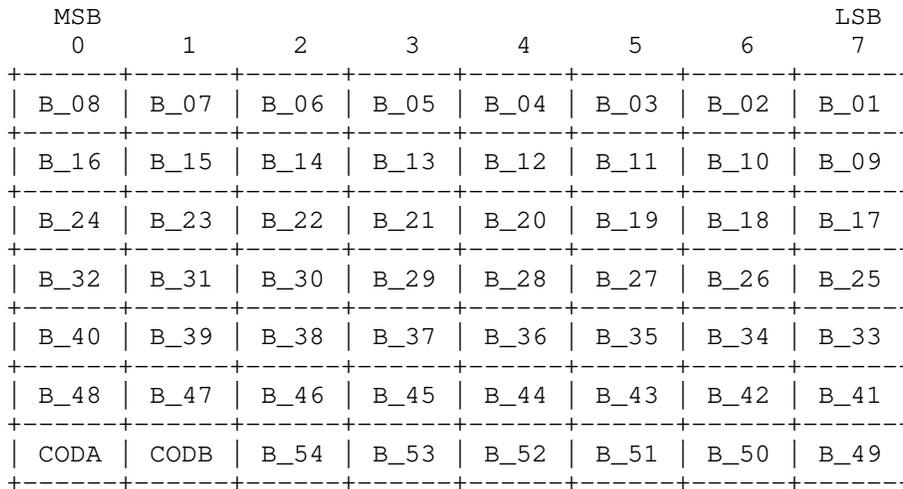


Figure 2: Packed MELPe 2400 bps Payload Octets

3.1.2. 1200 bps Bitstream Structure

The 1200 bps MELPe RTP payload is constructed as per Figure 3. Note that CODA, CODB, and CODC MUST be filled with 1, 0, and 0

respectively as per Section 3.1. RSV0 MUST be coded as 0.

MSB								LSB
0	1	2	3	4	5	6	7	
B_08	B_07	B_06	B_05	B_04	B_03	B_02	B_01	
B_16	B_15	B_14	B_13	B_12	B_11	B_10	B_09	
B_24	B_23	B_22	B_21	B_20	B_19	B_18	B_17	
B_32	B_31	B_30	B_29	B_28	B_27	B_26	B_25	
B_40	B_39	B_38	B_37	B_36	B_35	B_34	B_33	
B_48	B_47	B_46	B_45	B_44	B_43	B_42	B_41	
B_56	B_55	B_54	B_53	B_52	B_51	B_50	B_49	
B_64	B_63	B_62	B_61	B_60	B_59	B_58	B_57	
B_72	B_71	B_70	B_69	B_68	B_67	B_66	B_65	
B_80	B_79	B_78	B_77	B_76	B_75	B_74	B_73	
CODA	CODB	CODC	RSV0	RSV0	RSV0	RSV0	B_81	

Figure 3: Packed MELPe 1200 bps Payload Octets

### 3.1.3. 600 bps Bitstream Structure

The 600 bps MELPe RTP payload is constructed as per Figure 4. Note CODA MUST be filled with 0 and CODB SHOULD be filled with 1 as per Section 3.1. CODB MAY contain an end-to-end framing bit if required by the endpoints.

MSB								LSB
0	1	2	3	4	5	6	7	
B_08	B_07	B_06	B_05	B_04	B_03	B_02	B_01	
B_16	B_15	B_14	B_13	B_12	B_11	B_10	B_09	
B_24	B_23	B_22	B_21	B_20	B_19	B_18	B_17	
B_32	B_31	B_30	B_29	B_28	B_27	B_26	B_25	

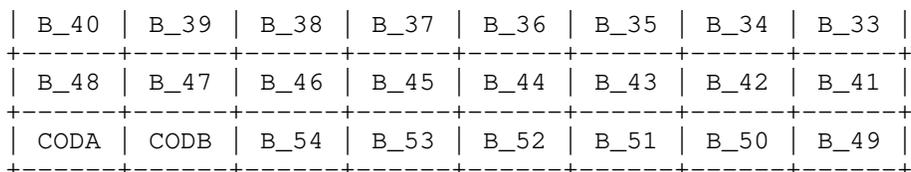


Figure 4: Packed MELPe 600 bps Payload Octets

## 3.1.4. Comfort Noise Bitstream Definition

The comfort noise MELPe RTP payload is constructed as per Figure 5. Note that CODA, CODB, and CODC MUST be filled with 1, 0, and 1 respectively as per Section 3.1.

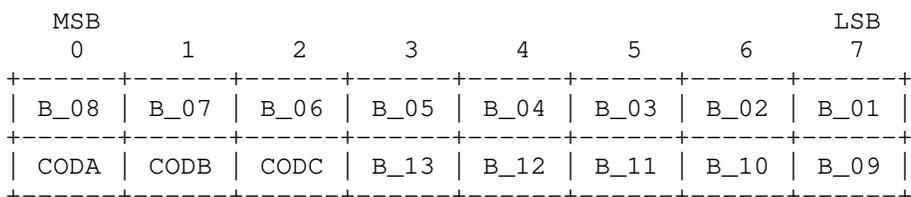


Figure 5: Packed MELPe Comfort Noise Payload Octets

## 3.2. TSVCIS Bitstream Definition

The TSVCIS augmented speech data as packed parameters MUST be placed immediately after a corresponding MELPe 2400 bps payload in the same RTP packet. The packed parameters are counted in octets (TC). The preferred placement SHOULD be used for TSVCIS payloads with TC less than or equal to 77 octets, and is shown in Figure 6. In the preferred placement, a single trailing octet SHALL be appended to include a two-bit rate code, CODA and CODB, (both bits set to one) and a six-bit modified count (MTC). The special modified count value of all ones (representing a MTC value of 63) SHALL NOT be used for this format as it is used as the indicator for the alternate packing format shown next. In a standard implementation, the TSVCIS speech coder uses a minimum of 15 octets for parameters in octet packed form. The modified count (MTC) MUST be reduced by 15 from the full octet count (TC). Computed MTC = TC-15. This accommodates a maximum of 77 parameter octets (maximum value of MTC is 62, 77 is the sum of 62+15).



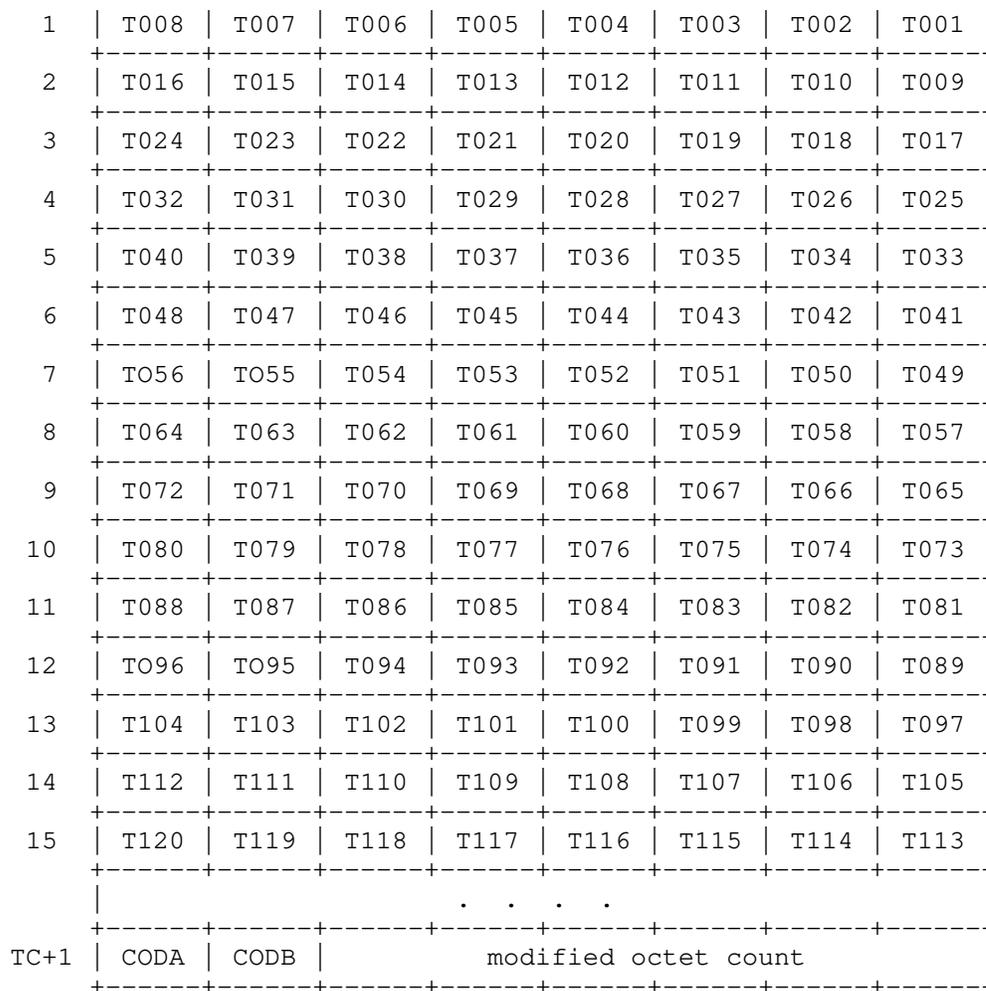
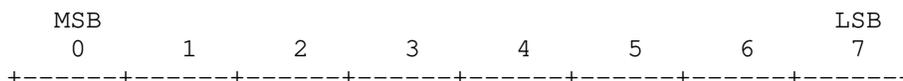


Figure 6: Preferred Packed TSVCIIS Payload Octets

In order to accommodate all other NRL VDR configurations, an alternate parameter placement MUST use two trailing bytes as shown in Figure 7. The last trailing byte MUST be filled with a two-bit rate code, CODA and CODB, (both bits set to one) and its six-bit count field MUST be filled with ones. The second to last trailing byte MUST contain the parameter count (TC) in octets (a value from 1 and 255, inclusive). The value of zero SHALL be considered as reserved.



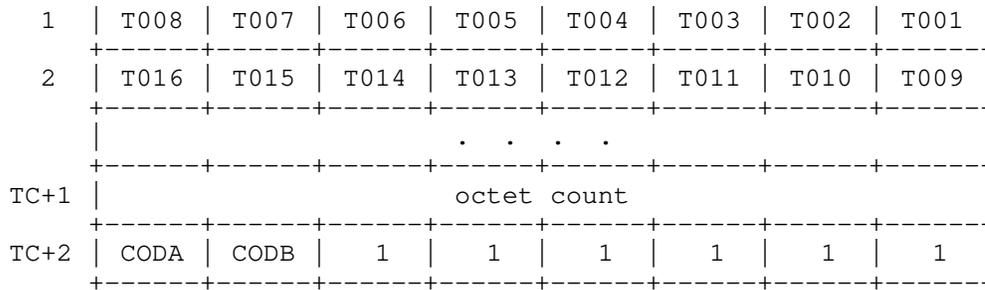


Figure 7: Length Unrestricted Packed TSVCSIS Payload Octets

### 3.3. Multiple TSVCSIS Frames in an RTP Packet

A TSVCSIS RTP packet payload consists of zero or more consecutive TSVCSIS coder frames (each consisting of MELPe 2400 and TSVCSIS coder data), with the oldest frame first, followed by zero or one MELPe comfort noise frame. The presence of a comfort noise frame can be determined by its rate code bits in its last octet.

The default packetization interval is one coder frame (22.5, 67.5, or 90 ms) according to the coder bitrate (2400, 1200, or 600 bps). For some applications, a longer packetization interval is used to reduce the packet rate.

A TSVCSIS RTP packet without coder and comfort noise frames MAY be used periodically by an endpoint to indicate connectivity by an otherwise idle receiver.

TSVCSIS coder frames in a single RTP packet MAY have varying TSVCSIS parameter octet counts. Its packed parameter octet count (length) is indicated in the trailing byte(s). All MELPe frames in a single RTP packet MUST be of the same coder bitrate. For all MELPe coder frames, the coder rate bits in the trailing byte identify the contents and length as per Table 1.

It is important to observe that senders have the following additional restrictions:

Senders SHOULD NOT include more TSVCSIS or MELPe frames in a single RTP packet than will fit in the MTU of the RTP transport protocol.

Frames MUST NOT be split between RTP packets.

It is RECOMMENDED that the number of frames contained within an RTP packet be consistent with the application. For example, in telephony and other real-time applications where delay is important, then the

fewer frames per packet the lower the delay, whereas for bandwidth-constrained links or delay-insensitive streaming messaging applications, more than one frame per packet or many frames per packet would be acceptable.

Information describing the number of frames contained in an RTP packet is not transmitted as part of the RTP payload. The way to determine the number of TSV CIS/MELPe frames is to identify each frame type and length thereby counting the total number of octets within the RTP packet.

### 3.4. Congestion Control Considerations

The target bitrate of TSV CIS can be adjusted at any point in time, thus allowing congestion management. Furthermore, the amount of encoded speech or audio data encoded in a single packet can be used for congestion control, since the packet rate is inversely proportional to the packet duration. A lower packet transmission rate reduces the amount of header overhead but at the same time increases latency and loss sensitivity, so it ought to be used with care.

Since UDP does not provide congestion control, applications that use RTP over UDP SHOULD implement their own congestion control above the UDP layer [RFC8085] and MAY also implement a transport circuit breaker [RFC8083]. Work in the RMCAT working group [RMCAT] describes the interactions and conceptual interfaces necessary between the application components that relate to congestion control, including the RTP layer, the higher-level media codec control layer, and the lower-level transport interface, as well as components dedicated to congestion control functions.

## 4. Payload Format Parameters

This RTP payload format is identified using the TSV CIS media subtype, which is registered in accordance with RFC 4855 [RFC4855] and per the media type registration template from RFC 6838 [RFC6838].

### 4.1. Media Type Definitions

Type name: audio

Subtype name: TSV CIS

Required parameters: N/A

Optional parameters:

ptime: the recommended length of time (in milliseconds) represented by the media in a packet. It SHALL use the nearest rounded-up ms integer packet duration. For TSVCIS, this corresponds to the following values: 23, 45, 68, 90, 112, 135, 156, and 180. Larger values can be used as long as they are properly rounded. See Section 6 of RFC 4566 [RFC4566].

maxptime: the maximum length of time (in milliseconds) that can be encapsulated in a packet. It SHALL use the nearest rounded-up ms integer packet duration. For TSVCIS, this corresponds to the following values: 23, 45, 68, 90, 112, 135, 156, and 180. Larger values can be used as long as they are properly rounded. See Section 6 of RFC 4566 [RFC4566].

bitrate: specifies the MELPe coder bitrates supported. Possible values are a comma-separated list of rates from the following set: 2400, 1200, 600. The modes are listed in order of preference; first is preferred. If "bitrate" is not present, the fixed coder bitrate of 2400 MUST be used.

tcmax: specifies the TSVCIS maximum value for TC supported or desired ranging from 1 to 255. If "tcmax" is not present, a default value of 35 is used.

Encoding considerations: This media subtype is framed and binary; see Section 4.8 of RFC 6838 [RFC6838].

Security considerations: Please see Section 8 of RFC XXXX.

[EDITOR NOTE - please replace XXXX with the RFC number of this document.]

Interoperability considerations: N/A

Published specification: [TSVCIS]

Applications that use this media type: N/A

Fragment identifier considerations: N/A

Additional information:

Clock Rate (Hz): 8000  
Channels: 1

Deprecated alias names for this type: N/A

Magic number(s): N/A

File extension(s): N/A

Macintosh file type code(s): N/A

Person & email address to contact for further information:

Victor Demjanenko, Ph.D.  
VOCAL Technologies, Ltd.  
520 Lee Entrance, Suite 202  
Buffalo, NY 14228  
United States of America  
Phone: +1 716 688 4675  
Email: victor.demjanenko@vocal.com

Intended usage: COMMON

Restrictions on usage: The media subtype depends on RTP framing and hence is only defined for transfer via RTP [RFC3550]. Transport within other framing protocols is not defined at this time.

Author: Victor Demjanenko

Change controller: IETF, contact <avt@ietf.org>

Provisional registration? (standards tree only): No

#### 4.2. Mapping to SDP

The mapping of the above-defined payload format media subtype and its parameters SHALL be done according to Section 3 of RFC 4855 [RFC4855].

The information carried in the media type specification has a specific mapping to fields in the Session Description Protocol (SDP) [RFC4566], which is commonly used to describe RTP sessions. When SDP is used to specify sessions employing the TSVCIIS codec, the mapping is as follows:

- o The media type ("audio") goes in SDP "m=" as the media name.
- o The media subtype (payload format name) goes in SDP "a=rtpmap" as the encoding name.
- o The parameter "bitrate" goes in the SDP "a=fmtp" attribute by copying it as a "bitrate=<value>" string.
- o The parameter "tcmx" goes in the SDP "a=fmtp" attribute by copying it as a "tcmx=<value>" string.

- o The parameters "ptime" and "maxptime" go in the SDP "a=ptime" and "a=maxptime" attributes, respectively.

When conveying information via SDP, the encoding name SHALL be "TSVCIS" (the same as the media subtype).

An example of the media representation in SDP for describing TSVCIS might be:

```
m=audio 49120 RTP/AVP 96
a=rtpmap:96 TSVCIS/8000
```

The optional media type parameter "bitrate", when present, MUST be included in the "a=fmtp" attribute in the SDP, expressed as a media type string in the form of a semicolon-separated list of parameter=value pairs. The string "value" can be one or more of 2400, 1200, and 600, separated by commas (where each bitrate value indicates the corresponding MELPe coder). An example of the media representation in SDP for describing TSVCIS when all three coder bitrates are supported might be:

```
m=audio 49120 RTP/AVP 96
a=rtpmap:96 TSVCIS/8000
a=fmtp:96 bitrate=2400,600,1200
```

The optional media type parameter "tcmx", when present, MUST be included in the "a=fmtp" attribute in the SDP, expressed as a media type string in the form of a semicolon-separated list of parameter=value pairs. The string "value" is an integer number in the range of 1 to 255 representing the maximum number of TSVCIS parameter octets supported. An example of the media representation in SDP for describing TSVCIS with a maximum of 101 octets supported is as follows:

```
m=audio 49120 RTP/AVP 96
a=rtpmap:96 TSVCIS/8000
a=fmtp:96 tcmx=101
```

The parameter "ptime" cannot be used for the purpose of specifying the TSVCIS operating mode, due to the fact that for certain values it will be impossible to distinguish which mode is about to be used (e.g., when ptime=68, it would be impossible to distinguish if the packet is carrying one frame of 67.5 ms or three frames of 22.5 ms).

Note that the payload format (encoding) names are commonly shown in upper case. Media subtypes are commonly shown in lower case. These names are case insensitive in both places. Similarly, parameter names are case insensitive in both the media subtype name and the

default mapping to the SDP a=fmtp attribute.

#### 4.3. Declarative SDP Considerations

For declarative media, the "bitrate" parameter specifies the possible bitrates used by the sender. Multiple TSVCSIS rtpmap values (such as 97, 98, and 99, as used below) MAY be used to convey TSVCSIS-coded voice at different bitrates. The receiver can then select an appropriate TSVCSIS codec by using 97, 98, or 99.

```
m=audio 49120 RTP/AVP 97 98 99
a=rtpmap:97 TSVCSIS/8000
a=fmtp:97 bitrate=2400
a=rtpmap:98 TSVCSIS/8000
a=fmtp:98 bitrate=1200
a=rtpmap:99 TSVCSIS/8000
a=fmtp:99 bitrate=600
```

For declarative media, the "tcmx" parameter specifies the maximum number of TSVCSIS packed parameter octets used by the sender or the sender's communications channel.

#### 4.4. Offer/Answer SDP Considerations

In the Offer/Answer model [RFC3264], "bitrate" is a bidirectional parameter. Both sides MUST use a common "bitrate" value or values. The offer contains the bitrates supported by the offerer, listed in its preferred order. The answerer MAY agree to any bitrate by listing the bitrate first in the answerer response. Additionally, the answerer MAY indicate any secondary bitrate or bitrates that it supports. The initial bitrate used by both parties SHALL be the first bitrate specified in the answerer response.

For example, if offerer bitrates are "2400,600" and answer bitrates are "600,2400", the initial bitrate is 600. If other bitrates are provided by the answerer, any common bitrate between the offer and answer MAY be used at any time in the future. Activation of these other common bitrates is beyond the scope of this document.

The use of a lower bitrate is often important for a case such as when one endpoint utilizes a bandwidth-constrained link (e.g., 1200 bps radio link or slower), where only the lower coder bitrate will work.

In the Offer/Answer model [RFC3264], "tcmx" is a bidirectional parameter. Both sides SHOULD use a common "tcmx" value. The offer contains the tcmx supported by the offerer. The answerer MAY agree to any tcmx equal or less than this value by stating the desired tcmx in the answerer response. The answerer alternatively MAY

identify its own tcm<sub>ax</sub> and rely on TSVCIIS ignoring any augmented data it cannot use.

#### 5. Discontinuous Transmissions

A primary application of TSVCIIS is for radio communications of voice conversations, and discontinuous transmissions are normal. When TSVCIIS is used in an IP network, TSVCIIS RTP packet transmissions may cease and resume frequently. RTP synchronization source (SSRC) sequence number gaps indicate lost packets to be filled by Packet Loss Concealment (PLC), while abrupt loss of RTP packets indicates intended discontinuous transmissions. Resumption of voice transmission SHOULD be indicated by the RTP marker bit (M) set to 1.

If a TSVCIIS coder so desires, it may send a MELPe comfort noise frame as per Appendix B of [SCIP210] prior to ceasing transmission. A receiver may optionally use comfort noise during its silence periods. No SDP negotiations are required.

#### 6. Packet Loss Concealment

TSVCIIS packet loss concealment (PLC) uses the special properties and coding for the pitch/voicing parameter of the MELPe 2400 bps coder. The PLC erasure indication utilizes any of the errored encodings of a non-voiced frame as identified in Table 1 of [MELPE]. For the sake of simplicity, it is preferred that a code value of 3 for the pitch/voicing parameter be used. Hence, set bits P0 and P1 to one and bits P2, P3, P4, P5, and P6 to zero.

When using PLC in 1200 bps or 600 bps mode, the MELPe 2400 bps decoder is called three or four times, respectively, to cover the loss of a low bitrate MELPe frame.

#### 7. IANA Considerations

This memo requests that IANA registers TSVCIIS as specified in Section 4.1. The media type is also requested to be added to the IANA registry for "RTP Payload Format MIME types" (<http://www.iana.org/assignments/rtp-parameters>).

#### 8. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550] and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. However, as discussed in [RFC7202], it is not an RTP payload format's responsibility to discuss or mandate what

solutions are used to meet such basic security goals as confidentiality, integrity, and source authenticity for RTP in general. This responsibility lies with anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this section discusses the security-impacting properties of the payload format itself.

This RTP payload format and the TSVCS decoder, to the best of our knowledge, do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Additionally, the RTP payload format does not contain any active content.

Please see the security considerations discussed in [RFC6562] regarding Voice Activity Detect (VAD) and its effect on bitrates.

## 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, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017, <<http://www.rfc-editor.org/info/rfc8174>>.
- [RFC2736] Handley, M. and C. Perkins, "Guidelines for Writers of RTP Payload Format Specifications", BCP 36, RFC 2736, DOI 10.17487/RFC2736, December 1999, <<http://www.rfc-editor.org/info/rfc2736>>.
- [RFC8088] Westerlund, M., "How to Write an RTP Payload Format", RFC 8088, DOI 10.17487/RFC8088, May 2017, <<http://www.rfc-editor.org/info/rfc8088>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<http://www.rfc-editor.org/info/rfc3264>>.

- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<http://www.rfc-editor.org/info/rfc3550>>.
- [RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <<http://www.rfc-editor.org/info/rfc3551>>.
- [RFC8130] Demjanenko, V., and D. Satterlee, "RTP Payload Format for the Mixed Excitation Linear Prediction Enhanced (MELPe) Codec", RFC 8130, DOI 10.tbd/RFC8130, March 2017, <<http://www.rfc-editor.org/info/rfc8130>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<http://www.rfc-editor.org/info/rfc3711>>.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <<http://www.rfc-editor.org/info/rfc4566>>.
- [RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <<http://www.rfc-editor.org/info/rfc4855>>.
- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<http://www.rfc-editor.org/info/rfc5124>>.
- [RFC6562] Perkins, C. and JM. Valin, "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP", RFC 6562, DOI 10.17487/RFC6562, March 2012, <<http://www.rfc-editor.org/info/rfc6562>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<http://www.rfc-editor.org/info/rfc6838>>.
- [RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", RFC 8083, DOI 10.17487/RFC8083, March 2017, <<http://www.rfc-editor.org/info/rfc8083>>.

- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", RFC 8085, DOI 10.17487/RFC8085, March 2017, <<http://www.rfc-editor.org/info/rfc8085>>.
- [NRLVDR] Heide, D., Cohen, A., Lee, Y., and T. Moran, "Universal Vocoder Using Variable Data Rate Vocoding", Naval Research Lab, NRL/FR/5555-13-10,239, June 2013.
- [MELP] Department of Defense Telecommunications Standard, "Analog-to-Digital Conversion of Voice by 2,400 Bit/Second Mixed Excitation Linear Prediction (MELP)", MIL-STD-3005, December 1999.
- [MELPE] North Atlantic Treaty Organization (NATO), "The 600 Bit/S, 1200 Bit/S and 2400 Bit/S NATO Interoperable Narrow Band Voice Coder", STANAG No. 4591, January 2006.
- [SCIP210] National Security Agency, "SCIP Signaling Plan", SCIP-210, December 2007.

## 10.2. Informative References

- [TSVCIS] National Security Agency, "Tactical Secure Voice Cryptographic Interoperability Specification (TSVCIS) Version 3.1", NSA 09-01A, March 2019.
- [RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <<http://www.rfc-editor.org/info/rfc4585>>.
- [RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, DOI 10.17487/RFC7201, April 2014, <<http://www.rfc-editor.org/info/rfc7201>>.
- [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, <<http://www.rfc-editor.org/info/rfc7202>>.
- [RMCAT] IETF, RTP Media Congestion Avoidance Techniques (rmcat) Working Group, <<https://datatracker.ietf.org/wg/rmcat/about/>>.

## Authors' Addresses

Victor Demjanenko, Ph.D.

VOCAL Technologies, Ltd.  
520 Lee Entrance, Suite 202  
Buffalo, NY 14228  
United States of America

Phone: +1 716 688 4675  
Email: victor.demjanenko@vocal.com

John Punaro  
VOCAL Technologies, Ltd.  
520 Lee Entrance, Suite 202  
Buffalo, NY 14228  
United States of America

Phone: +1 716 688 4675  
Email: john.punaro@vocal.com

David Satterlee  
VOCAL Technologies, Ltd.  
520 Lee Entrance, Suite 202  
Buffalo, NY 14228  
United States of America

Phone: +1 716 688 4675  
Email: david.satterlee@vocal.com

AVTCORE Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 8, 2021

J. Uberti  
S. Holmer  
M. Flodman  
D. Hong  
Google  
J. Lennox  
8x8 / Jitsi  
July 7, 2020

RTP Payload Format for VP9 Video  
draft-ietf-payload-vp9-10

Abstract

This memo describes an RTP payload format for the VP9 video codec. The payload format has wide applicability, as it supports applications from low bit-rate peer-to-peer usage, to high bit-rate video conferences. It includes provisions for temporal and spatial scalability.

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 <https://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 January 8, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Conventions, Definitions and Acronyms . . . . .	3
3. Media Format Description . . . . .	3
4. Payload Format . . . . .	5
4.1. RTP Header Usage . . . . .	5
4.2. VP9 Payload Descriptor . . . . .	6
4.2.1. Scalability Structure (SS): . . . . .	11
4.3. Frame Fragmentation . . . . .	13
4.4. Scalable encoding considerations . . . . .	13
4.5. Examples of VP9 RTP Stream . . . . .	14
4.5.1. Reference picture use for scalable structure . . . . .	14
5. Feedback Messages and Header Extensions . . . . .	14
5.1. Reference Picture Selection Indication (RPSI) . . . . .	15
5.2. Full Intra Request (FIR) . . . . .	15
5.3. Layer Refresh Request (LRR) . . . . .	15
5.4. Frame Marking . . . . .	16
6. Payload Format Parameters . . . . .	17
6.1. Media Type Definition . . . . .	17
6.2. SDP Parameters . . . . .	19
6.2.1. Mapping of Media Subtype Parameters to SDP . . . . .	19
6.2.2. Offer/Answer Considerations . . . . .	20
7. Security Considerations . . . . .	20
8. Congestion Control . . . . .	21
9. IANA Considerations . . . . .	21
10. Acknowledgments . . . . .	21
11. References . . . . .	21
11.1. Normative References . . . . .	21
11.2. Informative References . . . . .	23
Authors' Addresses . . . . .	23

## 1. Introduction

This memo describes an RTP payload specification applicable to the transmission of video streams encoded using the VP9 video codec [VP9-BITSTREAM]. The format described in this document can be used both in peer-to-peer and video conferencing applications.

The VP9 video codec was developed by Google, and is the successor to its earlier VP8 [RFC6386] codec. Above the compression improvements and other general enhancements above VP8, VP9 is also designed in a way that allows spatially-scalable video encoding.

## 2. Conventions, Definitions and Acronyms

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

## 3. Media Format Description

The VP9 codec can maintain up to eight reference frames, of which up to three can be referenced by any new frame.

VP9 also allows a frame to use another frame of a different resolution as a reference frame. (Specifically, a frame may use any references whose width and height are between 1/16th that of the current frame and twice that of the current frame, inclusive.) This allows internal resolution changes without requiring the use of key frames.

These features together enable an encoder to implement various forms of coarse-grained scalability, including temporal, spatial and quality scalability modes, as well as combinations of these, without the need for explicit scalable coding tools.

Temporal layers define different frame rates of video; spatial and quality layers define different and possibly dependent representations of a single input frame. Spatial layers allow a frame to be encoded at different resolutions, whereas quality layers allow a frame to be encoded at the same resolution but at different qualities (and thus with different amounts of coding error). VP9 supports quality layers as spatial layers without any resolution changes; hereinafter, the term "spatial layer" is used to represent both spatial and quality layers.

This payload format specification defines how such temporal and spatial scalability layers can be described and communicated.

Temporal and spatial scalability layers are associated with non-negative integer IDs. The lowest layer of either type has an ID of 0, and is sometimes referred to as the "base" temporal or spatial layer.

Layers are designed (and MUST be encoded) such that if any layer, and all higher layers, are removed from the bitstream along either of the two dimensions, the remaining bitstream is still correctly decodable.

For terminology, this document uses the term "frame" to refer to a single encoded VP9 frame for a particular resolution/quality, and "picture" to refer to all the representations (frames) at a single

instant in time. A picture thus consists of one or more frames, encoding different spatial layers.

Within a picture, a frame with spatial layer ID equal to SID, where  $SID > 0$ , can depend on a frame of the same picture with a lower spatial layer ID. This "inter-layer" dependency can result in additional coding gain compared to the case where only traditional "inter-picture" dependency is used, where a frame depends on previously coded frame in time. For simplicity, this payload format assumes that, within a picture and if inter-layer dependency is used, a spatial layer SID frame can depend only on the immediately previous spatial layer SID-1 frame, when  $S > 0$ . Additionally, if inter-picture dependency is used, a spatial layer SID frame is assumed to only depend on a previously coded spatial layer SID frame.

Given above simplifications for inter-layer and inter-picture dependencies, a flag (the D bit described below) is used to indicate whether a spatial layer SID frame depends on the spatial layer SID-1 frame. Given the D bit, a receiver only needs to additionally know the inter-picture dependency structure for a given spatial layer frame in order to determine its decodability. Two modes of describing the inter-picture dependency structure are possible: "flexible mode" and "non-flexible mode". An encoder can only switch between the two on the first packet of a key frame with temporal layer ID equal to 0.

In flexible mode, each packet can contain up to 3 reference indices, which identify all frames referenced by the frame transmitted in the current packet for inter-picture prediction. This (along with the D bit) enables a receiver to identify if a frame is decodable or not and helps it understand the temporal layer structure. Since this is signaled in each packet it makes it possible to have very flexible temporal layer hierarchies and patterns which are changing dynamically.

In non-flexible mode, the inter-picture dependency (the reference indices) of a Picture Group (PG) MUST be pre-specified as part of the scalability structure (SS) data. In this mode, each packet has an index to refer to one of the described pictures in the PG, from which the pictures referenced by the picture transmitted in the current packet for inter-picture prediction can be identified.

(Editor's Note: A "Picture Group", as used in this document, is not the same thing as the term "Group of Pictures" as it is traditionally used in video coding, i.e. to mean an independently-decodable run of pictures beginning with a keyframe. Suggestions for better terminology are welcome.)

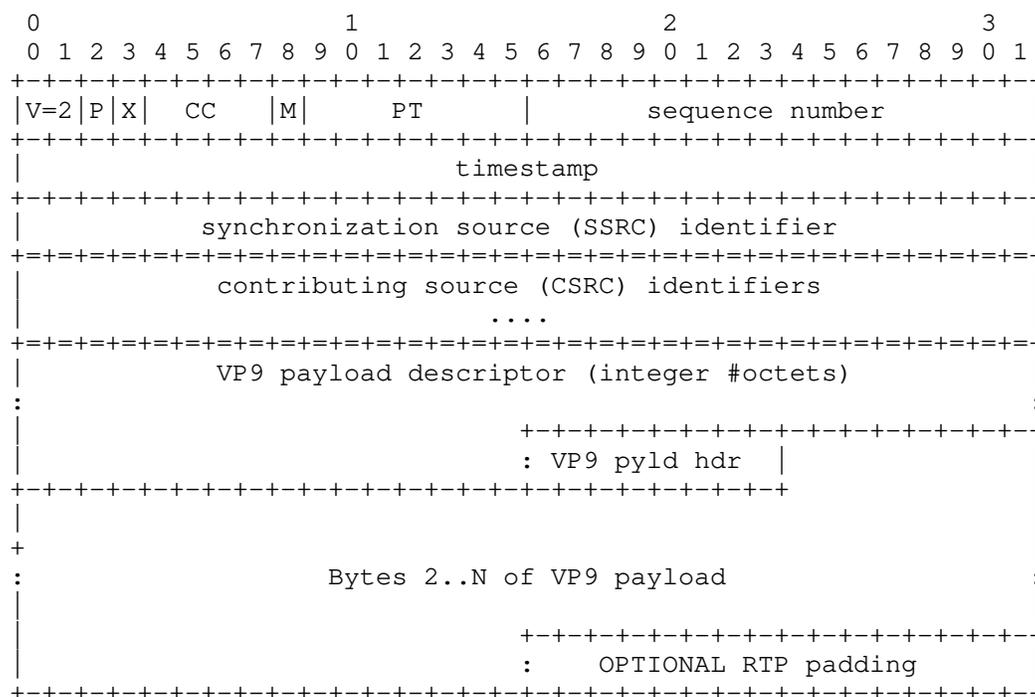
The SS data can also be used to specify the resolution of each spatial layer present in the VP9 stream for both flexible and non-flexible modes.

#### 4. Payload Format

This section describes how the encoded VP9 bitstream is encapsulated in RTP. To handle network losses usage of RTP/AVPF [RFC4585] is RECOMMENDED. All integer fields in the specifications are encoded as unsigned integers in network octet order.

##### 4.1. RTP Header Usage

The general RTP payload format for VP9 is depicted below.



The VP9 payload descriptor will be described in Section 4.2; the VP9 payload header is described in [VP9-BITSTREAM]. OPTIONAL RTP padding MUST NOT be included unless the P bit is set. The figure specifically shows the format for the first packet in a frame. Subsequent packets will not contain the VP9 payload header, and will have later octets in the frame payload.

Figure 1

Marker bit (M): MUST be set to 1 for the final packet of the highest spatial layer frame (the final packet of the picture), and 0 otherwise. Unless spatial scalability is in use for this picture, this will have the same value as the E bit described below. Note this bit MUST be set to 1 for the target spatial layer frame if a stream is being rewritten to remove higher spatial layers.

Payload Type (PT): In line with the policy in Section 3 of [RFC3551], applications using the VP9 RTP payload profile MUST assign a dynamic payload type number to be used in each RTP session and provide a mechanism to indicate the mapping. See Section 6.2 for the mechanism to be used with the Session Description Protocol (SDP) [RFC4566].

Timestamp: The RTP timestamp indicates the time when the input frame was sampled, at a clock rate of 90 kHz. If the input picture is encoded with multiple layer frames, all of the frames of the picture MUST have the same timestamp.

If a frame has the VP9 `show_frame` field set to 0 (i.e., it is meant only to populate a reference buffer, without being output) its timestamp MAY alternately be set to be the same as the subsequent frame with `show_frame` equal to 1. (This will be convenient for playing out pre-encoded content packaged with VP9 "superframes", which typically bundle `show_frame==0` frames with a subsequent `show_frame==1` frame.) Every frame with `show_frame==1`, however, MUST have a unique timestamp modulo the  $2^{32}$  wrap of the field.

The remaining RTP Fixed Header Fields (V, P, X, CC, sequence number, SSRC and CSRC identifiers) are used as specified in Section 5.1 of [RFC3550].

#### 4.2. VP9 Payload Descriptor

In flexible mode (with the F bit below set to 1), The first octets after the RTP header are the VP9 payload descriptor, with the following structure.

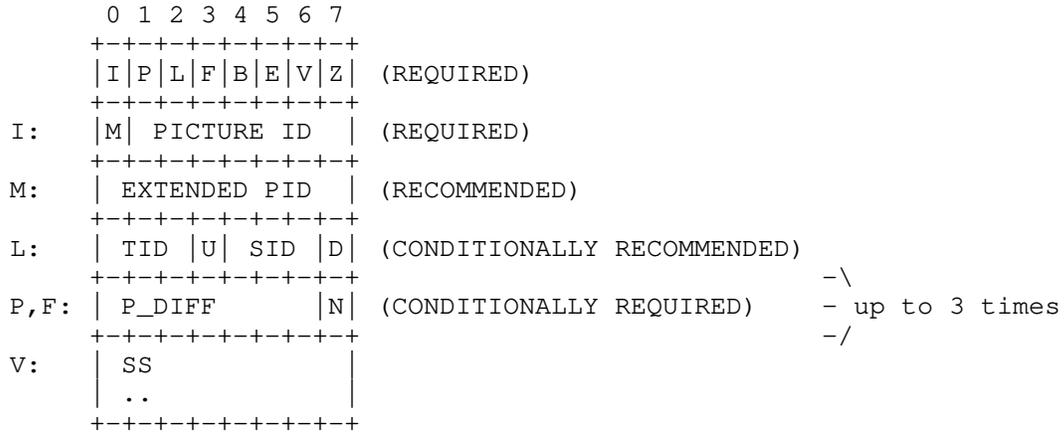


Figure 2

In non-flexible mode (with the F bit below set to 0), The first octets after the RTP header are the VP9 payload descriptor, with the following structure.

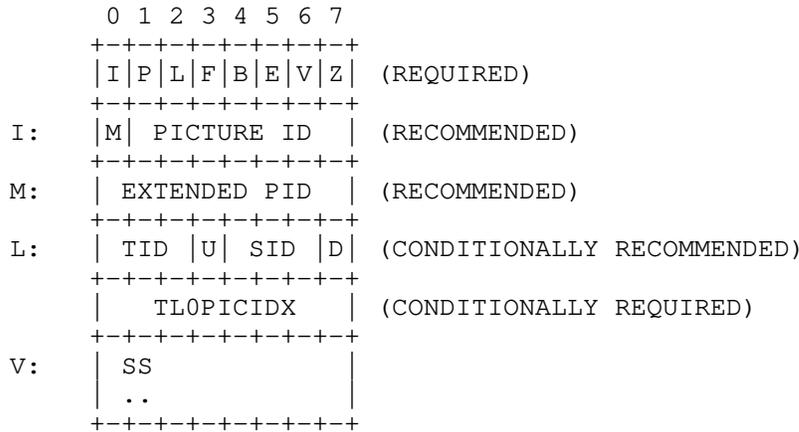


Figure 3

I: Picture ID (PID) present. When set to one, the OPTIONAL PID MUST be present after the mandatory first octet and specified as below.

Otherwise, PID MUST NOT be present. If the SS field was present in the stream's most recent start of a keyframe (i.e., non-flexible scalability mode is in use), then the PID MUST also be present in every packet.

- P: Inter-picture predicted frame. When set to zero, the frame does not utilize inter-picture prediction. In this case, up-switching to a current spatial layer's frame is possible from directly lower spatial layer frame. P SHOULD also be set to zero when encoding a layer synchronization frame in response to an LRR [I-D.ietf-avtext-lrr] message (see Section 5.3). When P is set to zero, the TID field (described below) MUST also be set to 0 (if present). Note that the P bit does not forbid intra-picture, inter-layer prediction from earlier frames of the same picture, if any.
- L: Layer indices present. When set to one, the one or two octets following the mandatory first octet and the PID (if present) is as described by "Layer indices" below. If the F bit (described below) is set to 1 (indicating flexible mode), then only one octet is present for the layer indices. Otherwise if the F bit is set to 0 (indicating non-flexible mode), then two octets are present for the layer indices.
- F: Flexible mode. F set to one indicates flexible mode and if the P bit is also set to one, then the octets following the mandatory first octet, the PID, and layer indices (if present) are as described by "Reference indices" below. This MUST only be set to 1 if the I bit is also set to one; if the I bit is set to zero, then this MUST also be set to zero and ignored by receivers. The value of this F bit MUST only change on the first packet of a key picture. A key picture is a picture whose base spatial layer frame is a key frame, and which thus completely resets the encoder state. This packet will have its P bit equal to zero, SID or D bit (described below) equal to zero, and B bit (described below) equal to 1.
- B: Start of a frame. MUST be set to 1 if the first payload octet of the RTP packet is the beginning of a new VP9 frame, and MUST NOT be 1 otherwise. Note that this frame might not be the first frame of a picture.
- E: End of a frame. MUST be set to 1 for the final RTP packet of a VP9 frame, and 0 otherwise. This enables a decoder to finish decoding the frame, where it otherwise may need to wait for the next packet to explicitly know that the frame is complete. Note that, if spatial scalability is in use, more frames from the same picture may follow; see the description of the M bit above.

- V: Scalability structure (SS) data present. When set to one, the OPTIONAL SS data MUST be present in the payload descriptor. Otherwise, the SS data MUST NOT be present.
- Z: Not a reference frame for upper spatial layers. If set to 1, indicates that frames with higher spatial layers SID+1 of the current and following pictures do not depend on the current spatial layer SID frame. This enables a decoder which is targeting a higher spatial layer to know that it can safely discard this packet's frame without processing it, without having to wait for the "D" bit in the higher-layer frame (see below).

The mandatory first octet is followed by the extension data fields that are enabled:

- M: The most significant bit of the first octet is an extension flag. The field MUST be present if the I bit is equal to one. If set, the PID field MUST contain 15 bits; otherwise, it MUST contain 7 bits. See PID below.

Picture ID (PID): Picture ID represented in 7 or 15 bits, depending on the M bit. This is a running index of the pictures. The field MUST be present if the I bit is equal to one. If M is set to zero, 7 bits carry the PID; else if M is set to one, 15 bits carry the PID in network byte order. The sender may choose between a 7- or 15-bit index. The PID SHOULD start on a random number, and MUST wrap after reaching the maximum ID. The receiver MUST NOT assume that the number of bits in PID stay the same through the session.

In the non-flexible mode (when the F bit is set to 0), this PID is used as an index to the picture group (PG) specified in the SS data below. In this mode, the PID of the key frame corresponds to the first specified frame in the PG. Then subsequent PIDs are mapped to subsequently specified frames in the PG (modulo N\_G, specified in the SS data below), respectively.

All frames of the same picture MUST have the same PID value.

Frames (and their corresponding pictures) with the VP9 show\_frame field equal to 0 MUST have distinct PID values from subsequent pictures with show\_frame equal to 1. Thus, a Picture as defined in this specification is different than a VP9 Superframe.

All frames of the same picture MUST have the same value for show\_frame.

Layer indices: This information is optional but recommended whenever encoding with layers. For both flexible and non-flexible modes, one octet is used to specify a layer frame's temporal layer ID (TID) and spatial layer ID (SID) as shown both in Figure 2 and Figure 3. Additionally, a bit (U) is used to indicate that the current frame is a "switching up point" frame. Another bit (D) is used to indicate whether inter-layer prediction is used for the current frame.

In the non-flexible mode (when the F bit is set to 0), another octet is used to represent temporal layer 0 index (TLOPICIDX), as depicted in Figure 3. The TLOPICIDX is present so that all minimally required frames - the base temporal layer frames - can be tracked.

The TID and SID fields indicate the temporal and spatial layers and can help middleboxes and endpoints quickly identify which layer a packet belongs to.

TID: The temporal layer ID of current frame. In the case of non-flexible mode, if PID is mapped to a picture in a specified PG, then the value of TID MUST match the corresponding TID value of the mapped picture in the PG.

U: Switching up point. If this bit is set to 1 for the current picture with temporal layer ID equal to TID, then "switch up" to a higher frame rate is possible as subsequent higher temporal layer pictures will not depend on any picture before the current picture (in coding order) with temporal layer ID greater than TID.

SID: The spatial layer ID of current frame. Note that frames with spatial layer SDI > 0 may be dependent on decoded spatial layer SID-1 frame within the same picture. Different frames of the same picture MUST have distinct spatial layer IDs, and frames' spatial layers MUST appear in increasing order within the frame.

D: Inter-layer dependency used. MUST be set to one if current spatial layer SID frame depends on spatial layer SID-1 frame of the same picture. MUST only be set to zero if current spatial layer SID frame does not depend on spatial layer SID-1 frame of the same picture. For the base layer frame (with SID equal to 0), this D bit MUST be set to zero.

TLOPICIDX: 8 bits temporal layer zero index. TLOPICIDX is only present in the non-flexible mode (F = 0). This is a running index for the temporal base layer pictures, i.e., the pictures

with TID set to 0. If TID is larger than 0, TLOPICIDX indicates which temporal base layer picture the current picture depends on. TLOPICIDX MUST be incremented when TID is equal to 0. The index SHOULD start on a random number, and MUST restart at 0 after reaching the maximum number 255.

Reference indices: When P and F are both set to one, indicating a non-key frame in flexible mode, then at least one reference index has to be specified as below. Additional reference indices (total of up to 3 reference indices are allowed) may be specified using the N bit below. When either P or F is set to zero, then no reference index is specified.

P\_DIFF: The reference index (in 7 bits) specified as the relative PID from the current picture. For example, when P\_DIFF=3 on a packet containing the picture with PID 112 means that the picture refers back to the picture with PID 109. This calculation is done modulo the size of the PID field, i.e., either 7 or 15 bits.

N: 1 if there is additional P\_DIFF following the current P\_DIFF.

#### 4.2.1. Scalability Structure (SS):

The scalability structure (SS) data describes the resolution of each frame within a picture as well as the inter-picture dependencies for a picture group (PG). If the VP9 payload descriptor's "V" bit is set, the SS data is present in the position indicated in Figure 2 and Figure 3.

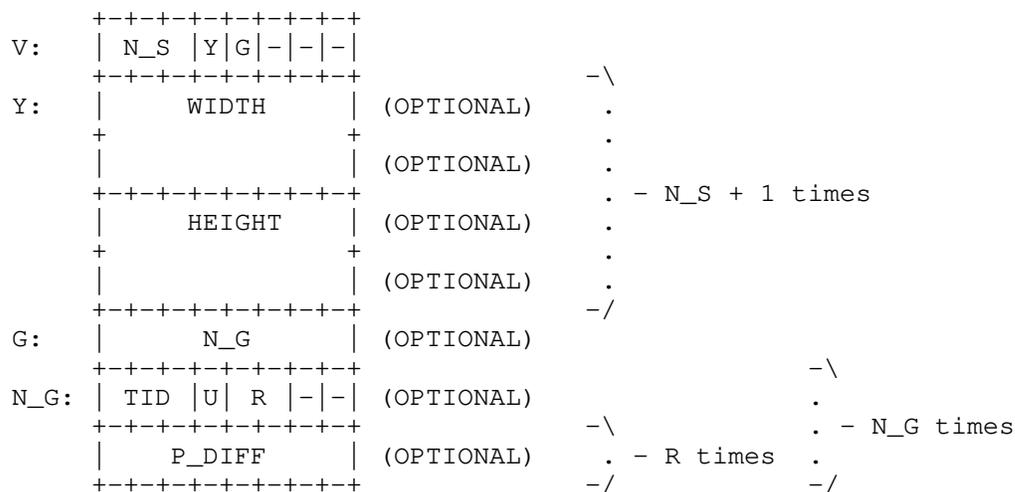


Figure 4

N\_S: N\_S + 1 indicates the number of spatial layers present in the VP9 stream.

Y: Each spatial layer's frame resolution present. When set to one, the OPTIONAL WIDTH (2 octets) and HEIGHT (2 octets) MUST be present for each layer frame. Otherwise, the resolution MUST NOT be present.

G: PG description present flag.

-: Bit reserved for future use. MUST be set to zero and MUST be ignored by the receiver.

N\_G: N\_G indicates the number of pictures in a Picture Group (PG). If N\_G is greater than 0, then the SS data allows the inter-picture dependency structure of the VP9 stream to be pre-declared, rather than indicating it on the fly with every packet. If N\_G is greater than 0, then for N\_G pictures in the PG, each picture's temporal layer ID (TID), switch up point (U), and the R reference indices (P\_DIFFs) are specified.

The first picture specified in the PG MUST have TID set to 0.

G set to 0 or N\_G set to 0 indicates that either there is only one temporal layer or no fixed inter-picture dependency information is present going forward in the bitstream.

Note that for a given picture, all frames follow the same inter-picture dependency structure. However, the frame rate of each spatial layer can be different from each other and this can be controlled with the use of the D bit described above. The specified dependency structure in the SS data MUST be for the highest frame rate layer.

In a scalable stream sent with a fixed pattern, the SS data SHOULD be included in the first packet of every key frame. This is a packet with P bit equal to zero, SID or D bit equal to zero, and B bit equal to 1. The SS data MUST only be changed on the picture that corresponds to the first picture specified in the previous SS data's PG (if the previous SS data's N\_G was greater than 0).

#### 4.3. Frame Fragmentation

VP9 frames are fragmented into packets, in RTP sequence number order, beginning with a packet with the B bit set, and ending with a packet with the E bit set. There is no mechanism for finer-grained access to parts of a VP9 frame.

#### 4.4. Scalable encoding considerations

In addition to the use of reference frames, VP9 has several additional forms of inter-frame dependencies, largely involving probability tables for the entropy and tree encoders. In VP9 syntax, the syntax element "error\_resilient\_mode" resets this additional inter-frame data, allowing a frame's syntax to be decoded independently.

Due to the requirements of scalable streams, a VP9 encoder producing a scalable stream needs to ensure that a frame does not depend on a previous frame (of the same or a previous picture) that can legitimately be removed from the stream. Thus, a frame that follows a removable frame (in full decode order) MUST be encoded with "error\_resilient\_mode" set to true.

For spatially-scalable streams, this means that "error\_resilient\_mode" needs to be turned on for the base spatial layer; it can however be turned off for higher spatial layers, assuming they are sent with inter-layer dependency (i.e. with the "D" bit set). For streams that are only temporally-scalable without spatial scalability, "error\_resilient\_mode" can additionally be turned off for any picture that immediately follows a temporal layer 0 frame.

4.5. Examples of VP9 RTP Stream

4.5.1. Reference picture use for scalable structure

As discussed in Section 3, the VP9 codec can maintain up to eight reference frames, of which up to three can be referenced or updated by any new frame. This section illustrates one way that a scalable structure (with three spatial layers and three temporal layers) can be constructed using these reference frames.

Temporal	Spatial	References	Updates
0	0	0	0
0	1	0,1	1
0	2	1,2	2
2	0	0	6
2	1	1,6	7
2	2	2,7	-
1	0	0	3
1	1	1,3	4
1	2	2,4	5
2	0	3	6
2	1	4,6	7
2	2	5,7	-

Example scalability structure

This structure is constructed such that the "U" bit can always be set.

5. Feedback Messages and Header Extensions

5.1. Reference Picture Selection Indication (RPSI)

The reference picture selection index is a payload-specific feedback message defined within the RTCP-based feedback format. The RPSI message is generated by a receiver and can be used in two ways. Either it can signal a preferred reference picture when a loss has been detected by the decoder -- preferably then a reference that the decoder knows is perfect -- or, it can be used as positive feedback information to acknowledge correct decoding of certain reference pictures. The positive feedback method is useful for VP9 used for point to point (unicast) communication. The use of RPSI for VP9 is preferably combined with a special update pattern of the codec's two special reference frames -- the golden frame and the altref frame -- in which they are updated in an alternating leapfrog fashion. When a receiver has received and correctly decoded a golden or altref frame, and that frame had a PictureID in the payload descriptor, the receiver can acknowledge this simply by sending an RPSI message back to the sender. The message body (i.e., the "native RPSI bit string" in [RFC4585]) is simply the PictureID of the received frame.

Note: because all frames of the same picture must have the same inter-picture reference structure, there is no need for a message to specify which frame is being selected.

5.2. Full Intra Request (FIR)

The Full Intra Request (FIR) [RFC5104] RTCP feedback message allows a receiver to request a full state refresh of an encoded stream.

Upon receipt of an FIR request, a VP9 sender MUST send a picture with a keyframe for its spatial layer 0 layer frame, and then send frames without inter-picture prediction (P=0) for any higher layer frames.

5.3. Layer Refresh Request (LRR)

The Layer Refresh Request [I-D.ietf-avtext-lrr] allows a receiver to request a single layer of a spatially or temporally encoded stream to be refreshed, without necessarily affecting the stream's other layers.

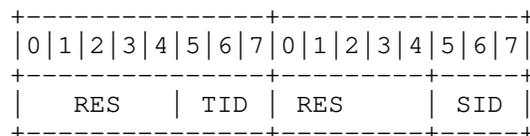


Figure 5

Figure 5 shows the format of LRR's layer index fields for VP9 streams. The two "RES" fields MUST be set to 0 on transmission and ignored on reception. See Section 4.2 for details on the TID and SID fields.

Identification of a layer refresh frame can be derived from the reference IDs of each frame by backtracking the dependency chain until reaching a point where only decodable frames are being referenced. Therefore it's recommended for both the flexible and the non-flexible mode that, when upgrade frames are being encoded in response to a LRR, those packets should contain layer indices and the reference fields so that the decoder or an MCU can make this derivation.

Example:

LRR {1,0}, {2,1} is sent by an MCU when it is currently relaying {1,0} to a receiver and which wants to upgrade to {2,1}. In response the encoder should encode the next frames in layers {1,1} and {2,1} by only referring to frames in {1,0}, or {0,0}.

In the non-flexible mode, periodic upgrade frames can be defined by the layer structure of the SS, thus periodic upgrade frames can be automatically identified by the picture ID.

5.4. Frame Marking

The Frame Marking RTP header extension [I-D.ietf-avtext-framemarking] is a mechanism to provide information about frames of video streams in a largely codec-independent manner. However, for its extension for scalable codecs, the specific manner in which codec layers are identified needs to be specified specifically for each codec. This section defines how frame marking is used with VP9.

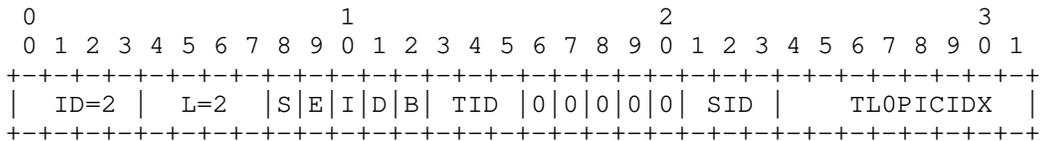


Figure 6

When this header extension is used with VP9, the TID and SID fields MUST match the values in the packet which the header extension is attached to; see Section 4.2 for details on these fields.

See [I-D.ietf-avtext-framemarking] for explanations of the other fields, which are generic.

## 6. Payload Format Parameters

This payload format has two optional parameters.

### 6.1. Media Type Definition

This registration is done using the template defined in [RFC6838] and following [RFC4855].

Type name: video

Subtype name: VP9

Required parameters: None.

Optional parameters:

These parameters are used to signal the capabilities of a receiver implementation. If the implementation is willing to receive media, both parameters MUST be provided. These parameters MUST NOT be used for any other purpose.

**max-fr:** The value of max-fr is an integer indicating the maximum frame rate in units of frames per second that the decoder is capable of decoding.

**max-fs:** The value of max-fs is an integer indicating the maximum frame size in units of macroblocks that the decoder is capable of decoding.

The decoder is capable of decoding this frame size as long as the width and height of the frame in macroblocks are less than  $\text{int}(\sqrt{\text{max-fs} * 8})$  - for instance, a max-fs of 1200 (capable of supporting 640x480 resolution) will support widths and heights up to 1552 pixels (97 macroblocks).

**profile-id:** The value of profile-id is an integer indicating the default coding profile, the subset of coding tools that may have been used to generate the stream or that the receiver supports). Table 1 lists all of the profiles defined in section 7.2 of [VP9-BITSTREAM] and the corresponding integer values to be used.

If no profile-id is present, Profile 0 MUST be inferred.

Informative note: See Table 2 for capabilities of coding profiles defined in section 7.2 of [VP9-BITSTREAM].

Encoding considerations:

This media type is framed in RTP and contains binary data; see Section 4.8 of [RFC6838].

Security considerations: See Section 7 of RFC xxxx.

[RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Interoperability considerations: None.

Published specification: VP9 bitstream format [VP9-BITSTREAM] and RFC XXXX.

[RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Applications which use this media type:

For example: Video over IP, video conferencing.

Fragment identifier considerations: N/A.

Additional information: None.

Person & email address to contact for further information:

Jonathan Lennox <jonathan.lennox@8x8.com>

Intended usage: COMMON

Restrictions on usage:

This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550].

Author: Jonathan Lennox <jonathan.lennox@8x8.com>

Change controller:

IETF AVTCORE Working Group delegated from the IESG.

Profile	profile-id
0	0
1	1
2	2
3	3

Table 1: Table 1. Table of profile-id integer values representing the VP9 profile corresponding to the set of coding tools supported.

Profile	Bit Depth	SRGB Colorspace	Chroma Subsampling
0	8	No	YUV 4:2:0
1	8	Yes	YUV 4:2:0, 4:4:0 or 4:4:4
2	10 or 12	No	YUV 4:2:0
3	10 or 12	Yes	YUV 4:2:0, 4:4:0 or 4:4:4

Table 2: Table 2. Table of profile capabilities.

## 6.2. SDP Parameters

The receiver MUST ignore any fmtp parameter unspecified in this memo.

### 6.2.1. Mapping of Media Subtype Parameters to SDP

The media type video/VP9 string is mapped to fields in the Session Description Protocol (SDP) [RFC4566] as follows:

- o The media name in the "m=" line of SDP MUST be video.
- o The encoding name in the "a=rtpmap" line of SDP MUST be VP9 (the media subtype).
- o The clock rate in the "a=rtpmap" line MUST be 90000.
- o The parameters "max-fs", and "max-fr", MUST be included in the "a=fmtp" line of SDP if SDP is used to declare receiver capabilities. These parameters are expressed as a media subtype

string, in the form of a semicolon separated list of parameter=value pairs.

- o The OPTIONAL parameter profile-id, when present, SHOULD be included in the "a=fmtp" line of SDP. This parameter is expressed as a media subtype string, in the form of a parameter=value pair. When the parameter is not present, a value of 0 MUST be used for profile-id.

#### 6.2.1.1. Example

An example of media representation in SDP is as follows:

```
m=video 49170 RTP/AVPF 98
a=rtpmap:98 VP9/90000
a=fmtp:98 max-fr=30; max-fs=3600; profile-id=0;
```

#### 6.2.2. Offer/Answer Considerations

When VP9 is offered over RTP using SDP in an Offer/Answer model [RFC3264] for negotiation for unicast usage, the following limitations and rules apply:

- o The parameter identifying a media format configuration for VP9 is profile-id. This media format configuration parameter MUST be used symmetrically; that is, the answerer MUST either maintain all configuration parameters or remove the media format (payload type) completely if one or more of the parameter values are not supported.
- o To simplify the handling and matching of these configurations, the same RTP payload type number used in the offer SHOULD also be used in the answer, as specified in [RFC3264]. An answer MUST NOT contain the payload type number used in the offer unless the configuration is exactly the same as in the offer.

### 7. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. SAVPF [RFC5124]. However, as "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format's responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity and source authenticity for RTP in general. This responsibility lays on anyone

using RTP in an application. They can find guidance on available security mechanisms in Options for Securing RTP Sessions [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this security consideration section discusses the security impacting properties of the payload format itself.

This RTP payload format and its media decoder do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing, and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Nor does the RTP payload format contain any active content.

## 8. Congestion Control

Congestion control for RTP SHALL be used in accordance with RFC 3550 [RFC3550], and with any applicable RTP profile; e.g., RFC 3551 [RFC3551]. The congestion control mechanism can, in a real-time encoding scenario, adapt the transmission rate by instructing the encoder to encode at a certain target rate. Media aware network elements MAY use the information in the VP9 payload descriptor in Section 4.2 to identify non-reference frames and discard them in order to reduce network congestion. Note that discarding of non-reference frames cannot be done if the stream is encrypted (because the non-reference marker is encrypted).

## 9. IANA Considerations

The IANA is requested to register the media type registration "video/vp9" as specified in Section 6.1. The media type is also requested to be added to the IANA registry for "RTP Payload Format MIME types" <<http://www.iana.org/assignments/rtp-parameters>>.

## 10. Acknowledgments

Alex Eleftheriadis, Yuki Ito, Won Kap Jang, Sergio Garcia Murillo, Roi Sasson, Timothy Terriberry, Emircan Uysaler, and Thomas Volkert commented on the development of this document and provided helpful comments and feedback.

## 11. References

### 11.1. Normative References

- [I-D.ietf-avtext-framemarking]  
Zanaty, M., Berger, E., and S. Nandakumar, "Frame Marking RTP Header Extension", draft-ietf-avtext-framemarking-10 (work in progress), November 2019.

- [I-D.ietf-avtext-lrr]  
Lennox, J., Hong, D., Uberti, J., Holmer, S., and M. Flodman, "The Layer Refresh Request (LRR) RTCP Feedback Message", draft-ietf-avtext-lrr-07 (work in progress), July 2017.
- [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>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/info/rfc3264>>.
- [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>>.
- [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>>.
- [RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <<https://www.rfc-editor.org/info/rfc4855>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/info/rfc6838>>.

## [VP9-BITSTREAM]

Grange, A., de Rivaz, P., and J. Hunt, "VP9 Bitstream & Decoding Process Specification", Version 0.6, March 2016, <<https://storage.googleapis.com/downloads.webmproject.org/docs/vp9/vp9-bitstream-specification-v0.6-20160331-draft.pdf>>.

## 11.2. Informative References

- [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>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/info/rfc5124>>.
- [RFC6386] Bankoski, J., Koleszar, J., Quillio, L., Salonen, J., Wilkins, P., and Y. Xu, "VP8 Data Format and Decoding Guide", RFC 6386, DOI 10.17487/RFC6386, November 2011, <<https://www.rfc-editor.org/info/rfc6386>>.
- [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>>.

## Authors' Addresses

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

Email: [justin@uberti.name](mailto:justin@uberti.name)

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

Email: holmer@google.com

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

Email: mflodman@google.com

Danny Hong  
Google, Inc.  
1585 Charleston Road  
Mountain View, CA 94043  
US

Email: dannyhong@google.com

Jonathan Lennox  
8x8, Inc. / Jitsi  
1350 Broadway  
New York, NY 10018  
US

Email: jonathan.lennox@8x8.com

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 14, 2021

S. Zhao  
S. Wenger  
Tencent  
July 13, 2020

RTP Payload Format for Essential Video Coding (EVC)  
draft-zhao-avtcore-rtp-enc-02

## Abstract

This memo describes an RTP payload format for the video coding standard ISO/IEC International Standard 23094-1 [ISO23094-1], also known as Essential Video Coding [EVC] and developed by ISO/IEC JTC1/SC29/WG11. The RTP payload format allows for packetization of one or more Network Abstraction Layer (NAL) units in each RTP packet payload as well as fragmentation of a NAL unit into multiple RTP packets. The payload format has wide applicability in videoconferencing, Internet video streaming, and high-bitrate entertainment-quality video, among other applications.

## 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 <https://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 January 14, 2021.

## Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1.	Introduction . . . . .	3
1.1.	Overview of the EVC Codec . . . . .	3
1.1.1.	Coding-Tool Features (informative) . . . . .	4
1.1.2.	Systems and Transport Interfaces . . . . .	6
1.1.3.	Parallel Processing Support (informative) . . . . .	8
1.1.4.	NAL Unit Header . . . . .	8
1.2.	Overview of the Payload Format . . . . .	9
2.	Conventions . . . . .	10
3.	Definitions and Abbreviations . . . . .	10
3.1.	Definitions . . . . .	10
3.1.1.	Definitions from the EVC Specification . . . . .	10
3.1.2.	Definitions Specific to This Memo . . . . .	12
3.2.	Abbreviations . . . . .	13
4.	RTP Payload Format . . . . .	15
4.1.	RTP Header Usage . . . . .	15
4.2.	Payload Header Usage . . . . .	17
4.3.	Payload Structures . . . . .	17
4.3.1.	Single NAL Unit Packets . . . . .	17
4.3.2.	Aggregation Packets (APs) . . . . .	18
4.3.3.	Fragmentation Units . . . . .	22
4.4.	Decoding Order Number . . . . .	25
5.	Packetization Rules . . . . .	26
6.	De-packetization Process . . . . .	27
7.	Payload Format Parameters . . . . .	29
8.	Use with Feedback Messages . . . . .	29
8.1.	Picture Loss Indication (PLI) . . . . .	29
8.2.	Slice Loss Indication (SLI) . . . . .	29
8.3.	Reference Picture Selection Indication (RPSI) . . . . .	29
8.4.	Full Intra Request (FIR) . . . . .	29
9.	Frame marking . . . . .	29
10.	Security Considerations . . . . .	29
11.	Congestion Control . . . . .	30
12.	IANA Considerations . . . . .	31
13.	Acknowledgements . . . . .	31
14.	References . . . . .	32
14.1.	Normative References . . . . .	32
14.2.	Informative References . . . . .	33
	Authors' Addresses . . . . .	34

## 1. Introduction

The [EVC] specification, which will be formally designated (once approved) as ISO/IEC International Standard 23094-1 [ISO23094-1], is planned for ratification in early 2020. A draft that's currently in the approval process of ISO/IEC can be found as [EVC]. One goal of MPEG is to keep [EVC]'s baseline essentially royalty free by agreement among the key contributors, whereas more advanced profiles follow a reasonable and non-discriminatory licensing terms policy. Both baseline and higher profiles of [EVC] are reported to provide coding efficiency gains over [HEVC] and [AVC] under certain configurations.

Editor's note: Is it necessary to add comparison with [VVC]?

This memo describes an RTP payload format for [EVC]. It shares its basic design with the NAL unit-based RTP payload formats of H.264 Video Coding [RFC6184], Scalable Video Coding (SVC) [RFC6190], High Efficiency Video Coding (HEVC) [RFC7798], and Versatile Video Coding (VVC) [draft-ietf-avtcore-rtp-vvc]. With respect to design philosophy, security, congestion control, and overall implementation complexity, it has similar properties to those earlier payload format specifications. This is a conscious choice, as at least RFC 6184 is widely deployed and generally known in the relevant implementer communities. Certain mechanisms known from [RFC6190] were incorporated as EVC supports temporal scalability. [EVC] does not offer higher forms of scalability.

### 1.1. Overview of the EVC Codec

[EVC], [AVC], [HEVC] and [VVC] share a similar hybrid video codec design. In this memo, we provide a very brief overview of those features of EVC that are, in some form, addressed by the payload format specified herein. Implementers have to read, understand, and apply the ISO/IEC specifications pertaining to EVC to arrive at interoperable, well-performing implementations. The EVC standard has a baseline profile and on top of that, a main profile, the latter including more advanced features. A "toolset" syntax element allows encoders to mark a bitstream as to what of the many independent coding tools are exercised in the bitstream, in a spirit similar to the `general_constraint_flags` of [VVC].

Conceptually, all [EVC], [AVC], [HEVC] and [VVC] include a Video Coding Layer (VCL), which is often used to refer to the coding-tool features, and a Network Abstraction Layer (NAL), which is often used to refer to the systems and transport interface aspects of the codecs.

### 1.1.1.1. Coding-Tool Features (informative)

#### Coding blocks and transform structure

[EVC] uses a traditional quad-tree coding structure, which divides the encoded image into blocks of up to 128x128 luma samples, which can be recursively divided into smaller blocks. The main profile adds two advanced coding structure tools: Binary Ternary Tree (BTT) that allows non-square coding units and segmentation that changes the processing order of the segmentation unit from traditional left-scanning order processing to right-scanning order processing Unit Coding Order (SUCCO). In the main profile, the picture can be divided into rectangular tiles, and these tiles can be independently encoded and/or decoded in parallel.

When predicting a data block using intra prediction or inter prediction, the remaining data is usually added to the prediction block. The residual data is added to the prediction block. The residual data is obtained by applying an inverse quantization process and an inverse transform. [EVC] includes integer discrete cosine transform (DCT2) and scalar quantization. For the main profile, Improved Quantization and Transform (IQT) uses a different mapping/clipping function for quantization. An inverse zig-zag scanning order is used for coefficient coding. Advanced Coefficient Coding (ADCC) in the main profile can code coefficient values more efficiently, for example, indicated by the last non-zero coefficient. In main profile, Adaptive Transformation Selection (ATS) is also available and can be applied to integer versions of DST7 or DCT8, and not just DCT2.

#### Entropy coding

[EVC] uses a similar binary arithmetic coding mechanism as [AVC]. The mechanism includes a binarization step and a probability update defined by a lookup table. In the main profile, the derivation process of syntax elements based on adjacent blocks makes the context modeling and initialization process more efficient.

#### In-loop filtering

The baseline profile of [EVC] uses the deblocking filter defined in H.263 Annex J. In the main profile, compared to the deblocking filter in the baseline profile, an Advanced Deblocking Filter (ADDB) can be used, which can further reduce artifacts. The main profile also defines two additional in-loop filters that can be used to improve the quality of decoded pictures before output and/or for inter prediction. A Walsh-Hadamard Transform Domain Filter (HTDF) is applied to the luma samples before deblocking, and the scanning

process is used to determine 4 adjacent samples for filtering. An adaptive Loop Filter (ALF) allows to send signals of up to 25 different filters for the luma components, and the best filter can be selected through the classification process for each 4x4 block. The filter parameters of the ALF filter are signaled in the Adaptation Parameter Set (APS).

#### Inter-prediction

The basis of [EVC] inter prediction is motion compensation using interpolation filters with a quarter sample resolution. In baseline profile, a motion vector signal is transmitted using one of three spatially neighboring motion vectors and a temporally collocated motion vector as a predictor. The motion vector difference may be signaled relative to the selected predictor, but for the case where no motion vector difference is signaled and there is no remaining data in the block, there is a specific mode called a skip mode. The main profile includes six additional tools to provide improved inter prediction. With advanced Motion Interpolation and Signaling (AMIS), adjacent blocks can be conceptually merged to indicate that they use the same motion, but more advanced schemes can also be used to create predictions from the basic model list of candidate predictors. The Merge with Motion Vector Difference (MMVD) tool uses a process similar to the concept of merging neighboring blocks, but also allows the use of expressions that include a starting point, motion amplitude, and direction of motion to send a motion vector signal.

Using Advanced Motion Vector Prediction (AMVP), candidate motion vector predictions for the block can be derived from its neighboring blocks in the same picture and collocated blocks in the reference picture. The Adaptive Motion Vector Resolution (AMVR) tool provides a way to reduce the accuracy of a motion vector from a quarter sample to half sample, full sample, double sample, or quad sample, which provides the efficiency advantage, such as when sending large motion vector differences. The main profile also includes the Decoder-side Motion Vector Refinement (DMVR), which uses a bilateral template matching process to refine the motion vectors in a bidirectional fashion.

#### Intra prediction and intra-coding

Intra prediction in [EVC] is performed on adjacent samples of coding units in a partitioned structure. For the baseline profile, all coding units are square, and there are five different prediction modes: DC (mean value of the neighborhood), horizontal, vertical, and two different diagonal directions. In the main profile, intra prediction can be applied to any rectangular coding unit, and there are 28 additional direction modes available in the so-called Enhanced

Intra Prediction Directions (EIPD). In the main profile, an encoder can also use Intra Block Copy (IBC), where a previously decoded sample blocks of the same picture is used as a predictor. A displacement vector in integer sample precision is signaled to indicate where the prediction block in the current picture is used for this mode.

#### Decoded picture buffer management

In the previous technology, decoded pictures can be stored in a decoded picture buffer (Decoded Picture Buffer, DPB) for predicting pictures that follow them in decoding order. In the baseline profile, the management of the DPB (i.e. the process of adding and deleting reference pictures) is controlled by the information in the SPS. For the main profile, if a Reference Picture List (RPL) scheme is used, DPB management can be controlled by information that is signaled at the picture level.

#### 1.1.2. Systems and Transport Interfaces

[EVC] inherited the basic systems and transport interfaces designs from [AVC] and [HEVC]. These include the NAL-unit-based syntax structure, the hierarchical syntax and data unit structure and the Supplemental Enhancement Information (SEI) message mechanism. The hierarchical syntax and data unit structure consists of a sequence-level parameter set (SPS), two picture-level parameter sets (PPS and APS, each of which can apply to one or more pictures), slice-level header parameters, and lower-level parameters.

A number of key components that influenced the Network Abstraction Layer design of [EVC] as well as this memo are described below

##### Sequence parameter set

The Sequence Parameter Set (SPS) contains syntax elements pertaining to a coded video sequence (CVS), which is a group of pictures, starting with a random access point, and followed by pictures that may depend on each other and the random access point picture. In MPEG-2, the equivalent of a CVS was a Group of Pictures (GOP), which normally started with an I frame and was followed by P and B frames. While more complex in its options of random access points, EVC retains this basic concept. In many TV-like applications, a CVS contains a few hundred milliseconds to a few seconds of video. In video conferencing (without switching MCUs involved), a CVS can be as long in duration as the whole session.

##### Picture and Adaptation parameter set

The Picture Parameter Set and the Adaptation Parameter Set (PPS and APS, respectively) carry information pertaining to a single picture. The PPS contains information that is likely to stay constant from picture to picture—at least for pictures for a certain type—whereas the APS contains information, such as adaptive loop filter coefficients, that are likely to change from picture to picture.

#### Profile, level and toolsets

Profiles and levels follow the same design considerations as known from [AVC], [HEVC], and in fact video codecs as old as MPEG-1 visual. A profile defines a set of tools (not to confuse with the "toolset" discussed below) that a decoder compliant with this profile has to support. In [EVC], profiles are defined in Annex A. Formally, they are defined as a set of constraints that a bitstream needs to conform to. In [EVC], the baseline profile is much more severely constrained than main profile, reducing implementation complexity. Levels relate to bitstream complexity in dimensions such as maximum sample decoding rate, maximum picture size, and similar parameters that are directly related to computational complexity.

Profiles and levels are signaled in the highest parameter set available, the SPS.

[EVC] contains another mechanism related to the use of coding tools, known as the toolset syntax element. This syntax element, also located in the SPS, is a bitmask that allows encoders to indicate which coding tools they are using, within the menu of profiles offered by the profile that is also signaled. No decoder conformance point is associated with the toolset, but a bitstream that were using a coding tool that is indicated as not used in the toolset syntax element would obviously be non-compliant. While MPEG specifically rules out the use of the toolset syntax element as a conformance point, walled garden implementations could do so without incurring the interoperability problems MPEG fears, and create bitstreams and decoders that do not support one or more given tools. That, in turn, may be useful to mitigate certain patent related risks.

#### Bitstream and elementary stream

Above the Coded Video Sequence (CVS), [EVC] defines a video bitstream that can be used in the MPEG systems context as an elementary stream. For the purpose of this memo, this is not relevant.

#### Random access support

Editor's note: At this point, the authors believe [EVC] supports only clean random access. WG input is solicited.

Temporal scalability support

[EVC] includes support for temporal scalability through the generalized reference picture selection approach known since [AVC]/SVC. Up to six temporal layers are supported. The temporal layer is signaled in the NAL unit header (which co-serves as the payload header in this memo), in the nuh\_temporal\_id field.

Reference picture management

placeholder

SEI Message

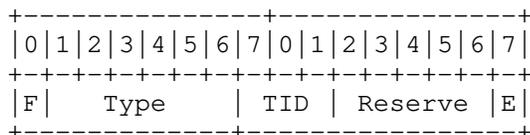
[EVC] inherits many of [HEVC]'s SEI Messages, occasionally with changes in syntax and/or semantics making them applicable to EVC.

1.1.3. Parallel Processing Support (informative)

Placeholder

1.1.4. NAL Unit Header

[EVC] maintains the NAL unit concept of [HEVC] with different parameter options. EVC also uses a two-byte NAL unit header, as shown in Figure 1. The payload of a NAL unit refers to the NAL unit excluding the NAL unit header.



The Structure of the EVC NAL Unit Header

Figure 1

The semantics of the fields in the NAL unit header are as specified in [EVC] and described briefly below for convenience. In addition to the name and size of each field, the corresponding syntax element name in [EVC] is also provided.

F: 1 bit

forbidden\_zero\_bit. Required to be zero in [EVC]. Note that the inclusion of this bit in the NAL unit header was included to enable transport of EVC video over MPEG-2 transport systems

(avoidance of start code emulations) [MPEG2S]. In the context of this memo, the value 1 may be used to indicate a syntax violation, e.g., for a NAL unit resulted from aggregating a number of fragmented units of a NAL unit but missing the last fragment, as described in Section xxx. (section # placeholder)

Type: 6 bits

`nal_unit_type_plus1`. This field specifies the NAL unit type as defined in Table 4 of [EVC]. If the value of this field is less than and equal to 23, the NAL unit is a VCL NAL unit. Otherwise, the NAL unit is a non-VCL NAL unit. For a reference of all currently defined NAL unit types and their semantics, please refer to Section 7.4.2.2 in [EVC].

TID: 3 bits

`nuh_temporal_id`. This field specifies the temporal identifier of the NAL unit. The value of `TemporalId` is equal to TID. `TemporalId` shall be equal to 0 if it is a IDR NAL unit type (NAL unit type 1).

Reserve: 5 bits

`nuh_reserved_zero_5bits`. This field shall be equal to the version of the [EVC] specification. Values of `nuh_reserved_zero_5bits` greater than 0 are reserved for future use by ISO/IEC. Decoders conforming to a profile specified in [EVC] Annex A shall ignore (i.e., remove from the bitstream and discard) all NAL units with values of `nuh_reserved_zero_5bits` greater than 0.

E: 1 bit

`nuh_extension_flag`. This field shall be equal the version of the [EVC] specification. Value of `nuh_extesion_flag` equal to 1 is reserved for future use by ISO/IEC. Decoders conforming to a profile specified in Annex A shall ignore (i.e., remove from the bitstream and discard) all NAL units with values of `nuh_extension_flag` equal to 1.

## 1.2. Overview of the Payload Format

This payload format defines the following processes required for transport of [EVC] coded data over RTP [RFC3550]:

- o Usage of RTP header with this payload format

- o Packetization of [EVC] coded NAL units into RTP packets using three types of payload structures: a single NAL unit, aggregation, and fragment unit packet
- o Transmission of [EVC] NAL units of the same bitstream within a single RTP stream.
- o Media type parameters to be used with the Session Description Protocol (SDP) [RFC4566]
- o Frame-marking mapping [FrameMarking]

## 2. Conventions

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 above.

## 3. Definitions and Abbreviations

### 3.1. Definitions

This document uses the terms and definitions of EVC. Section 3.1.1 lists relevant definitions from [EVC] for convenience. Section 3.1.2 provides definitions specific to this memo.

#### 3.1.1. Definitions from the EVC Specification

**Access Unit:** A set of NAL units that are associated with each other according to a specified classification rule, are consecutive in decoding order, and contain exactly one coded picture.

**Bitstream:** A sequence of bits, in the form of a NAL unit stream or a byte stream, that forms the representation of coded pictures and associated data forming one or more coded video sequences (CVSs).

**Coded Picture:** A coded representation of a picture containing all CTUs of the picture.

**Coded Video Sequence (CVS):** A sequence of access units that consists, in decoding order, of an IDR access unit, followed by zero or more access units that are not IDR access units, including all subsequent access units up to but not including any subsequent access unit that is an IDR access unit.

**Coding Tree Block (CTB):** An  $N \times N$  block of samples for some value of  $N$  such that the division of a component into CTBs is a partitioning.

**Coding Tree Unit (CTU):** A CTB of luma samples, two corresponding CTBs of chroma samples of a picture that has three sample arrays, or a CTB of samples of a monochrome picture or a picture that is coded using three separate colour planes and syntax structures used to code the samples.

**Decoded Picture:** A decoded picture is derived by decoding a coded picture.

**Decoded Picture Buffer (DPB):** A buffer holding decoded pictures for reference, output reordering, or output delay specified for the hypothetical reference decoder in Annex C of [EVC] specification.

**Dynamic Range Adjustment (DRA):** A mapping process that is applied to decoded picture prior to cropping and output as part of the decoding process and is controlled by parameters conveyed in an Adaptation Parameter Set (APS).

**Hypothetical Reference Decoder (HRD):** A hypothetical decoder model that specifies constraints on the variability of conforming NAL unit streams or conforming byte streams that an encoding process may produce.

**Instantaneous Decoding Refresh (IDR) access unit:** An access unit in which the coded picture is an IDR picture.

**Instantaneous Decoding Refresh (IDR) picture:** A coded picture for which each VCL NAL unit has `NalUnitType` equal to `IDR_NUT`.

**Level:** A defined set of constraints on the values that may be taken by the syntax elements and variables of this document, or the value of a transform coefficient prior to scaling.

**Network Abstraction Layer (NAL) unit:** A syntax structure containing an indication of the type of data to follow and bytes containing that data in the form of an RBSP interspersed as necessary.

**Network Abstraction Layer (NAL) Unit Stream:** A sequence of NAL units.

**Non-IDR Picture:** A coded picture that is not an IDR picture.

**Non-VCL NAL Unit:** A NAL unit that is not a VCL NAL unit.

Picture Parameter Set (PPS): A syntax structure containing syntax elements that apply to zero or more entire coded pictures as determined by a syntax element found in each slice header.

Picture Order Count (POC): A variable that is associated with each picture, uniquely identifies the associated picture among all pictures in the CVS, and, when the associated picture is to be output from the decoded picture buffer, indicates the position of the associated picture in output order relative to the output order positions of the other pictures in the same CVS that are to be output from the decoded picture buffer.

Raw Byte Sequence Payload (RBSP): A syntax structure containing an integer number of bytes that is encapsulated in a NAL unit and that is either empty or has the form of a string of data bits containing syntax elements followed by an RBSP stop bit and zero or more subsequent bits equal to 0.

Sequence Parameter Set (SPS): A syntax structure containing syntax elements that apply to zero or more entire CVSs as determined by the content of a syntax element found in the PPS referred to by a syntax element found in each slice header.

Tile row: A rectangular region of CTUs having a height specified by syntax elements in the PPS and a width equal to the width of the picture.

Tile scan: A specific sequential ordering of CTUs partitioning a picture in which the CTUs are ordered consecutively in CTU raster scan in a tile whereas tiles in a picture are ordered consecutively in a raster scan of the tiles of the picture.

Video coding layer (VCL) NAL unit: A collective term for coded slice NAL units and the subset of NAL units that have reserved values of NalUnitType that are classified as VCL NAL units in this document.

### 3.1.2. Definitions Specific to This Memo

Media-Aware Network Element (MANE): A network element, such as a middlebox, selective forwarding unit, or application-layer gateway that is capable of parsing certain aspects of the RTP payload headers or the RTP payload and reacting to their contents.

Editor Notes: the following informative needs to be updated along with frame marking update

Informative note: The concept of a MANE goes beyond normal routers or gateways in that a MANE has to be aware of the signaling (e.g.,

to learn about the payload type mappings of the media streams), and in that it has to be trusted when working with Secure RTP (SRTP). The advantage of using MANEs is that they allow packets to be dropped according to the needs of the media coding. For example, if a MANE has to drop packets due to congestion on a certain link, it can identify and remove those packets whose elimination produces the least adverse effect on the user experience. After dropping packets, MANEs must rewrite RTCP packets to match the changes to the RTP stream, as specified in Section 7 of [RFC3550].

**NAL unit decoding order:** A NAL unit order that conforms to the constraints on NAL unit order given in Section 8.2 and 8.3 in [EVC], follow the Order of NAL units in the bitstream.

**NAL unit output order:** A NAL unit order in which NAL units of different access units are in the output order of the decoded pictures corresponding to the access units, as specified in [EVC], and in which NAL units within an access unit are in their decoding order.

**RTP stream:** See [RFC7656]. Within the scope of this memo, one RTP stream is utilized to transport one or more temporal sub-layers.

**Transmission order:** The order of packets in ascending RTP sequence number order (in modulo arithmetic). Within an aggregation packet, the NAL unit transmission order is the same as the order of appearance of NAL units in the packet.

### 3.2. Abbreviations

APS	Adaptation Parameter Set
ATS	Adaptive Transform Selection
B	Bi-predictive
CBR	Constant Bit Rate
CPB	Coded Picture Buffer
CTB	Coding Tree Block
CTU	Coding Tree Unit
CVS	Coded Video Sequence
DPB	Decoded Picture Buffer

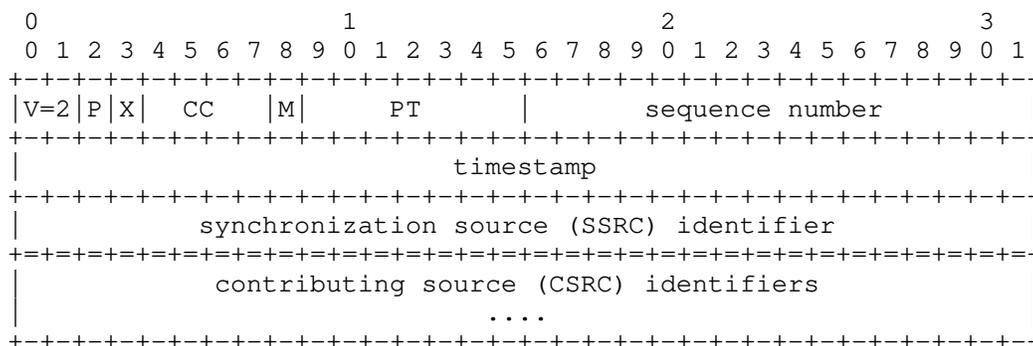
HRD	Hypothetical Reference Decoder
HSS	Hypothetical Stream Scheduler
I	Intra
IDR	Instantaneous Decoding Refresh
LSB	Least Significant Bit
LTRP	Long-Term Reference Picture
MMVD	Merge with Motion Vector Difference
MSB	Most Significant Bit
NAL	Network Abstraction Layer
P	Predictive
POC	Picture Order Count
PPS	Picture Parameter Set
QP	Quantization Parameter
Rbsp	Raw Byte Sequence Payload
RGB	Same as GBR
SAR	Sample Aspect Ratio
SEI	Supplemental Enhancement Information
SODB	String Of Data Bits
SPS	Sequence Parameter Set
STRP	Short-Term Reference Picture
VBR	Variable Bit Rate
VCL	Video Coding Layer

#### 4. RTP Payload Format

##### 4.1. RTP Header Usage

The format of the RTP header is specified in [RFC3550] (reprinted as Figure 2 for convenience). This payload format uses the fields of the header in a manner consistent with that specification.

The RTP payload (and the settings for some RTP header bits) for aggregation packets and fragmentation units are specified in Section 4.3.2 and Section 4.3.3, respectively.



RTP Header According to {{RFC3550}}

Figure 2

The RTP header information to be set according to this RTP payload format is set as follows:

Marker bit (M): 1 bit

Set for the last packet of the access unit, carried in the current RTP stream. This is in line with the normal use of the M bit in video formats to allow an efficient playout buffer handling.

Editor notes: The informative note below needs updating once the NAL unit type table is stable in the [EVC] spec.

Informative note: The content of a NAL unit does not tell whether or not the NAL unit is the last NAL unit, in decoding order, of an access unit. An RTP sender implementation may

obtain this information from the video encoder. If, however, the implementation cannot obtain this information directly from the encoder, e.g., when the bitstream was pre-encoded, and also there is no timestamp allocated for each NAL unit, then the sender implementation can inspect subsequent NAL units in decoding order to determine whether or not the NAL unit is the last NAL unit of an access unit as follows. A NAL unit is determined to be the last NAL unit of an access unit if it is the last NAL unit of the bitstream. A NAL unit `nal_uX` is also determined to be the last NAL unit of an access unit if both the following conditions are true: 1) the next VCL NAL unit `nal_uY` in decoding order has the high-order bit of the first byte after its NAL unit header equal to 1 or `nal_unit_type` equal to 27, and 2) all NAL units between `nal_uX` and `nal_uY`, when present, have `nal_unit_type` in the range of 24 to 26, inclusive, equal to 28 or 29.

Payload Type (PT): 7 bits

The assignment of an RTP payload type for this new payload format is outside the scope of this document and will not be specified here. The assignment of a payload type has to be performed either through the profile used or in a dynamic way.

Sequence Number (SN): 16 bits

Set and used in accordance with [RFC3550].

Timestamp: 32 bits

The RTP timestamp is set to the sampling timestamp of the content. A 90 kHz clock rate MUST be used. If the NAL unit has no timing properties of its own (e.g., parameter sets or certain SEI NAL units), the RTP timestamp MUST be set to the RTP timestamp of the coded picture of the access unit in which the NAL unit (according to Annex D of [EVC]) is included. Receivers MUST use the RTP timestamp for the display process, even when the bitstream contains picture timing SEI messages or decoding unit information SEI messages as specified in [EVC].

Synchronization source (SSRC): 32 bits

Used to identify the source of the RTP packets. When using SRST, by definition a single SSRC is used for all parts of a single bitstream.

#### 4.2. Payload Header Usage

The first two bytes of the payload of an RTP packet are referred to as the payload header. The payload header consists of the same fields (F, TID, Reserve and E) as the NAL unit header as shown in Section 1.1.4, irrespective of the type of the payload structure.

The TID value indicates (among other things) the relative importance of an RTP packet, for example, because NAL units belonging to higher temporal sub-layers are not used for the decoding of lower temporal sub-layers. A lower value of TID indicates a higher importance. More-important NAL units MAY be better protected against transmission losses than less-important NAL units.

#### 4.3. Payload Structures

Three different types of RTP packet payload structures are specified. A receiver can identify the type of an RTP packet payload through the Type field in the payload header.

The Three different payload structures are as follows:

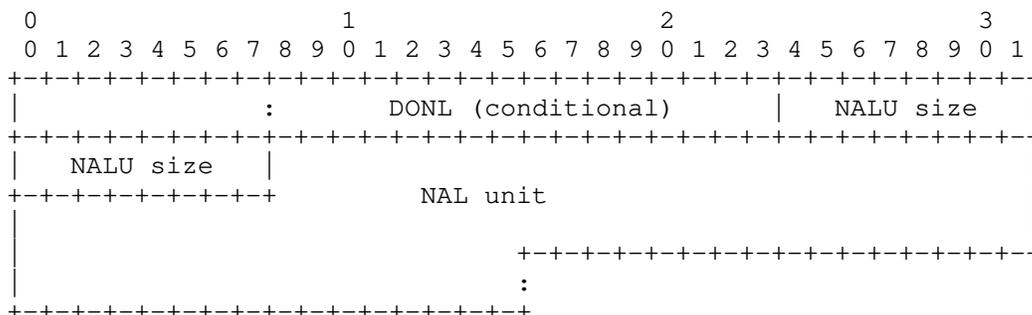
- o Single NAL unit packet: Contains a single NAL unit in the payload, and the NAL unit header of the NAL unit also serves as the payload header. This payload structure is specified in Section 4.3.1.
- o Aggregation Packet (AP): Contains more than one NAL unit within one access unit. This payload structure is specified in Section 4.3.2.
- o Fragmentation Unit (FU): Contains a subset of a single NAL unit. This payload structure is specified in Section 4.3.3.

##### 4.3.1. Single NAL Unit Packets

A single NAL unit packet contains exactly one NAL unit, and consists of a payload header (denoted as PayloadHdr), a conditional 16-bit DONL field (in network byte order), and the NAL unit payload data (the NAL unit excluding its NAL unit header) of the contained NAL unit, as shown in Figure 3.







The Structure of the First Aggregation Unit in an AP

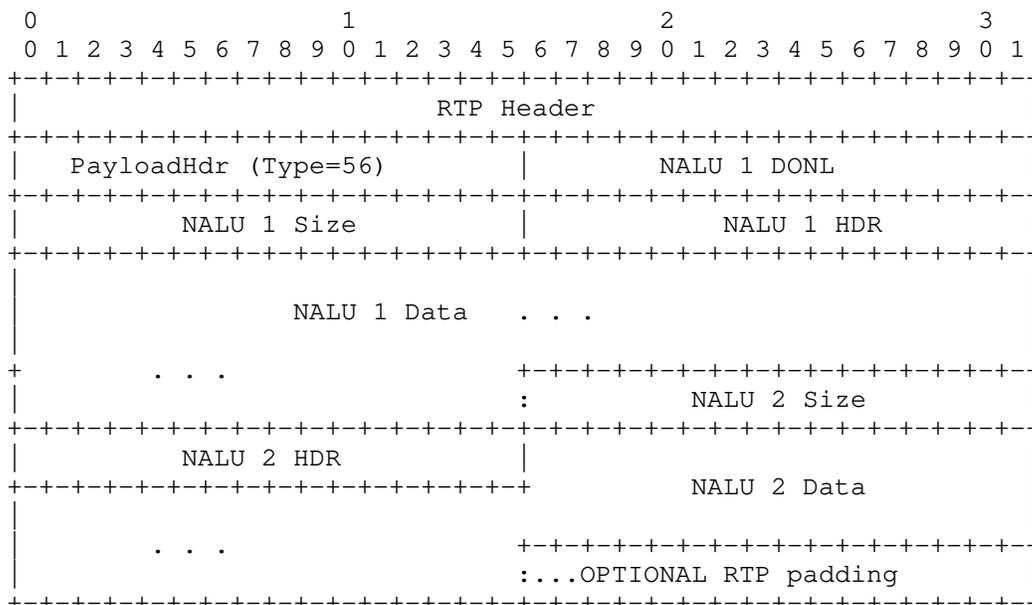
Figure 5

The DONL field, when present, specifies the value of the 16 least significant bits of the decoding order number of the aggregated NAL unit.

If `sprop-max-don-diff` is greater than 0 for any of the RTP streams, the DONL field MUST be present in an aggregation unit that is the first aggregation unit in an AP, and the variable DON for the aggregated NAL unit is derived as equal to the value of the DONL field. Otherwise (`sprop-max-don-diff` is equal to 0 for all the RTP streams), the DONL field MUST NOT be present in an aggregation unit that is the first aggregation unit in an AP.

An aggregation unit that is not the first aggregation unit in an AP will be followed immediately by a 16-bit unsigned size information (in network byte order) that indicates the size of the NAL unit in bytes (excluding these two octets, but including the NAL unit header), followed by the NAL unit itself, including its NAL unit header, as shown in Figure 6.





An Example of an AP Containing  
Two Aggregation Units with the DONL Field

Figure 8

### 4.3.3. Fragmentation Units

Fragmentation Units (FUs) are introduced to enable fragmenting a single NAL unit into multiple RTP packets, possibly without cooperation or knowledge of the EVC [EVC] encoder. A fragment of a NAL unit consists of an integer number of consecutive octets of that NAL unit. Fragments of the same NAL unit MUST be sent in consecutive order with ascending RTP sequence numbers (with no other RTP packets within the same RTP stream being sent between the first and last fragment).

When a NAL unit is fragmented and conveyed within FUs, it is referred to as a fragmented NAL unit. APs MUST NOT be fragmented. FUs MUST NOT be nested; i.e., an FU must not contain a subset of another FU.

The RTP timestamp of an RTP packet carrying an FU is set to the NALU-time of the fragmented NAL unit.

An FU consists of a payload header (denoted as PayloadHdr), an FU header of one octet, a conditional 16-bit DONL field (in network byte order), and an FU payload, as shown in Figure 9.



When set to 1, the E bit indicates the end of a fragmented NAL unit, i.e., the last byte of the payload is also the last byte of the fragmented NAL unit. When the FU payload is not the last fragment of a fragmented NAL unit, the E bit MUST be set to 0.

FuType: 6 bits

The field FuType MUST be equal to the field Type of the fragmented NAL unit.

The DONL field, when present, specifies the value of the 16 least significant bits of the decoding order number of the fragmented NAL unit.

If sprop-max-don-diff is greater than 0 for any of the RTP streams, and the S bit is equal to 1, the DONL field MUST be present in the FU, and the variable DON for the fragmented NAL unit is derived as equal to the value of the DONL field. Otherwise (sprop-max-don-diff is equal to 0 for all the RTP streams, or the S bit is equal to 0), the DONL field MUST NOT be present in the FU.

A non-fragmented NAL unit MUST NOT be transmitted in one FU; i.e., the Start bit and End bit must not both be set to 1 in the same FU header.

The FU payload consists of fragments of the payload of the fragmented NAL unit so that if the FU payloads of consecutive FUs, starting with an FU with the S bit equal to 1 and ending with an FU with the E bit equal to 1, are sequentially concatenated, the payload of the fragmented NAL unit can be reconstructed. The NAL unit header of the fragmented NAL unit is not included as such in the FU payload, but rather the information of the NAL unit header of the fragmented NAL unit is conveyed in F, TID, Reserve and E fields of the FU payload headers of the FUs and the FuType field of the FU header of the FUs. An FU payload MUST NOT be empty.

If an FU is lost, the receiver SHOULD discard all following fragmentation units in transmission order corresponding to the same fragmented NAL unit, unless the decoder in the receiver is known to gracefully handle incomplete NAL units.

A receiver in an endpoint or in a MANE MAY aggregate the first n-1 fragments of a NAL unit to an (incomplete) NAL unit, even if fragment n of that NAL unit is not received. In this case, the forbidden\_zero\_bit of the NAL unit MUST be set to 1 to indicate a syntax violation.

#### 4.4. Decoding Order Number

For each NAL unit, the variable `AbsDon` is derived, representing the decoding order number that is indicative of the NAL unit decoding order.

Let NAL unit `n` be the `n`-th NAL unit in transmission order within an RTP stream.

If `sprop-max-don-diff` is equal to 0 for all the RTP streams carrying the HEVC bitstream, `AbsDon[n]`, the value of `AbsDon` for NAL unit `n`, is derived as equal to `n`.

Otherwise (`sprop-max-don-diff` is greater than 0 for any of the RTP streams), `AbsDon[n]` is derived as follows, where `DON[n]` is the value of the variable `DON` for NAL unit `n`:

- o If `n` is equal to 0 (i.e., NAL unit `n` is the very first NAL unit in transmission order), `AbsDon[0]` is set equal to `DON[0]`.
- o Otherwise (`n` is greater than 0), the following applies for derivation of `AbsDon[n]`:

If `DON[n] == DON[n-1]`,  
`AbsDon[n] = AbsDon[n-1]`

If (`DON[n] > DON[n-1]` and `DON[n] - DON[n-1] < 32768`),  
`AbsDon[n] = AbsDon[n-1] + DON[n] - DON[n-1]`

If (`DON[n] < DON[n-1]` and `DON[n-1] - DON[n] >= 32768`),  
`AbsDon[n] = AbsDon[n-1] + 65536 - DON[n-1] + DON[n]`

If (`DON[n] > DON[n-1]` and `DON[n] - DON[n-1] >= 32768`),  
`AbsDon[n] = AbsDon[n-1] - (DON[n-1] + 65536 - DON[n])`

If (`DON[n] < DON[n-1]` and `DON[n-1] - DON[n] < 32768`),  
`AbsDon[n] = AbsDon[n-1] - (DON[n-1] - DON[n])`

For any two NAL units `m` and `n`, the following applies:

- o `AbsDon[n]` greater than `AbsDon[m]` indicates that NAL unit `n` follows NAL unit `m` in NAL unit decoding order.
- o When `AbsDon[n]` is equal to `AbsDon[m]`, the NAL unit decoding order of the two NAL units can be in either order.

- o AbsDon[n] less than AbsDon[m] indicates that NAL unit n precedes NAL unit m in decoding order.

Informative note: When two consecutive NAL units in the NAL unit decoding order have different values of AbsDon, the absolute difference between the two AbsDon values may be greater than or equal to 1.

Informative note: There are multiple reasons to allow for the absolute difference of the values of AbsDon for two consecutive NAL units in the NAL unit decoding order to be greater than one. An increment by one is not required, as at the time of associating values of AbsDon to NAL units, it may not be known whether all NAL units are to be delivered to the receiver. For example, a gateway might not forward VCL NAL units of higher sub-layers or some SEI NAL units when there is congestion in the network. In another example, the first intra-coded picture of a pre-encoded clip is transmitted in advance to ensure that it is readily available in the receiver, and when transmitting the first intra-coded picture, the originator does not exactly know how many NAL units will be encoded before the first intra-coded picture of the pre-encoded clip follows in decoding order. Thus, the values of AbsDon for the NAL units of the first intra-coded picture of the pre-encoded clip have to be estimated when they are transmitted, and gaps in values of AbsDon may occur.

## 5. Packetization Rules

The following packetization rules apply:

- o If sprop-max-don-diff is greater than 0 for any of the RTP streams, the transmission order of NAL units carried in the RTP stream MAY be different than the NAL unit decoding order and the NAL unit output order.
- o A NAL unit of a small size SHOULD be encapsulated in an aggregation packet together with one or more other NAL units in order to avoid unnecessary packetization overhead for small NAL units. For example, non-VCL NAL units such as access unit delimiters, parameter sets, or SEI NAL units are typically small and can often be aggregated with VCL NAL units without violating MTU size constraints.

- o Each non-VCL NAL unit SHOULD, when possible from an MTU size match viewpoint, be encapsulated in an aggregation packet together with its associated VCL NAL unit, as typically a non-VCL NAL unit would be meaningless without the associated VCL NAL unit being available.
- o For carrying exactly one NAL unit in an RTP packet, a single NAL unit packet MUST be used.

## 6. De-packetization Process

The general concept behind de-packetization is to get the NAL units out of the RTP packets in an RTP stream and pass them to the decoder in the NAL unit decoding order.

The de-packetization process is implementation dependent. Therefore, the following description should be seen as an example of a suitable implementation. Other schemes may be used as well, as long as the output for the same input is the same as the process described below. The output is the same when the set of output NAL units and their order are both identical. Optimizations relative to the described algorithms are possible.

All normal RTP mechanisms related to buffer management apply. In particular, duplicated or outdated RTP packets (as indicated by the RTP sequence number and the RTP timestamp) are removed. To determine the exact time for decoding, factors such as a possible intentional delay to allow for proper inter-stream synchronization must be factored in.

NAL units with NAL unit type values in the range of 0 to 55, inclusive, may be passed to the decoder. NAL-unit-like structures with NAL unit type values in the range of 56 to 63, inclusive, MUST NOT be passed to the decoder.

The receiver includes a receiver buffer, which is used to compensate for transmission delay jitter within individual RTP streams and across RTP streams, to reorder NAL units from transmission order to the NAL unit decoding order. In this section, the receiver operation is described under the assumption that there is no transmission delay jitter within an RTP stream. To make a difference from a practical receiver buffer that is also used for compensation of transmission delay jitter, the receiver buffer is hereafter called the de-packetization buffer in this section. Receivers should also prepare for transmission delay jitter; that is, either reserve separate buffers for transmission delay jitter buffering and de-packetization buffering or use a receiver buffer for both transmission delay jitter and de-packetization. Moreover, receivers should take transmission

delay jitter into account in the buffering operation, e.g., by additional initial buffering before starting of decoding and playback.

When `sprop-max-don-diff` is equal to 0 for the received RTP stream, the de-packetization buffer size is zero bytes, and the process described in the remainder of this paragraph applies. The NAL units carried in the RTP stream are directly passed to the decoder in their transmission order, which is identical to their decoding order. When there are several NAL units of the same RTP stream with the same NTP timestamp, the order to pass them to the decoder is their transmission order.

Informative note: The mapping between RTP and NTP timestamps is conveyed in RTCP SR packets. In addition, the mechanisms for faster media timestamp synchronization discussed in [RFC6051] may be used to speed up the acquisition of the RTP-to-wall-clock mapping.

When `sprop-max-don-diff` is greater than 0 for the received RTP stream the process described in the remainder of this section applies.

There are two buffering states in the receiver: initial buffering and buffering while playing. Initial buffering starts when the reception is initialized. After initial buffering, decoding and playback are started, and the buffering-while-playing mode is used.

Regardless of the buffering state, the receiver stores incoming NAL units, in reception order, into the de-packetization buffer. NAL units carried in RTP packets are stored in the de-packetization buffer individually, and the value of `AbsDon` is calculated and stored for each NAL unit.

Initial buffering lasts until condition A (the difference between the greatest and smallest `AbsDon` values of the NAL units in the de-packetization buffer is greater than or equal to the value of `sprop-max-don-diff`) or condition B (the number of NAL units in the de-packetization buffer is greater than the value of `sprop-depack-buf-nalus`) is true.

After initial buffering, whenever condition A or condition B is true, the following operation is repeatedly applied until both condition A and condition B become false:

- o The NAL unit in the de-packetization buffer with the smallest value of `AbsDon` is removed from the de-packetization buffer and passed to the decoder.

When no more NAL units are flowing into the de-packetization buffer, all NAL units remaining in the de-packetization buffer are removed from the buffer and passed to the decoder in the order of increasing AbsDon values.

## 7. Payload Format Parameters

Placeholder

## 8. Use with Feedback Messages

Placeholder

### 8.1. Picture Loss Indication (PLI)

Placeholder

### 8.2. Slice Loss Indication (SLI)

Placeholder

### 8.3. Reference Picture Selection Indication (RPSI)

Placeholder

### 8.4. Full Intra Request (FIR)

Placeholder

## 9. Frame marking

placeholder

## 10. Security Considerations

The scope of this Security Considerations section is limited to the payload format itself and to one feature of [EVC] that may pose a particularly serious security risk if implemented naively. The payload format, in isolation, does not form a complete system. Implementers are advised to read and understand relevant security-related documents, especially those pertaining to RTP (see the Security Considerations section in [RFC3550] ), and the security of the call-control stack chosen (that may make use of the media type registration of this memo). Implementers should also consider known security vulnerabilities of video coding and decoding implementations in general and avoid those.

Within this RTP payload format, neither the various media-plane-based mechanisms, nor the signaling part of this memo, seems to pose a security risk beyond those common to all RTP-based systems.

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. However, as "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format's responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this section discusses the security impacting properties of the payload format itself.

Because the data compression used with this payload format is applied end-to-end, any encryption needs to be performed after compression. A potential denial-of-service threat exists for data encodings using compression techniques that have non-uniform receiver-end computational load. The attacker can inject pathological datagrams into the bitstream that are complex to decode and that cause the receiver to be overloaded. EVC is particularly vulnerable to such attacks, as it is extremely simple to generate datagrams containing NAL units that affect the decoding process of many future NAL units. Therefore, the usage of data origin authentication and data integrity protection of at least the RTP packet is RECOMMENDED, for example, with SRTP [RFC3711].

End-to-end security with authentication, integrity, or confidentiality protection will prevent a MANE from performing media-aware operations other than discarding complete packets. In the case of confidentiality protection, it will even be prevented from discarding packets in a media-aware way. To be allowed to perform such operations, a MANE is required to be a trusted entity that is included in the security context establishment.

## 11. Congestion Control

Congestion control for RTP SHALL be used in accordance with RTP [RFC3550] and with any applicable RTP profile, e.g., AVP [RFC3551]. If best-effort service is being used, an additional requirement is that users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within an acceptable range. Packet loss

is considered acceptable if a TCP flow across the same network path, and experiencing the same network conditions, would achieve an average throughput, measured on a reasonable timescale, that is not less than all RTP streams combined is achieving. This condition can be satisfied by implementing congestion-control mechanisms to adapt the transmission rate, the number of layers subscribed for a layered multicast session, or by arranging for a receiver to leave the session if the loss rate is unacceptably high.

The bitrate adaptation necessary for obeying the congestion control principle is easily achievable when real-time encoding is used, for example, by adequately tuning the quantization parameter. However, when pre-encoded content is being transmitted, bandwidth adaptation requires the pre-coded bitstream to be tailored for such adaptivity. The key mechanism available in [EVC] is temporal scalability. A media sender can remove NAL units belonging to higher temporal sub-layers (i.e., those NAL units with a high value of TID) until the sending bitrate drops to an acceptable range.

The mechanisms mentioned above generally work within a defined profile and level and, therefore, no renegotiation of the channel is required. Only when non-downgradable parameters (such as profile) are required to be changed does it become necessary to terminate and restart the RTP stream(s). This may be accomplished by using different RTP payload types.

MANES MAY remove certain unusable packets from the RTP stream when that RTP stream was damaged due to previous packet losses. This can help reduce the network load in certain special cases. For example, MANES can remove those FUs where the leading FUs belonging to the same NAL unit have been lost or those dependent slice segments when the leading slice segments belonging to the same slice have been lost, because the trailing FUs or dependent slice segments are meaningless to most decoders. MANES can also remove higher temporal scalable layers if the outbound transmission (from the MANE's viewpoint) experiences congestion.

## 12. IANA Considerations

Placeholder

## 13. Acknowledgements

Large parts of this specification share text with the RTP payload format for HEVC [RFC7798]. We thank the authors of that specification for their excellent work.

## 14. References

### 14.1. Normative References

- [EVC] "ISO/IEC FDIS 23094-1 Essential Video Coding", 2020, <<https://www.iso.org/standard/57797.html>>.
- [ISO23094-1] "ISO/IEC DIS Information technology --- General video coding --- Part 1 Essential video coding", n.d., <<https://www.iso.org/standard/57797.html>>.
- [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>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/info/rfc3711>>.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <<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>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/info/rfc5104>>.

- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/info/rfc5124>>.
- [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>>.
- [RFC8082] Wenger, S., Lennox, J., Burman, B., and M. Westerlund, "Using Codec Control Messages in the RTP Audio-Visual Profile with Feedback with Layered Codecs", RFC 8082, DOI 10.17487/RFC8082, March 2017, <<https://www.rfc-editor.org/info/rfc8082>>.
- [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>>.

#### 14.2. Informative References

- [AVC] "ITU-T Recommendation H.264 - Advanced video coding for generic audiovisual services", 2014, <<https://www.iso.org/standard/66069.html>>.
- [draft-ietf-avtcore-rtp-vcv]  
Zhao, S, . and . Wenger, S, "RTP payload format for VVC", Work in Progress draft-ietf-avtcore-rtp-vcv , 2020, <<https://datatracker.ietf.org/doc/draft-ietf-avtcore-rtp-vcv/>>.
- [FrameMarking]  
Berger, E, ., Nandakumar, S, ., and . Zanaty M, "Frame Marking RTP Header Extension", Work in Progress draft-berger-avtext-framemarking , 2015.
- [HEVC] "High efficiency video coding, ITU-T Recommendation H.265", 2017, <<https://www.iso.org/standard/69668.html>>.
- [MPEG2S] ISO/IEC, ., "Information technology - Generic coding of moving pictures and associated audio information - Part 1: Systems, ISO International Standard 13818-1", 2013.
- [RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, DOI 10.17487/RFC6051, November 2010, <<https://www.rfc-editor.org/info/rfc6051>>.

- [RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <<https://www.rfc-editor.org/info/rfc6184>>.
- [RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <<https://www.rfc-editor.org/info/rfc6190>>.
- [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>>.
- [RFC7798] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S., and M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <<https://www.rfc-editor.org/info/rfc7798>>.
- [VVC] "ISO/IEC DIS 23090-3 Versatile Video Coding (Draft 8), Joint Video Experts Team (JVET)", January 2020, <<https://www.iso.org/standard/73022.html>>.

## Authors' Addresses

Shuai Zhao  
Tencent  
2747 Park Blvd  
Palo Alto 94588  
USA

Email: [shuai.zhao@ieee.org](mailto:shuai.zhao@ieee.org)

Stephan Wenger  
Tencent  
2747 Park Blvd  
Palo Alto 94588

Email: [stewe@stewe.org](mailto:stewe@stewe.org)