

Workgroup: AVTCore
Internet-Draft:
draft-ietf-avtcore-multi-party-rtt-mix-20
Updates: [4103](#) (if approved)
Published: 26 May 2021
Intended Status: Standards Track
Expires: 27 November 2021
Authors: G. Hellstrom
Gunnar Hellstrom Accessible Communication
RTP-mixer formatting of multiparty Real-time text

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](#)
 - 1.1. [Terminology](#)
 - 1.2. [Selected solution and considered alternatives](#)
 - 1.3. [Intended application](#)
2. [Overview of the two specified solutions and selection of method](#)
 - 2.1. [The RTP-mixer-based solution for multiparty-aware endpoints](#)
 - 2.2. [Mixing for multiparty-unaware endpoints](#)
 - 2.3. [Offer/answer considerations](#)
 - 2.4. [Actions depending on capability negotiation result](#)
3. [Details for the RTP-mixer-based mixing method for multiparty-aware endpoints](#)
 - 3.1. [Use of fields in the RTP packets](#)
 - 3.2. [Initial transmission of a BOM character](#)
 - 3.3. [Keep-alive](#)
 - 3.4. [Transmission interval](#)
 - 3.5. [Only one source per packet](#)
 - 3.6. [Do not send received text to the originating source](#)
 - 3.7. [Clean incoming text](#)
 - 3.8. [Redundant transmission principles](#)
 - 3.9. [Text placement in packets](#)
 - 3.10. [Empty T140blocks](#)
 - 3.11. [Creation of the redundancy](#)

- [3.12. Timer offset fields](#)
- [3.13. Other RTP header fields](#)
- [3.14. Pause in transmission](#)
- [3.15. RTCP considerations](#)
- [3.16. Reception of multiparty contents](#)
- [3.17. Performance considerations](#)
- [3.18. Security for session control and media](#)
- [3.19. SDP offer/answer examples](#)
- [3.20. Packet sequence example from interleaved transmission](#)
- [3.21. Maximum character rate "cps"](#)
- [4. Presentation level considerations](#)
 - [4.1. Presentation by multiparty-aware endpoints](#)
 - [4.2. Multiparty mixing for multiparty-unaware endpoints](#)
- [5. Relation to Conference Control](#)
 - [5.1. Use with SIP centralized conferencing framework](#)
 - [5.2. Conference control](#)
- [6. Gateway Considerations](#)
 - [6.1. Gateway considerations with Textphones](#)
 - [6.2. Gateway considerations with WebRTC](#)
- [7. Updates to RFC 4103](#)
- [8. Congestion considerations](#)
- [9. IANA Considerations](#)
 - [9.1. Registration of the "rtt-mixer" SDP media attribute](#)
- [10. Security Considerations](#)
- [11. Change history](#)
 - [11.1. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-20](#)
 - [11.2. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-19](#)
 - [11.3. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-18](#)
 - [11.4. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-17](#)
 - [11.5. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-16](#)
 - [11.6. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-15](#)
 - [11.7. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-14](#)
 - [11.8. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-13](#)
 - [11.9. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-12](#)
 - [11.10. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-11](#)
 - [11.11. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-10](#)
 - [11.12. Changes included in draft-ietf-avtcore-multi-party-rtt-mix-09](#)

11.13.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-08
11.14.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-07
11.15.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-06
11.16.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-05
11.17.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-04
11.18.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-03
11.19.	Changes included in draft-ietf-avtcore-multi-party-rtt-mix-02
11.20.	Changes to draft-ietf-avtcore-multi-party-rtt-mix-01
11.21.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-03 to draft-ietf-avtcore-multi-party-rtt-mix-00
11.22.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-02 to -03
11.23.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-01 to -02
11.24.	Changes from draft-hellstrom-avtcore-multi-party-rtt-source-00 to -01
12.	References
12.1.	Normative References
12.2.	Informative References
	Acknowledgements
	Author's Address

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 (SDS), 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	-
	v
<Eve-typing> But I need to be back to	^
the hotel by 11 because I need	-
<Bob-typing> I wou	-
	v
of course, I underst	

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
		I will arrive by TGV.
My flight is to Orly		Convenient to the main
	Hi all, can we plan	station.
	for the seminar?	
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.

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
	I will arrive by TGV.
[Bob]:My flight is to	Convenient to the main
Only.	station.
[Eve]:Hi all, can we	
plan for the seminar.	
[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	
	-
	v
of course, I underst	

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-avtc core-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-avtc core-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-avtc core-multi-party-rtt-mix-17

Actions on Gen-ART review comments.

Actions on SecDir review comments.

11.5. Changes included in draft-ietf-avtc core-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-avtc core-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-avtc core-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-avtcore-multi-party-rtt-mix-12

Changes according to responses on comments from Brian Rosen in Avtcore list on 2020-12-05 and -06.

Changes according to responses to comments by Bernard Aboba in avtcore list 2020-12-06.

Introduction of an optional 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-avtcore-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-avtcore-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-avtcore-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 fmtp-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 (SRTP)", 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