AVTCORE
Internet-Draft
Updates: 3550, 4585 (if approved)
Intended status: Standards Track
Expires: April 30, 2015

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

October 27, 2014

Sending Multiple Media Streams in a Single RTP Session
draft-ietf-avtcore-rtp-multi-stream-06

Abstract

This memo expands and clarifies the behaviour of Real-time Transport Protocol (RTP) endpoints that use multiple synchronization sources (SSRCs). This occurs, for example, when an endpoint sends multiple media streams in a single RTP session. This memo updates RFC 3550 with regards to handling multiple SSRCs per endpoint in RTP sessions, with a particular focus on RTCP behaviour. It also updates RFC 4585 to update and clarify the calculation of the timeout of SSRCs and the inclusion of feedback messages.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 30, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction 2
2. Terminology 3
3. Use Cases For Multi-Stream Endpoints 3
3.1. Endpoints with Multiple Capture Devices 3
3.2. Multiple Media Types in a Single RTP Session 3
3.3. Multiple Stream Mixers 4
3.4. Multiple SSRCs for a Single Media Source 4
4. Use of RTP by endpoints that send multiple media streams . . 5
5. Use of RTCP by Endpoints that send multiple media streams . . 5
5.1. RTCP Reporting Requirement 5
5.2. Initial Reporting Interval 5
5.3. Aggregation of Reports into Compound RTCP Packets 6
5.3.1. Maintaining AVG_RTCP_SIZE 7
5.3.2. Scheduling RTCP with Multiple Reporting SSRCs 8
5.4. Use of RTP/AVPF Feedback 10
5.4.1. Choice of SSRC for Feedback Packets 10
5.4.2. Scheduling an RTCP Feedback Packet 11
6. RTCP Considerations for Streams with Disparate Rates 12
6.1. Timing out SSRCs 13
6.1.1. Problems with RTP/AVPF the T_rr_interval Parameter . 13
6.1.2. Avoiding Premature Timeout 14
6.1.3. Interoperability Between RTP/AVP and RTP/AVPF 15
6.1.4. Updated SSRC Timeout Rules 15
6.2. Tuning RTCP transmissions 16
6.2.1. RTP/AVP and RTP/SAVP 16
6.2.2. RTP/AVPF and RTP/SAVPF 18
7. Security Considerations 19
8. IANA Considerations 19
9. Open Issues 19
10. References 20
10.1. Normative References 20
10.2. Informative References 20
Authors' Addresses 21

1. Introduction

At the time the Real-Time Transport Protocol (RTP) [RFC3550] was originally designed, and for quite some time after, endpoints in RTP sessions typically only transmitted a single media stream, and thus used a single synchronization source (SSRC) per RTP session, where separate RTP sessions were typically used for each distinct media type. Recently, however, a number of scenarios have emerged in which endpoints wish to send multiple RTP media streams, distinguished by distinct RTP synchronization source (SSRC) identifiers, in a single RTP session. These are outlined in Section 3. Although the initial design of RTP did consider such scenarios, the specification was not consistently written with such use cases in mind. The specifications are thus somewhat unclear.

This memo updates [RFC3550] to clarify behaviour in use cases where endpoints use multiple SSRCs. It also updates [RFC4585] in regards to the timeout of inactive SSRCs to resolve problematic behaviour as well as clarifying the inclusion of feedback messages.

2. 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 RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

3. Use Cases For Multi-Stream Endpoints

This section discusses several use cases that have motivated the development of endpoints that sends RTP data using multiple SSRCs in a single RTP session.

3.1. Endpoints with Multiple Capture Devices

The most straightforward motivation for an endpoint to send multiple simultaneous RTP streams in a session is the scenario where an endpoint has multiple capture devices, and thus media sources, of the same media type and characteristics. For example, telepresence endpoints, of the type described by the CLUE Telepresence Framework [I-D.ietf-clue-framework], often have multiple cameras or microphones covering various areas of a room, and hence send several RTP streams.

3.2. Multiple Media Types in a Single RTP Session

Recent work has updated RTP [I-D.ietf-avtcore-multi-media-rtp-session] and SDP [I-D.ietf-mmusic-sdp-bundle-negotiation] to remove the historical assumption in RTP that media sources of different media types would

always be sent on different RTP sessions. In this work, a single endpoint's audio and video RTP media streams (for example) are instead sent in a single RTP session to reduce the number of transport layer flows used.

3.3. Multiple Stream Mixers

There are several RTP topologies which can involve a central device that itself generates multiple RTP media streams in a session. An example is a mixer providing centralized compositing for a multi-capture scenario like that described in Section 3.1. In this case, the centralized node is behaving much like a multi-capturer endpoint, generating several similar and related sources.

A more complex example is the selective forwarding middlebox, described in Section 3.7 of [I-D.ietf-avtcore-rtp-topologies-update]. This is a middlebox that receives media streams from several endpoints, and then selectively forwards modified versions of some RTP streams toward the other endpoints to which it is connected. For each connected endpoint, a separate media source appears in the session for every other source connected to the middlebox, "projected" from the original streams, but at any given time many of them can appear to be inactive (and thus are receivers, not senders, in RTP). This sort of device is closer to being an RTP mixer than an RTP translator, in that it terminates RTCP reporting about the mixed streams, and it can re-write SSRCs, timestamps, and sequence numbers, as well as the contents of the RTP payloads, and can turn sources on and off at will without appearing to be generating packet loss. Each projected stream will typically preserve its original RTCP source description (SDES) information.

3.4. Multiple SSRCs for a Single Media Source

There are also several cases where a single media source results in the usage of multiple SSRCs within the same RTP session. Transport robustness tools like RTP Retransmission [RFC4588] result in multiple SSRCs, one with source data, and another with the repair data. Scalable encoders and their RTP payload formats, like H.264's extension for Scalable Video Coding(SVC) [RFC6190] can be transmitted in a configuration where the scalable layers are distributed over multiple SSRCs within the same session, to enable RTP packet stream level (SSRC) selection and routing in conferencing middleboxes.

4. Use of RTP by endpoints that send multiple media streams

Every RTP endpoint will have an allocated share of the available session bandwidth, as determined by signalling and congestion control. The endpoint MUST keep its total media sending rate within this share. However, endpoints that send multiple media streams do not necessarily need to subdivide their share of the available bandwidth independently or uniformly to each media stream and its SSRCs. In particular, an endpoint can vary the allocation to different streams depending on their needs, and can dynamically change the bandwidth allocated to different SSRCs (for example, by using a variable rate codec), provided the total sending rate does not exceed its allocated share. This includes enabling or disabling media streams and their redundancy streams as more or less bandwidth becomes available.

5. Use of RTCP by Endpoints that send multiple media streams

The RTP Control Protocol (RTCP) is defined in Section 6 of [RFC3550]. The description of the protocol is phrased in terms of the behaviour of "participants" in an RTP session, under the assumption that each endpoint is a participant with a single SSRC. However, for correct operation in cases where endpoints can send multiple media streams, the specification needs to be interpreted with each SSRC counting as a participant in the session, so that an endpoint that has multiple SSRCs counts as multiple participants. The following describes several concrete cases where this applies.

5.1. RTCP Reporting Requirement

An RTP endpoint that has multiple SSRCs MUST treat each SSRC as a separate participant in the RTP session, sending RTCP reports for each of its SSRCs in every RTCP reporting interval. If the mechanism in [I-D.ietf-avtcore-rtp-multi-stream-optimisation] is not used, then each SSRC will send RTCP reports for all other SSRCs, including those co-located at the same endpoint.

If the endpoint has some SSRCs that are sending data and some that are only receivers, then they will receive different shares of the RTCP bandwidth and calculate different base RTCP reporting intervals. Otherwise, all SSRCs at an endpoint will calculate the same base RTCP reporting interval. The actual reporting intervals for each SSRC are randomised in the usual way, but reports can be aggregated as described in Section 5.3.

5.2. Initial Reporting Interval

When a participant joins a unicast session, the following text from Section 6.2 of [RFC3550] applies: "For unicast sessions... the delay before sending the initial compound RTCP packet MAY be zero." This also applies to the individual SSRCs of an endpoint that has multiple SSRCs, and such endpoints MAY send an initial RTCP packet for each of their SSRCs immediately upon joining a unicast session.

Caution has to be exercised, however, when an endpoint (or middlebox) with a large number of SSRCs joins a unicast session, since immediate transmission of many RTCP reports can create a significant burst of traffic, leading to transient congestion and packet loss due to queue overflows. Implementers are advised to consider sending immediate RTCP packets for only a small number of SSRCs (e.g., the one or two SSRCs they consider most important), with the initial RTCP packets for their other SSRCs being sent after the calculated initial RTCP reporting interval, to avoid self congestion.

(TBD: is this recommendation sufficiently strong?)

5.3. Aggregation of Reports into Compound RTCP Packets

As outlined in Section 5.1, an endpoint with multiple SSRCs has to treat each SSRC as a separate participant when it comes to sending RTCP reports. This will lead to each SSRC sending a compound RTCP packet in each reporting interval. Since these packets are coming from the same endpoint, it might reasonably be expected that they can be aggregated to reduce overheads. Indeed, Section 6.1 of [RFC3550] allows RTP translators and mixers to aggregate packets in similar circumstances:

> "It is RECOMMENDED that translators and mixers combine individual RTCP packets from the multiple sources they are forwarding into one compound packet whenever feasible in order to amortize the packet overhead (see Section 7). An example RTCP compound packet as might be produced by a mixer is shown in Fig. 1. If the overall length of a compound packet would exceed the MTU of the network path, it SHOULD be segmented into multiple shorter compound packets to be transmitted in separate packets of the underlying protocol. This does not impair the RTCP bandwidth estimation because each compound packet represents at least one distinct participant. Note that each of the compound packets MUST begin with an SR or RR packet."

The allows RTP translators and mixers to generate compound RTCP packets that contain multiple SR or RR packets from different SSRCs, as well as any of the other packet types. There are no restrictions on the order in which the RTCP packets can occur within the compound packet, except the regular rule that the compound RTCP packet starts

with an SR or RR packet. Due to this rule, correctly implemented RTP endpoints will be able to handle compound RTCP packets that contain RTCP packets relating to multiple SSRCs.

Accordingly, endpoints that use multiple SSRCs MAY aggregate the RTCP packets sent by their different SSRCs into compound RTCP packets, provided 1) the resulting compound RTCP packets begin with an SR or RR packet; 2) they maintain the average RTCP packet size as described in Section 5.3.1; and 3) they schedule packet transmission and manage aggregation as described in Section 5.3.2.

5.3.1. Maintaining AVG_RTCP_SIZE

The RTCP scheduling algorithm in [RFC3550] works on a per-SSRC basis. Each SSRC sends a single compound RTCP packet in each RTCP reporting interval. When an endpoint uses multiple SSRCs, it is desirable to aggregate the compound RTCP packets sent by its SSRCs, reducing the overhead by forming a larger compound RTCP packet. This aggregation can be done as described in Section 5.3.2, provided the average RTCP packet size calculation is updated as follows.

Participants in an RTP session update their estimate of the average RTCP packet size (avg_rtcp_size) each time they send or receive an RTCP packet (see Section 6.3.3 of [RFC3550]). When a compound RTCP packet that contains RTCP packets from several SSRCs is sent or received, the avg_rtcp_size estimate for each SSRC that is reported upon is updated using div_packet_size rather than the actual packet size:

```
avg_rtcp_size = (1/16) * div_packet_size + (15/16) * avg_rtcp_size
```

where div_packet_size is packet_size divided by the number of SSRCs reporting in that compound packet. The number of SSRCs reporting in a compound packet is determined by counting the number of different SSRCs that are the source of Sender Report (SR) or Receiver Report (RR) RTCP packets within the compound RTCP packet. Non-compound RTCP packets (i.e., RTCP packets that do not contain an SR or RR packet [RFC5506]) are considered report on a single SSRC.

An SSRC doesn’t follow the above rule, and instead uses the full RTCP compound packet size to calculate avg_rtcp_size, will derive an RTCP reporting interval that is overly large by a factor that is proportional to the number of SSRCs aggregated into compound RTCP packets and the size of set of SSRCs being aggregated relative to the total number of participants. This increased RTCP reporting interval can cause premature timeouts if it is more than five times the interval chosen by the SSRCs that understand compound RTCP that

aggregate reports from many SSRCs. A 1500 octet MTU can fit six typical size reports into a compound RTCP packet, so this is a real concern if endpoints aggregate RTCP reports from multiple SSRCs. If compatibility with non-updated endpoints is a concern, the number of reports from different SSRCs aggregated into a single compound RTCP packet SHOULD be limited.

5.3.2. Scheduling RTCP with Multiple Reporting SSRCs

When implementing RTCP packet scheduling for cases where multiple reporting SSRCs are aggregating their RTCP packets in the same compound packet there are a number of challenges. First of all, we have the goal of not changing the general properties of the RTCP packet transmissions, which include the general inter-packet distribution, and the behaviour for dealing with flash joins as well as other dynamic events.

The below specified mechanism deals with:

- That one can't have a-priori knowledge about which RTCP packets are to be sent, or their size, prior to generating the packets. In which case, the time from generation to transmission ought to be as short as possible to minimize the information that becomes stale.

- That one has an MTU limit, that one ought to avoid exceeding, as that requires lower-layer fragmentation (e.g., IP fragmentation) which impacts the packets' probability of reaching the receiver(s).

Schedule all the endpoint's local SSRCs individually for transmission using the regular calculation of Tn for the profile being used. Each time a SSRC's Tn timer expires, do the regular reconsideration. If the reconsideration indicates that an RTCP packet is to be sent:

1. Consider if an additional SSRC can be added. That consideration is done by picking the SSRC which has the Tn value closest in time to now (Tc).

2. Calculate how much space for RTCP packets would be needed to add that SSRC.

3. If the considered SSRC's RTCP Packets fit within the lower layer datagram's Maximum Transmission Unit, taking the necessary protocol headers into account and the consumed space by prior SSRCs, then add that SSRC's RTCP packets to the compound packet and go again to Step 1.

4. If the considered SSRC's RTCP Packets will not fit within the compound packet, then transmit the generated compound packet.

5. Update the RTCP Parameters for each SSRC that has been included in the sent RTCP packet. The Tp value for each SSRC MUST be updated as follows:

   For the first SSRC: As this SSRC was the one that was reconsidered the tp value is set to the tc as defined in RTP [RFC3550].

   For any additional SSRC: The tp value SHALL be set to the transmission time this SSRC would have had it not been aggregated and given the current existing session context. This value is derived by taking this SSRC's Tn value and performing reconsideration and updating tn until tp + T <= tn. Then set tp to this tn value.

6. For the sent SSRCs calculate new tn values based on the updated parameters and reschedule the timers.

Reverse reconsideration needs to be performed as specified in RTP [RFC3550]. It is important to note that under the above algorithm when performing reconsideration, the value of tp can actually be larger than tc. However, that still has the desired effect of proportionally pulling the tp value towards tc (as well as tn) as the group size shrinks in direct proportion the reduced group size.

The above algorithm has been shown in simulations to maintain the inter-RTCP-packet transmission distribution for the SSRCs and consume the same amount of bandwidth as non-aggregated packets in RTP sessions with static sets of participants. With this algorithm the actual transmission interval for any SSRC triggering an RTCP compound packet transmission is following the regular transmission rules. It also handles the cases where the number of SSRCs that can be included in an aggregated packet varies. An SSRC that previously was aggregated and fails to fit in a packet still has its own transmission scheduled according to normal rules. Thus, it will trigger a transmission in due time, or the SSRC will be included in another aggregate.

The algorithm's behaviour under SSRC group size changes is under investigation. However, it is expected to be well behaved based on the following analyses.

RTP sessions where the number of SSRC are growing: When the group size is growing, the Td values grow in proportion to the number of new SSRCs in the group. The reconsideration when the timer for

the tn expires, that SSRC will reconsider the transmission and with a certain probability reschedule the tn timer. This part of the reconsideration algorithm is only impacted by the above algorithm by having tp values that are in the future instead of set to the time of the actual last transmission at the time of updating tp. Thus the scheduling causes in worst case a plateau effect for that SSRC. That effect depends on how far into the future tp can advance.

RTP sessions where the number of SSRC are shrinking: When the group shrinks, reverse reconsideration moves the tp and tn values towards tc proportionally to the number of SSRCs that leave the session compared to the total number of participants when they left. Thus the also group size reductions need to be handled.

In general the potential issue that might exist depends on how far into the future the tp value can drift compared to the actual packet transmissions that occur. That drift can only occur for an SSRC that never is the trigger for RTCP packet transmission and always gets aggregated and where the calculated packet transmission interval randomly occurs so that tn - tp for this SSRC is on average larger than the ones that gets transmitted.

5.4. Use of RTP/AVPF Feedback

This section discusses the transmission of RTP/AVPF feedback packets when the transmitting endpoint has multiple SSRCs.

5.4.1. Choice of SSRC for Feedback Packets

When an RTP/AVPF endpoint has multiple SSRCs, it can choose what SSRC to use as the source for the RTCP feedback packets it sends. Several factors can affect that choice:

- RTCP feedback packets relating to a particular media type SHOULD be sent by an SSRC that receives that media type. For example, when audio and video are multiplexed onto a single RTP session, endpoints will use their audio SSRC to send feedback on the audio received from other participants.

- RTCP feedback packets and RTCP codec control messages that are notifications or indications regarding RTP data processed by an endpoint MUST be sent from the SSRC used by that RTP data. This includes notifications that relate to a previously received request or command.

- If separate SSRCs are used to send and receive media, then the corresponding SSRC SHOULD be used for feedback, since they have

differing RTCP bandwidth fractions. This can also effect the consideration if the SSRC can be used in immediate mode or not.

- Some RTCP feedback packet types requires consistency in the SSRC used. For example, if one sets a TMMBR limitation, the same SSRC needs to be used to remove the limitation.

When an RTCP feedback packet is sent as part of a compound RTCP packet that aggregates reports from multiple SSRCs, there is no requirement that the compound packet contains an SR or RR packet generated by the sender of the RTCP feedback packet. For reduced-size RTCP packets, aggregation of RTCP feedback packets from multiple sources is not limited further than Section 4.2.2 of [RFC5506].

5.4.2. Scheduling an RTCP Feedback Packet

When an SSRC has a need to transmit a feedback packet in early mode it follows the scheduling rules defined in Section 3.5 in RTP/AVPF [RFC4585]. When following these rules the following clarifications need to be taken into account:

- That a session is considered to be point-to-point or multiparty not based on the number of SSRCs, but the number of endpoints directly seen in the RTP session by the endpoint. TBD: Clarify what is considered to "see" an endpoint?

- Note that when checking if there is already a scheduled compound RTCP packet containing feedback messages (Step 2 in Section 3.5.2), that check is done considering all local SSRCs.

TBD: The above does not allow an SSRC that is unable to send either an early or regular RTCP packet with the feedback message within the T_max_fb_delay to trigger another SSRC to send an early packet to which it could piggyback. Nor does it allow feedback to piggyback on even regular RTCP packet transmissions that occur within T_max_fb_delay. A question is if either of these behaviours ought to be allowed. The latter appears simple and straight forward. Instead of discarding a FB message in step 4a: alternative 2, one could place such messages in a cache with a discard time equal to T_max_fb_delay, and in case any of the SSRCs schedule an RTCP packet for transmission within that time, it includes this message. The former case can have more widespread impact on the application, and possibly also on the RTCP bandwidth consumption as it allows for more massive bursts of RTCP packets. Still, on a time scale of a regular reporting interval, it ought to have no effect on the RTCP bandwidth as the extra feedback messages increase the avg_rtcp_size.

6. RTCP Considerations for Streams with Disparate Rates

An RTP session has a single set of parameters that configure the session bandwidth. These are the RTCP sender and receiver fractions (e.g., the SDP "b=RR:" and "b=RS:" lines), and the parameters of the RTP/AVPF profile [RFC4585] (e.g., trr-int) if that profile (or its secure extension, RTP/SAVPF [RFC5124]) is used. As a consequence, the base RTCP reporting interval, before randomisation, will be the same for every sending SSRC in an RTP session. Similarly, every receiving SSRC in an RTP session will have the same base reporting interval, although this can differ from the reporting interval chosen by sending SSRCs. This uniform RTCP reporting interval for all SSRCs can result in RTCP reports being sent more often than is considered desirable for a particular media type.

For example, consider a scenario when an audio flow sending at tens of kilobits per second is multiplexed into an RTP session with a multi-megabit high quality video flow. If the session bandwidth is configured based on the video sending rate, and the default RTCP bandwidth fraction of 5% of the session bandwidth is used, it is likely that the RTCP bandwidth will exceed the audio sending rate. If the reduced minimum RTCP interval described in Section 6.2 of [RFC3550] is then used in the session, as appropriate for video where rapid feedback on damaged I-frames is wanted, the uniform reporting interval for all senders could mean that audio sources are expected to send RTCP packets more often than they send audio data packets. This bandwidth mismatch can be reduced by careful tuning of the RTCP parameters, especially trr_int when the RTP/AVPF profile is used, cannot be avoided entirely, as it is inherent in the design of the RTCP timing rules, and affects all RTP sessions that contain flows with greatly mismatched bandwidth.

Sending multiple media types in a single RTP session causes that session to contain more SSRCs than if each media type was sent in a separate RTP session. For example, if two participants each send an audio and a video flow in a single RTP session, that session will comprise four SSRCs, but if separate RTP sessions had been used for audio and video, each of those two RTP sessions would comprise only two SSRCs. Sending multiple media streams in an RTP session hence increases the amount of cross reporting between the SSRCs, as each SSRC reports on all other SSRCs in the session. This increases the size of the RTCP reports, causing them to be sent less often than would be the case if separate RTP sessions where used for a given RTCP bandwidth.

Finally, when an RTP session contains multiple media types, it is important to note that the RTCP reception quality reports, feedback messages, and extended report blocks used might not be applicable to

all media types. Endpoints will need to consider the media type of each SSRC only send or process reports and feedback that apply to that particular SSRC and its media type. Signalling solutions might have shortcomings when it comes to indicating that a particular set of RTCP reports or feedback messages only apply to a particular media type within an RTP session.

From an RTCP perspective, therefore, it can be seen that there are advantages to using separate RTP sessions for each media stream, rather than sending multiple media streams in a single RTP session. However, these are frequently offset by the need to reduce port use, to ease NAT/firewall traversal, achieved by combining media streams into a single RTP session. The following sections consider some of the issues with using RTCP in sessions with multiple media streams in more detail.

6.1. Timing out SSRCs

Various issues have been identified with timing out SSRC values when sending multiple media streams in an RTP session.

6.1.1. Problems with RTP/AVPF the T_rr_interval Parameter

The RTP/AVPF profile includes a method to prevent RTCP reports from being sent too often. This mechanism is described in Section 3.5.3 of [RFC4585], and is controlled by the T_rr_interval parameter. It works as follows. When a regular RTCP report is sent, a new random value, T_rr_current_interval, is generated, drawn evenly in the range 0.5 to 1.5 times T_rr_interval. If a regular RTCP packet is to be sent earlier then T_rr_current_interval seconds after the previous regular RTCP packet, and there are no feedback messages to be sent, then that regular RTCP packet is suppressed, and the next regular RTCP packet is scheduled. The T_rr_current_interval is recalculated each time a regular RTCP packet is sent. The benefit of suppression is that it avoids wasting bandwidth when there is nothing requiring frequent RTCP transmissions, but still allows utilization of the configured bandwidth when feedback is needed.

Unfortunately this suppression mechanism skews the distribution of the RTCP sending intervals compared to the regular RTCP reporting intervals. The standard RTCP timing rules, including reconsideration and the compensation factor, result in the intervals between sending RTCP packets having a distribution that is skewed towards the upper end of the range [0.5/1.21828, 1.5/1.21828]*Td, where Td is the deterministic calculated RTCP reporting interval. With Td = 5s this distribution covers the range [2.052s, 6.156s]. In comparison, the RTP/AVPF suppression rules act in an interval that is 0.5 to 1.5 times T_rr_interval; for T_rr_interval = 5s this is [2.5s, 7.5s].

The effect of this is that the time between consecutive RTCP packets when using T_rr_interval suppression can become large. The maximum time interval between sending one regular RTCP packet and the next, when T_rr_interval is being used, occurs when T_rr_current_interval takes its maximum value and a regular RTCP packet is suppressed at the end of the suppression period, then the next regular RTCP packet is scheduled after its largest possible reporting interval. Taking the worst case of the two intervals gives a maximum time between two RTCP reports of 1.5*T_rr_interval + 1.5/1.21828*Td.

This behaviour can be surprising when Td and T_rr_interval have the same value. That is, when T_rr_interval is configured to match the regular RTCP reporting interval. In this case, one might expect that regular RTCP packets are sent according to their usual schedule, but feedback packets can be sent early. However, the above-mentioned issue results in the RTCP packets actually being sent in the range [0.5*Td, 2.731*Td] with a highly non-uniform distribution, rather than the range [0.41*Td, 1.23*Td]. This is perhaps unexpected, but is not a problem in itself. However, when coupled with packet loss, it raises the issue of premature timeout.

6.1.2. Avoiding Premature Timeout

In RTP/AVP [RFC3550] the timeout behaviour is simple, and is 5 times Td, where Td is calculated with a Tmin value of 5 seconds. In other words, if the configured RTCP bandwidth allows for an average RTCP reporting interval shorter than 5 seconds, the timeout is 25 seconds of no activity from the SSRC (RTP or RTCP), otherwise the timeout is 5 average reporting intervals.

RTP/AVPF [RFC4585] introduces different timeout behaviours depending on the value of T_rr_interval. When T_rr_interval is 0, it uses the same timeout calculation as RTP/AVP. However, when T_rr_interval is non-zero, it replaces Tmin in the timeout calculation, most likely to speed up detection of timed out SSRCs. However, using a non-zero T_rr_interval has two consequences for RTP behaviour.

First, due to suppression, the number of RTP and RTCP packets sent by an SSRC that is not an active RTP sender can become very low, because of the issue discussed in Section 6.1.1. As the RTCP packet interval can be as long as 2.73*Td, then during a 5*Td time period an endpoint might in fact transmit only a single RTCP packet. The long intervals result in fewer RTCP packets, to a point where a single RTCP packet loss can sometimes result in timing out an SSRC.

Second, the RTP/AVPF changes to the timeout rules reduce robustness to misconfiguration. It is common to use RTP/AVPF configured such that RTCP packets can be sent frequently, to allow rapid feedback,

however this makes timeouts very sensitive to T_rr_interval. For example, if two SSRCs are configured one with T_rr_interval = 0.1s and the other with T_rr_interval = 0.6s, then this small difference will result in the SSRC with the shorter T_rr_interval timing out the other if it stops sending RTP packets, since the other RTCP reporting interval is more than five times its own. When RTP/AVP is used, or RTP/AVPF with T_rr_interval = 0, this is a non-issue, as the timeout period will be 25s, and differences between configured RTCP bandwidth can only cause premature timeouts when the reporting intervals are greater than 5s and differ by a factor of five. To limit the scope for such problematic misconfiguration, we propose an update to the RTP/AVPF timeout rules in Section 6.1.4.

6.1.3. Interoperability Between RTP/AVP and RTP/AVPF

If endpoints implementing the RTP/AVP and RTP/AVPF profiles (or their secure variants) are combined within a single RTP session, and the RTP/AVPF endpoints use a non-zero T_rr_interval that is significantly below 5 seconds, there is a risk that the RTP/AVPF endpoints will prematurely timeout the SSRCs of the RTP/AVP endpoints, due to their different RTCP timeout rules. Conversely, if the RTP/AVPF endpoints use a T_rr_interval that is significant larger than 5 seconds, there is a risk that the RTP/AVP endpoints will timeout the SSRCs of the RTP/AVPF endpoints.

Mixing endpoints using two different RTP profiles within a single RTP session is NOT RECOMMENDED. However, if mixed RTP profiles are used, and the RTP/AVPF endpoints are not updated to follow Section 6.1.4 of this memo, then the RTP/AVPF session SHOULD be configured to use T_rr_interval = 4 seconds to avoid premature timeouts.

The choice of T_rr_interval = 4 seconds for interoperability might appear strange. Intuitively, this value ought to be 5 seconds, to make both the RTP/AVP and RTP/AVPF use the same timeout period. However, the behaviour outlined in Section 6.1.1 shows that actual RTP/AVPF reporting intervals can be longer than expected. Setting T_rr_interval = 4 seconds gives actual RTCP intervals near to those expected by RTP/AVP, ensuring interoperability.

6.1.4. Updated SSRC Timeout Rules

To ensure interoperability and avoid premature timeouts, all SSRCs in an RTP session MUST use the same timeout behaviour. However, previous specification are inconsistent in this regard. To avoid interoperability issues, this memo updates the timeout rules as follows:

- For the RTP/AVP, RTP/SAVP, RTP/AVPF, and RTP/SAVPF profiles, the timeout interval SHALL be calculated using a multiplier of five times the deterministic RTCP reporting interval. That is, the timeout interval SHALL be 5*Td.

- For the RTP/AVP, RTP/SAVP, RTP/AVPF, and RTP/SAVPF profiles, calculation of Td SHALL be done using a Tmin value of 5 seconds and not the reduced minimal interval, even if the reduced minimum interval is used to calculate RTCP packet transmission intervals.

This changes the behaviour for the RTP/AVPF or RTP/SAVPF profiles when T_rr_interval != 0, a behaviour defined in Section 3.5.4 of RFC 4585, i.e. Tmin in the Td calculation is the T_rr_interval.

6.2. Tuning RTCP transmissions

This sub-section discusses what tuning can be done to reduce the downsides of the shared RTCP packet intervals. First, it is considered what possibilities exist for the RTP/AVP [RFC3551] profile, then what additional tools are provided by RTP/AVPF [RFC4585].

6.2.1. RTP/AVP and RTP/SAVP

When using the RTP/AVP or RTP/SAVP profiles, the options for tuning the RTCP reporting intervals are limited to the RTCP sender and receiver bandwidth, and whether the minimum RTCP interval is scaled according to the bandwidth. As the scheduling algorithm includes both randomisation and reconsideration, one cannot simply calculate the expected average transmission interval using the formula for Td given in Section 6.3.1 of [RFC3550]. However, by considering the inputs to that expression, and the randomisation and reconsideration rules, we can begin to understand the behaviour of the RTCP transmission interval.

Let’s start with some basic observations:

a. Unless the scaled minimum RTCP interval is used, then Td prior to randomization and reconsideration can never be less than Tmin. The default value of Tmin is 5 seconds.

b. If the scaled minimum RTCP interval is used, Td can become as low as 360 divided by RTP Session bandwidth in kilobits per second. In SDP the RTP session bandwidth is signalled using a "b=AS" line. An RTP Session bandwidth of 72kbps results in Tmin being 5 seconds. An RTP session bandwidth of 360kbps of course gives a Tmin of 1 second, and to achieve a Tmin equal to once every frame for a 25 frame-per-second video stream requires an RTP session

bandwidth of 9Mbps. Use of the RTP/AVPF or RTP/SAVPF profile allows more frequent RTCP reports for the same bandwidth, as discussed below.

c. The value of Td scales with the number of SSRCs and the average size of the RTCP reports, to keep the overall RTCP bandwidth constant.

d. The actual transmission interval for a Td value is in the range [0.5*Td/1.21828,1.5*Td/1.21828], and the distribution is skewed, due to reconsideration, with the majority of the probability mass being above Td. This means, for example, that for Td = 5s, the actual transmission interval will be distributed in the range [2.052s, 6.156s], and tending towards the upper half of the interval. Note that Tmin parameter limits the value of Td before randomisation and reconsideration are applied, so the actual transmission interval will cover a range extending below Tmin.

Given the above, we can calculate the number of SSRCs, n, that an RTP session with 5% of the session bandwidth assigned to RTCP can support while maintaining Td equal to Tmin. This will tell us how many media streams we can report on, keeping the RTCP overhead within acceptable bounds. We make two assumptions that simplify the calculation: that all SSRCs are senders, and that they all send compound RTCP packets comprising an SR packet with n-1 report blocks, followed by an SDES packet containing a 16 octet CNAME value [RFC7022] (such RTCP packets will vary in size between 54 and 798 octets depending on n, up to the maximum of 31 report blocks that can be included in an SR packet). If we put this packet size, and a 5% RTCP bandwidth fraction into the RTCP interval calculation in Section 6.3.1 of [RFC3550], and calculate the value of n needed to give Td = Tmin for the scaled minimum interval, we find n=9 SSRCs can be supported (irrespective of the interval, due to the way the reporting interval scales with the session bandwidth). We see that to support more SSRCs, we need to increase the RTCP bandwidth fraction from 5%; changing the session bandwidth does not help due to the limit of Tmin.

To conclude, with RTP/AVP and RTP/SAVP the key limitation for small unicast sessions is going to be the Tmin value. Thus the RTP session bandwidth configured in RTCP has to be sufficiently high to reach the reporting goals the application has following the rules for the scaled minimal RTCP interval.

6.2.2. RTP/AVPF and RTP/SAVPF

When using RTP/AVPF or RTP/SAVPF, we have a powerful additional tool for tuning RTCP transmissions: the T_rr_interval parameter. Use of this parameter allows short RTCP reporting intervals; alternatively it gives the ability to sent frequent RTCP feedback without sending frequent regular RTCP reports.

The use of the RTP/AVPF or RTP/SAVPF profile with T_rr_interval set to a value greater than zero allows more frequent RTCP feedback than the RTP/AVP or RTP/SAVP profiles, for a given RTCP bandwidth. This happens because Tmin is set to zero after the transmission of the initial RTCP report, causing the reporting interval for later packet to be determined by the usual RTCP bandwidth-based calculation, with Tmin=0, and the T_rr_interval. This has the effect that we are no longer restricted by the minimal interval (whether the default 5 second minimum, or the reduced minimum interval). Rather, the RTCP bandwidth and the T_rr_interval are the governing factors, allowing faster feedback. Applications that care about rapid regular RTCP feedback ought to consider using the RTP/AVPF or RTP/SAVPF profile, even if they don't use the feedback features of that profile.

The use of the RTP/AVPF or RTP/SAVPF profile allows RTCP feedback packets to be sent frequently, without also requiring regular RTCP reports to be sent frequently, since T_rr_interval limits the rate at which regular RTCP packets can be sent, while still permitting RTCP feedback packets to be sent. Applications that can use feedback packets for some media streams, e.g., video streams, but don't want frequent regular reporting for other media streams, can configure the T_rr_interval to a value so that the regular reporting for both audio and video is at a level that is considered acceptable for the audio. They could then use feedback packets, which will include RTCP SR/RR packets unless reduced size RTCP feedback packets [RFC5506] are used, for the video reporting. This allows the available RTCP bandwidth to be devoted on the feedback that provides the most utility for the application.

Using T_rr_interval still requires one to determine suitable values for the RTCP bandwidth value. Indeed, it might make this choice even more important, as this is more likely to affect the RTCP behaviour and performance than when using the RTP/AVP or RTP/SAVP profile, as there are fewer limitations affecting the RTCP transmission.

When T_rr_interval is non-zero, there are configurations that need to be avoided. If the RTCP bandwidth chosen is such that the Td value is smaller than, but close to, T_rr_interval, then the actual regular RTCP packet transmission interval can become very large, as discussed in Section 6.1.1. Therefore, for configuration where one intends to

have Td smaller than T_rr_interval, then Td is RECOMMENDED to be targeted at values less than 1/4th of T_rr_interval which results in that the range becomes [0.5*T_rr_interval, 1.81*T_rr_interval].

With the RTP/AVPF or RTP/SAVPF profile, using T_rr_interval = 0 with another low value significantly lower than Td still has utility, and different behaviour compared to the RTP/AVP profile. This avoids the Tmin limitations of RTP/AVP, thus allowing more frequent regular RTCP reporting. In fact this will result that the RTCP traffic becomes as high as the configured values.

(TBD: a future version of this memo will include examples of how to choose RTCP parameters for common scenarios)

There exists no method for using different regular RTCP reporting intervals depending on the media type or individual media stream, other than using a separate RTP session for the other stream.

7. Security Considerations

When using the secure RTP protocol (RTP/SAVP) [RFC3711], or the secure variant of the feedback profile (RTP/SAVPF) [RFC5124], the cryptographic context of a compound secure RTCP packet is the SSRC of the sender of the first RTCP (sub-)packet. This could matter in some cases, especially for keying mechanisms such as Mikey [RFC3830] which allow use of per-SSRC keying.

Otherwise, the standard security considerations of RTP apply; sending multiple media streams from a single endpoint in a single RTP session does not appear to have different security consequences than sending the same number of media streams spread across different RTP sessions.

8. IANA Considerations

No IANA actions are required.

9. Open Issues

At this stage this document contains a number of open issues. The below list tries to summarize the issues:

1. Do we need to provide a recommendation for unicast session joiners with many sources to not use 0 initial minimal interval from bit-rate burst perspective?

2. RTCP parameters for common scenarios in Section 6.2?

3. Is scheduling algorithm working well with dynamic changes?

4. Are the scheduling algorithm changes impacting previous implementations in such a way that the report aggregation has to be agreed on, and thus needs to be considered as an optimization?

5. An open question is if any improvements or clarifications ought to be allowed regarding FB message scheduling in multi-SSRC endpoints.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

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

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

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, February 2008.

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.

10.2. Informative References

[I-D.ietf-avtcore-multi-media-rtp-session] Westerlund, M., Perkins, C., and J. Lennox, "Sending Multiple Types of Media in a Single RTP Session", draft-ietf-avtcore-multi-media-rtp-session-06 (work in progress), October 2014.

[I-D.ietf-avtcore-rtp-multi-stream-optimisation]

Lennox, J., Westerlund, M., Wu, Q., and C. Perkins, "Sending Multiple Media Streams in a Single RTP Session: Grouping RTCP Reception Statistics and Other Feedback ", draft-ietf-avtcore-rtp-multi-stream-optimisation-00 (work in progress), July 2013.

[I-D.ietf-avtcore-rtp-topologies-update] Westerlund, M. and S. Wenger, "RTP Topologies", draft-ietf-avtcore-rtp-topologies-update-04 (work in progress), August 2014.

[I-D.ietf-clue-framework] Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-18 (work in progress), October 2014.

[I-D.ietf-mmusic-sdp-bundle-negotiation] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-12 (work in progress), October 2014.

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

[RFC3611] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004.

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

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

[RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla, "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)", RFC 7022, September 2013.

Authors' Addresses

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

Email: jonathan@vidyo.com

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

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

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: sunseawq@huawei.com

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

Email: csp@csperkins.org