AVTCORE
Internet-Draft
Updates: 3550, 4585 (if approved)
Intended status: Standards Track
Expires: January 04, 2015

J. Lennox
Vidyo
M. Westerlund
Ericsson
Q. Wu
Huawei
C. Perkins
University of Glasgow
July 03, 2014

Sending Multiple Media Streams in a Single RTP Session
draft-ietf-avtcore-rtp-multi-stream-05

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.

Table of Contents

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 they maintain the average RTCP packet size as described in Section 5.3.1, and schedule packet transmission and 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:

- o 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.

- o 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.

- o 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

It is possible for a single RTP session to carry streams of greatly differing bandwidth. There are two scenarios where this can occur. The first is when a single RTP session carries multiple flows of the same media type, but with very different quality; for example a video switching multi-point conference unit might send a full rate high-definition video stream of the active speaker but only thumbnails for the other participants, all sent in a single RTP session. The second scenarios occurs when audio and video flows are sent in a single RTP session, as discussed in [I-D.ietf-avtcore-multi-media-rtp-session].

An RTP session has a single set of parameters that configure the session bandwidth, the RTCP sender and receiver fractions (e.g., via 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 RTCP reporting interval will be the same for every SSRC in an RTP session. This uniform RTCP reporting interval can result in RTCP reports being sent more often than is considered desirable for a particular media type. For example, if an audio flow is multiplexed with a high quality video flow where the session bandwidth is configured to match the video bandwidth, this can result in the RTCP packets having a greater bandwidth allocation than the audio data rate. If the reduced minimum RTCP interval described in Section 6.2 of [RFC3550] is used in the session, which might be appropriate for video where rapid feedback is wanted, the audio sources could be expected to send RTCP packets more often than they send audio data packets. This is most likely undesirable, and while the mismatch can be reduced through careful tuning of the RTCP parameters, particularly trr_int in RTP/AVPF sessions, it is inherent in the design of the RTCP timing rules, and affects all RTP sessions containing flows with mismatched bandwidth.

Having multiple media types in one RTP session also results in more SSRCs being present in this RTP session. This increasing the amount of cross reporting between the SSRCs. From an RTCP perspective, two RTP sessions with half the number of SSRCs in each will be slightly more efficient. If someone needs either the higher efficiency due to the lesser number of SSRCs or the fact that one can’t tailor RTCP usage per media type, they need to use independent RTP sessions.

When it comes to configuring RTCP the need for regular periodic reporting needs to be weighted against any feedback or control messages being sent. Applications using RTP/AVPF or RTP/SAVPF are RECOMMENDED to consider setting the trr-int parameter to a value suitable for the application's needs, thus potentially reducing the need for regular reporting and thus releasing more bandwidth for use for feedback or control.

Another aspect of an RTP session with multiple media types is that the RTCP packets, RTCP Feedback Messages, or RTCP XR metrics used might not be applicable to all media types. Instead, all RTP/RTCP endpoints need to correlate the media type of the SSRC being referenced in a message or packet and only use those 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.

6.1. Timing out SSRCs

This section discusses issues around timing out SSRCs. After the discussion, clarified and mandated behaviour for SSRC timeout is specified.

6.1.1. AVPF T_rr_interval Behaviour

The RTP/AVPF profile includes a mechanism for suppressing regular RTCP reporting from being sent unnecessarily frequently if sufficient RTCP bandwidth is configured. This mechanism is defined in Section 3.5.3 of [RFC4585], and can be summarized as follows: if less than a randomized T_rr_interval value has passed since the last regular report, and no feedback messages need to be sent, then the RTCP regular report is suppressed. The randomization is done linearly in the interval 0.5 to 1.5 times T_rr_interval. The randomized T_rr_interval is recalculated after every transmitted regular packet, i.e when t_rr_last was updated. The benefit of the suppression mechanism 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 has some behaviour that is less than ideal. First of all, the randomized T_rr_interval is distributed over a larger range than the actual transmission interval for RTCP would be if T_rr_interval and Td had the same value. The reconsideration mechanism and its compensation factor result in the actual RTCP transmission intervals for a Td having a distribution that is exponentially growing more likely with higher values, and is bounded to the interval [0.5/1.21828, 1.5/1.21828]*Td, i.e. with a

Td value of 5 s [2.052, 6.156]. In comparison, the suppression acts in an interval that is 0.5 to 1.5 times the T_rr_interval, i.e. for T_rr_interval = 5 s this is [2.5, 7.5].

The effect of the above is that the time period between two RTCP packets when using T_rr_interval suppression can become very long compared to the average input values. The longest time interval between one transmitted regular RTCP compound packet and the next when T_rr_interval suppression is being used are: First the maximum T_rr_interval, i.e. 1.5*T_rr_interval. Assuming that the last suppressed packet would have been sent at 1.5*T_rr_interval, the maximum interval until a packet can be sent under the regular scheduling is 1.5/1.21828*Td. Thus, the maximum time in total is 1.5*T_rr_interval + 1.5/1.21828*Td.

If Td and T_rr_interval have the same value, i.e. the minimal interval desired (T_rr_interval) and the actual actual average interval specified by the RTCP scheduling algorithm (Td) are the same, one might expect that RTCP packets would be sent according to the regular mechanism. Instead, this algorithm results in the RTCP packets being sent anywhere from 0.5*Td to ~2.731*Td. The probability distribution over that time is also non-trivial in its shape, somewhat similar to a saw tooth.

Thus, we recommend that the AVPF regular transmission mechanism is revised in the future. This issue also has further implications as discussed in the next section.

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 RTCP bandwidth allowed for an RTCP interval more frequent than every 5 seconds on average, then timeout happened after 5*Td = 25 seconds of no activity from the SSRC (RTP or RTCP), otherwise it was 5 average reporting intervals.

RTP/AVPF [RFC4585] introduced two different behaviours depending on the value of T_rr_interval. When T_rr_interval was 0, it defaulted to the same Td calculation in RTP/AVP [RFC3550]. However, when T_rr_interval is non-zero the Tmin value become T_rr_interval in that calculation, most likely to enable speed up the detection of timed out SSRCs. However, using a non-zero T_rr_interval has two consequences for RTP behaviour.

First, the number of actually sent RTCP packets for an SSRC that currently is not an active RTP sender can become very low due to the issue discussed above 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 may in fact transmit only a single RTCP packet. The long intervals result in fewer RTCP packets, to a point where a one or two packet losses in RTCP result in timing out an SSRC.

Second, the change also increased RTP/AVPF's brittleness to both packet loss and configuration errors. In many cases, when one desires to use RTP/AVPF for its feedback, one will ensure that RTCP is configured for more frequent transmissions on average than every 5 seconds. Thus, many more RTP and RTCP packets can be transmitted during the time interval. Lets consider an implementation that would follow the RTP/AVP or RTP/AVPF with T_rr_interval = 0 rules for timeout, also when T_rr_interval is not zero. In such a case when the configured value of the T_rr_interval is significantly smaller than 5 seconds, e.g. less than 1 second, then a difference between using 0.1 seconds and 0.6 seconds has no significant impact on when an SSRC will be timed out. However, such a configuration difference between two endpoints following RFC 4585 will result in that the endpoint configured with T_rr_interval = 0.1 will frequently timeout SSRCs currently not sending RTP, from the endpoint configured with 0.6, as that is six times the Td value used by the endpoint configured with T_rr_interval=0.1, assuming sufficient bandwidth. For this reason such a change is implemented below in Section 6.1.4.

6.1.3. RTP/AVP and RTP/AVPF Interoperability

If endpoints implementing the RTP/AVP and RTP/AVPF profiles (or their secure variants) are combined in a single RTP session, and the RTP/AVPF endpoints use a non-zero T_rr_interval that is significantly lower than 5 seconds, then there is a risk that the RTP/AVPF endpoints will prematurely timeout the RTP/AVP SSRCs due to their different RTCP timeout intervals. 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 RTP/AVPF SSRCs.

If such mixed RTP profiles are used, (though this is NOT RECOMMENDED), and the AVPF endpoint is not updated to follow this specification, then the RTP/AVPF session SHOULD use a non-zero T_rr_interval that is 4 seconds.

It might appear strange to use a T_rr_interval of 4 seconds. It might be intuitive that this value ought to be 5 seconds, as then both the RTP/AVP and RTP/AVPF would use the same timeout period. However, considering regular RTCP transmission and their packet intervals for RTP/AVPF its mean value will (with non-zero T_rr_interval) be larger than T_rr_interval due to the scheduling algorithm's behaviour as discussed in Section 6.1.1. Thus, to enable an equal amount of regular RTCP transmissions in each directions between RTP/AVP and RTP/AVPF endpoints, taking the altered timeout intervals into account, the optimal value is around four (4), where almost four transmissions will on average occur in each direction between the different profile types given an otherwise good configuration of parameters in regards to T_rr_interval. If the RTCP bandwidth parameters are selected so that Td based on bandwidth is close to 4, i.e. close to T_rr_interval the risk increases that RTP/AVPF SSRCs will be timed out by RTP/AVP endpoints, as the RTP/AVPF SSRC might only manage two transmissions in the timeout period.

6.1.4. Specified Behaviour

The above considerations result in the following clarification and RTP/AVPF specification change.

All SSRCs used in an RTP session MUST use the same timeout behaviour to avoid premature timeouts. This will depend on the RTP profile and its configuration. The RTP specification provides several options that can influence the values used when calculating the time interval. To avoid interoperability issues when using this specification, this document makes several clarifications to the calculations.

For RTP/AVP, RTP/SAVP, RTP/AVPF, and RTP/SAVPF, the timeout interval SHALL be calculated using a multiplier of 5, i.e. the timeout interval becomes 5*Td. The Td calculation SHALL be done using a Tmin value of 5 seconds, not the reduced minimal interval even if 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 tuning one can do is very limited. The controls one has are limited to the RTCP bandwidth values and whether the minimum RTCP interval is scaled according to the bandwidth. As the scheduling algorithm includes both random factors and reconsideration, one can’t simply calculate the expected average transmission interval using the formula for Td. But it does indicate the important factors affecting the transmission interval, namely the RTCP bandwidth available for the role (Active Sender or Participant), the average RTCP packet size, and the number of SSRCs classified in the relevant role. Note that if the ratio of senders to total number of session participants is larger than the ratio of RTCP bandwidth for senders in relation to the total RTCP bandwidth, then senders and receivers are treated together.

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 5 seconds (assuming default Tmin of 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. In SDP the RTP session bandwidth is signalled using b=AS. An RTP Session bandwidth of 72 kbps results in Tmin being 5 seconds. An RTP session bandwidth of 360 kbps of course gives a Tmin of 1 second, and to achieve a Tmin equal to once every frame for a 25 Hz video stream requires an RTP session bandwidth of 9 Mbps! (The use of the RTP/AVPF or RTP/SAVPF profile allows a smaller Tmin, and hence more frequent RTCP reports, as discussed below).

c. Let's calculate the number (n) of SSRCs in the RTP session that 5% of the session bandwidth can support to yield a Td value equal to Tmin with minimal scaling. For this calculation we have to make two assumptions. The first is that we will consider most or all SSRC being senders, resulting in everyone sharing the available bandwidth. Secondly we will select an average RTCP packet size. This packet will consist of an SR, containing (n-1) report blocks up to 31 report blocks, and an SDES item with at least a CNAME (17 bytes in size) in it. Such a basic packet will be 800 bytes for n>=32. With these parameters, and as the bandwidth goes up the time interval is proportionally decreased (due to minimal scaling), thus all the example bandwidths 72 kbps, 360 kbps and 9 Mbps all support 9 SSRCs.

d. The actual transmission interval for a Td value is [0.5*Td/1.21828,1.5*Td/1.21828], which means that for Td = 5 seconds, the interval is actually [2.052,6.156] and the distribution is not uniform, but rather exponentially-increasing. The probability for sending at time X, given it is within the interval, is probability of picking X in the interval times the probability to randomly picking a number that is <=X within the interval with an uniform probability distribution. This results in that the majority of the probability mass is above the Td value.

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 get a quite powerful additional tool, the setting of the T_rr_interval which has several effects on the RTCP reporting. First of all as Tmin is set to 0 after the initial transmission, the regular reporting interval is instead determined by the regular bandwidth based calculation and the T_rr_interval. This has the effect that we are no longer restricted by the minimal interval or even the scaling rule for the minimal rule. Instead the RTCP bandwidth and the T_rr_interval are the governing factors.

Now it also becomes important to separate between the application’s need for regular reports and RTCP feedback packet types. In both regular RTCP mode, as in Early RTCP Mode, the usage of the T_rr_interval prevents regular RTCP packets, i.e. packets without any Feedback packets, to be sent more often than T_rr_interval. This value is applied to prevent any regular RTCP packet to be sent less than T_rr_interval times a uniformly distributed random value from the interval [0.5,1.5] after the previous regular packet packet. The random value recalculated after each regular RTCP packet transmission.

So applications that have a use for feedback packets for some media streams, for example video streams, but don’t want frequent regular reporting for audio, could 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, and can include other report information in addition to the feedback packet that needs to be sent. That way the available RTCP bandwidth can be focused for the use which provides the most utility for the application.

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

When using T_rr_interval, i.e. having it be non zero, there are configurations that have to be avoided. If the resulting Td value is smaller but close to T_rr_interval then the interval in which the actual regular RTCP packet transmission falls into becomes very large, from 0.5 times T_rr_interval up to 2.73 times the T_rr_interval. 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 RTP/AVPF, using a T_rr_interval of 0 or with another low value significantly lower than Td still has utility, and different behaviour compared to RTP/AVP. 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 within the specification for using different regular RTCP reporting intervals depending on the media type or individual media stream.

7. Security Considerations

In the secure RTP protocol (SRTP) [RFC3711], the cryptographic context of a compound SRTCP 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.

Other than that, the standard security considerations of RTP apply; sending multiple media streams from a single endpoint does not appear to have different security consequences than sending the same number of streams.

8. IANA Considerations

No IANA actions needed.

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-05 (work in progress), February 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-02 (work in progress), May 2014.

[I-D.ietf-clue-framework]
Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-16 (work in progress), June 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-07 (work in progress), April 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.

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