            Sending Multiple Media Streams in a Single RTP Session
                    draft-ietf-avtcore-rtp-multi-stream-10

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 May 27, 2016.

Copyright Notice

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 specification
   is thus somewhat unclear in places.

   This memo updates [RFC3550] to clarify behaviour in use cases where
   endpoints use multiple SSRCs. It also updates [RFC4585] to resolve
   problems with regards to timeout of inactive SSRCs, and to clarify
   behaviour around 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 single RTP session is when an endpoint
   has multiple capture devices, and hence can generate multiple media
   sources, of the same media type and characteristics. For example,
   telepresence systems 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 of each type within a single RTP session.

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 generate 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 multiple SSRCs can be used to send
data from a single media source within a single RTP session. These
include, but are not limited to, transport robustness tools, such as
the RTP retransmission payload format [RFC4588], that require one
SSRC to be used for the media data and another SSRC for the repair
data. Similarly, some layered media encoding schemes, for example
H.264 SVC [RFC6190], can be used in a configuration where each layer
is sent using a different SSRC within a single RTP session.

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

   RTP is inherently a group communication protocol. Each endpoint in
   an RTP session will use one or more SSRCs, as will some types of RTP
   level middlebox. Accordingly, unless restrictions on the number of
   SSRCs have been signalled, RTP endpoints can expect to receive RTP
   data packets sent using a number of different SSRCs, within a single
   RTP session. This can occur irrespective of whether the RTP session
   is running over a point-to-point connection or a multicast group,
   since middleboxes can be used to connect multiple transport
   connections together into a single RTP session (the RTP session is
   defined by the shared SSRC space, not by the transport connections).
   Furthermore, if RTP mixers are used, some SSRCs might only be visible
   in the contributing source (CSRC) list of an RTP packet and in RTCP,
   and might not appear directly as the SSRC of an RTP data packet.

   Every RTP endpoint will have an allocated share of the available
   session bandwidth, as determined by signalling and congestion
   control. The endpoint needs to 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 bandwidth
   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, or 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 have multiple SSRC values,
   implementations MUST treat each SSRC as a separate participant in the
   RTP session, so that an endpoint that has multiple SSRCs counts as
   multiple participants.

5.1. RTCP Reporting Requirement

   An RTP endpoint that has multiple SSRCs MUST treat each SSRC as a
   separate participant in the RTP session. Each SSRC will maintain its
   own RTCP-related state information, and hence will have its own RTCP
   reporting interval that determines when it sends RTCP reports. 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] is relevant: "For unicast sessions... the
delay before sending the initial compound RTCP packet MAY be zero."
The basic assumption is that this also ought to apply in the case of
multiple SSRCs. 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.

To ensure that the initial burst of traffic generated by an RTP
endpoint is no larger than would be generated by a TCP connection, an
RTP endpoint MUST NOT send more than four compound RTCP packets with
zero initial delay when it joins an RTP session, independently of the
number of SSRCs used by the endpoint. Each of those initial compound
RTCP packets MAY include aggregated reports from multiple SSRCs,
provided the total compound RTCP packet size does not exceed the MTU,
and the avg_rtcp_size is maintained as in Section 5.3.1. Aggregating
reports from several SSRCs in the initial compound RTCP packets
allows a substantial number of SSRCs to report immediately.
Endpoints SHOULD prioritize reports on SSRCs that are likely to be
most immediately useful, e.g., for SSRCs that are initially senders.

An endpoint that needs to report on more SSRCs than will fit into the
four compound RTCP reports that can be sent immediately MUST send the
other reports later, following the usual RTCP timing rules including
timer reconsideration. Those reports MAY be aggregated as described
in Section 5.3.

   Note: The above is chosen to match the TCP initial window of 4
   packets, not the larger TCP initial windows for which there is an
   ongoing experiment. The reason for this is a desire to be
   conservative, since an RTP endpoint will also in many cases start
   sending RTP data packets at the same time as these initial RTCP
   packets are sent.

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."

   This 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 can 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 to report on a single SSRC.

An SSRC that 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 five typical size reports into a compound RTCP packet, so this is a real concern if endpoints aggregate RTCP reports from multiple SSRCs.

The issue raised in the previous paragraph is mitigated by the modification in timeout behaviour specified in Section 7.1.2 of this memo. This mitigation is in place in those cases where the RTCP bandwidth is sufficiently high that an endpoint, using avg_rtcp_size calculated without taking into account the number of reporting SSRCs, can transmit more frequently than approximately every 5 seconds. Note, however, that the non-updated endpoint's RTCP reporting is still negatively impacted even if the premature timeout of its SSRCs are avoided. If compatibility with non-updated endpoints is a concern, the number of reports from different SSRCs aggregated into a single compound RTCP packet SHOULD either be limited to two reports, or aggregation ought not used at all. This will limit the non-updated endpoint's RTCP reporting interval to be no larger than twice the RTCP reporting interval that would be chosen by an endpoint following this specification.

5.3.2.   Scheduling RTCP when Aggregating Multiple SSRCs

   This section revises and extends the behaviour defined in Section 6.3
   of [RFC3550], and in Section 3.5.3 of [RFC4585] if the RTP/AVPF
   profile or the RTP/SAVPF profile is used, regarding actions to take
   when scheduling and sending RTCP packets where multiple reporting
   SSRCs are aggregating their RTCP packets into the same compound RTCP
   packet. These changes to the RTCP scheduling rules are needed to
   maintain important RTCP timing properties, including the inter-packet
   distribution, and the behaviour during flash joins and other changes
   in session membership.

   The variables tn, tp, tc, T, and Td used in the following are defined
   in Section 6.3 of [RFC3550]. The variables T_rr_interval and
   T_rr_last are defined in [RFC4585].

   Each endpoint MUST schedule RTCP transmission independently for each
   of its SSRCs using the regular calculation of tn for the RTP profile
   being used. Each time the timer tn expires for an SSRC, the endpoint
   MUST perform RTCP timer reconsideration and, if applicable,
   T_rr_interval based suppression. If the result indicates that a
   compound RTCP packet is to be sent by that SSRC, and the transmission
   is not an early RTCP packet [RFC4585], then the endpoint SHOULD try
   to aggregate RTCP packets of additional SSRCs that are scheduled in
   the future into the compound RTCP packet before it is sent. The
   reason to limit or not aggregate at due to backwards compatibility
   reasons was discussed earlier in Section 5.3.1.

   Aggregation proceeds as follows. The endpoint selects the SSRC that
   has the smallest tn value after the current time, tc, and prepares
   the RTCP packets that SSRC would send if its timer tn expired at tc.
   If those RTCP packets will fit into the compound RTCP packet that is
   being generated, taking into account the path MTU and the previously
   added RTCP packets, then they are added to the compound RTCP packet;
   otherwise they are discarded. This process is repeated for each
   SSRC, in order of increasing tn, until the compound RTCP packet is
   full, or all SSRCs have been aggregated. At that point, the compound
   RTCP packet is sent.

   When the compound RTCP packet is sent, the endpoint MUST update tp,
   tn, and T_rr_last (if applicable) for each SSRC that was included.
   These variables are updated as follows:

   a. For the first SSRC that reported in the compound RTCP packet, set
       the effective transmission time, tt, of that SSRC to tc.

   b. For each additional SSRC that reported in the compound RTCP
       packet, calculate the transmission time that SSRC would have had

if it had not been aggregated into the compound RTCP packet. This is derived by taking tn for that SSRC, then performing reconsideration and updating tn until tp + T <= tn. Once this is done, set the effective transmission time, tt, for that SSRC to the calculated value of tn. If the RTP/AVPF profile or the RTP/SAVPF profile is being used, then T_rr_interval based suppression MUST NOT be used in this calculation.

c. Calculate average effective transmission time, tt_avg, for the compound RTCP packet based on the tt values for all SSRCs sent in the compound RTCP packet. Set tp for each of the SSRCs sent in the compound RTCP packet to tt_avg. If the RTP/AVPF profile or the RTP/SAVPF profile is being used, set T_tt_last for each SSRC sent in the compound RTCP packet to tt_avg.

d. For each of the SSRCs sent in the compound RTCP packet, calculate new tn values based on the updated parameters and the usual RTCP timing rules, and reschedule the timers.

When using the RTP/AVPF profile or the RTP/SAVPF profile, the above mechanism only attempts to aggregate RTCP packets when the compound RTCP packet to be sent is not an early RTCP packet, and hence the algorithm in Section 3.5.3 of [RFC4585] will control RTCP scheduling. If T_rr_interval == 0, or if T_rr_interval != 0 and option 1, 2a, or 2b of the algorithm are chosen, then the above mechanism updates the necessary variables. However, if the transmission is suppressed per option 2c of the algorithm, then tp is updated to tc as aggregation has not taken place.

Reverse reconsideration MUST be performed following Section 6.3.4 of [RFC3550]. In some cases, this can lead to the value of tp after reverse reconsideration being larger than tc. This is not a problem, and has the desired effect of proportionally pulling the tp value towards tc (as well as tn) as the reporting interval shrinks in direct proportion the reduced group size.

The above algorithm has been shown in simulations to maintain the inter-RTCP packet transmission time distribution for each SSRC, and to consume the same amount of bandwidth as non-aggregated RTCP packets. With this algorithm the actual transmission interval for an SSRC triggering an RTCP compound packet transmission is following the regular transmission rules. The value tp is set to somewhere in the interval [0,1.5/1.21828*Td] ahead of tc. The actual value is average of one instance of tc and the randomized transmission times of the additional SSRCs, thus the lower range of the interval is more probable. This compensates for the bias that is otherwise introduced by picking the shortest tn value out of the N SSRCs included in aggregate.

The algorithm 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 as follows:

RTP sessions where the number of SSRC are growing: When the group
   size is growing, Td grows in proportion to the number of new SSRCs
   in the group. When reconsideration is performed due to expiry of
   the tn timer, 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 were in the future instead of set to the
   time of the actual last transmission at the time of updating tp.

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. The setting of the tp value forward in time related to the
   tc could be believed to have negative effect. However, the reason
   for this setting is to compensate for bias caused by picking the
   shortest tn out of the N aggregated. This bias remains over a
   reduction in the number of SSRCs. The reverse reconsideration
   compensates the reduction independently of aggregation being used
   or not. The negative effect that can occur on removing an SSRC is
   that the most favourable tn belonged to the removed SSRC. The
   impact of this is limited to delaying the transmission, in the
   worst case, one reporting interval.

In conclusion the investigations performed have found no significant
negative impact on the scheduling algorithm.

5.4. Use of RTP/AVPF or RTP/SAVPF Feedback

This section discusses the transmission of RTP/AVPF feedback packets
when the transmitting endpoint has multiple SSRCs. The guidelines in
this section also apply to endpoints using the RTP/SAVPF profile.

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 for that RTP data. This
   includes notifications that relate to a previously received
   request or command [RFC4585][RFC5104].

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 affect the
   consideration if the SSRC can be used in immediate mode or not.

o Some RTCP feedback packet types require consistency in the SSRC
   used. For example, if a TMMBR limitation [RFC5104] is set by an
   SSRC, the same SSRC needs to be used to remove the limitation.

o If several SSRCs are suitable for sending feedback, it might be
   desirable to use an SSRC that allows the sending of feedback as an
   early RTCP packet.

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 MUST schedule that packet following the algorithm in Section 3.5
of [RFC4585] modified as follows:

o To determine whether an RTP session is considered to be a point-
   to-point session or a multiparty session, an endpoint MUST count
   the number of distinct RTCP SDES CNAME values used by the SSRCs
   listed in the SSRC field of RTP data packets it receives and in
   the "SSRC of sender" field of RTCP SR, RR, RTPFB, or PSFB packets
   it receives. An RTP session is considered to be a multiparty
   session if more than one CNAME is used by those SSRCs, unless
   signalling indicates that the session is to be handled as point to
   point, or RTCP reporting groups
   [I-D.ietf-avtcore-rtp-multi-stream-optimisation] are used. If

RTCP reporting groups are used, an RTP session is considered to be
a point-to-point session if the endpoint receives only a single
reporting group, and considered to be a multiparty session if
multiple reporting groups are received, or if a combination of
reporting groups and SSRCs that are not part of a reporting group
are received. Endpoints MUST NOT determine whether an RTP session
is multiparty or point-to-point based on the type of connection
(unicast or multicast) used, or on the number of SSRCs received.

o When checking if there is already a scheduled compound RTCP packet
   containing feedback messages (Step 2 in Section 3.5.2 of
   [RFC4585]), that check MUST be done considering all local SSRCs.

o If an SSRC is not allowed to send an early RTCP packet, then the
   feedback message MAY be queued for transmission as part of any
   early or regular scheduled transmission that can occur within the
   maximum useful lifetime of the feedback message (T_max_fb_delay).
   This modifies the behaviour in bullet 4a) in Section 3.5.2 of
   [RFC4585].

The first bullet point above specifies a rule to determine if an RTP
session is to be considered a point-to-point session or a multiparty
session. This rule is straightforward to implement, but is known to
incorrectly classify some sessions as multiparty sessions. The known
problems are as follows:

Endpoint with multiple synchronization contexts: An endpoint that is
   part of a point-to-point session can have multiple synchronization
   contexts, for example due to forwarding an external media source
   into a interactive real-time conversation. In this case the
   classification will consider the peer as two endpoints, while the
   actual RTP/RTCP transmission will be under the control of one
   endpoint.

Selective Forwarding Middlebox: The SFM as defined in Section 3.7 of
   [I-D.ietf-avtcore-rtp-topologies-update] has control over the
   transmission and configurations between itself and each peer
   endpoint individually. It also fully controls the RTCP packets
   being forwarded between the individual legs. Thus, this type of
   middlebox can be compared to the RTP mixer, which uses its own
   SSRCs to mix or select the media it forwards, that will be
   classified as a point-to-point RTP session by the above rule.

In the above cases it is very reasonable to use RTCP reporting groups
[I-D.ietf-avtcore-rtp-multi-stream-optimisation]. If that extension
is used, an endpoint can indicate that the multitude of CNAMEs are in
fact under a single endpoint or middlebox control by using only a
single reporting group.

The above rules will also classify some sessions where the endpoint
is connected to an RTP mixer as being point to point. For example
the mixer could act as gateway to an Any Source Multicast based RTP
session for the discussed endpoint. However, this will in most cases
be okay, as the RTP mixer provides separation between the two parts
of the session. The responsibility falls on the mixer to act
accordingly in each domain.

Finally, we note that signalling mechanisms could be defined to
override the rules when it would result in the wrong classification.

6. Adding and Removing SSRCs

The set of SSRCs present in a single RTP session can vary over time
due to changes in the number of endpoints in the session, or due to
changes in the number or type of media streams being sent.

Every endpoint in an RTP session will have at least one SSRC that it
uses for RTCP reporting, and for sending media if desired. It can
also have additional SSRCs, for sending extra media streams or for
additional RTCP reporting. If the set of media streams being sent
changes, then the set of SSRCs being sent will change. Changes in
the media format or clock rate might also require changes in the set
of SSRCs used. An endpoint can also have more active SSRCs than it
has active RTP media streams, and send RTCP relating to SSRCs that
are not currently sending RTP data packets so that its peers are
aware of the SSRCs, and have the associated context (e.g., clock
synchronisation and an SDES CNAME) in place to be able to play out
media as soon as they becomes active.

In the following, we describe some considerations around adding and
removing RTP streams and their associated SSRCs.

6.1. Adding RTP Streams

When an endpoint joins an RTP session it can have zero, one, or more
RTP streams it will send, or that it is prepared to send. If it has
no RTP stream it plans to send, it still needs an SSRC that will be
used to send RTCP feedback. If it will send one or more RTP streams,
it will need the corresponding number of SSRC values. The SSRCs used
by an endpoint are made known to other endpoints in the RTP session
by sending RTP and RTCP packets. SSRCs can also be signalled using
non-RTP means (e.g., [RFC5576]). Unless restricted by signalling, an
endpoint can, at any time, send an additional RTP stream, identified
by a new SSRC (this might be associated with a signalling event, but
that is outside the scope of this memo). This makes the new SSRC
visible to the other endpoints in the session, since they share the
single SSRC space inherent in the definition of an RTP session.

An endpoint that has never sent an RTP stream will have an SSRC that
it uses for RTCP reporting. If that endpoint wants to start sending
an RTP stream, it is RECOMMENDED that it use its existing SSRC for
that stream, since otherwise the participant count in the RTP session
will be unnecessary increased, leading to a longer RTCP reporting
interval and larger RTCP reports due to cross reporting. If the
endpoint wants to start sending more than one RTP stream, it will
need to generate a new SSRC for the second and any subsequent RTP
streams.

An endpoint that has previously stopped sending an RTP stream, and
that wants to start sending a new RTP stream, cannot generally re-use
the existing SSRC, and often needs to generate a new SSRC, because an
SSRC cannot change media type (e.g., audio to video) or RTP timestamp
clock rate [RFC7160], and because the SSRC might be associated with a
particular semantic by the application (note: an RTP stream can pause
and restart using the same SSRC, provided RTCP is sent for that SSRC
during the pause; these rules only apply to new RTP streams reusing
an existing SSRC).

6.2. Removing RTP Streams

An SSRC is removed from an RTP session in one of two ways. When an
endpoint stops sending RTP and RTCP packets using an SSRC, then that
SSRC will eventually time out as described in Section 6.3.5 of
[RFC3550]. Alternatively, an SSRC can be explicitly removed from use
by sending an RTCP BYE packet as described in Section 6.3.7 of
[RFC3550]. It is RECOMMENDED that SSRCs be removed from use by
sending an RTCP BYE packet. Note that [RFC3550] requires that the
RTCP BYE SHOULD be the last RTP/RTCP packet sent in the RTP session
for an SSRC. If an endpoint needs to restart an RTP stream after
sending an RTCP BYE for its SSRC, it needs to generate a new SSRC
value for that stream.

The finality of sending RTCP BYE, means that endpoints needs to
consider if the ceasing of transmission of an RTP stream is temporary
or more permanent. Temporary suspension of media transmission using
a particular RTP stream (SSRC) needs to maintain that SSRC as an
active participant, by continuing RTCP transmission for it. That way
the media sending can be resume immediately, knowing that the context
is in place. Permanent transmission halting needs to send RTCP BYE
to allow the other participants to use the RTCP bandwidth resources
and clean up their state databases.

An endpoint that ceases transmission of all its RTP streams but
remains in the RTP session MUST maintain at least one SSRC that is to
be used for RTCP reporting and feedback (i.e., it cannot send a BYE
for all SSRCs, but needs to retain at least one active SSRC). As

some Feedback packets can be bound to media type there might be need
to maintain one SSRC per media type within an RTP session. An
alternative can be to create a new SSRC to use for RTCP reporting and
feedback. However, to avoid the perception that an endpoint drops
completely out of an RTP session such a new SSRC ought to be first
established before terminating all the existing SSRCs.

7. 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 [RFC3556]), 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, or too seldom, than is considered desirable for a RTP stream.

   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, but
   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.

   Different media rates or desired RTCP behaviours can also occur with
   SSRCs carrying the same media type. A common case in multiparty
   conferencing is when a small number of video streams are shown in
   high resolution, while the others are shown as low resolution
   thumbnails, with the choice of which is shown in high resolution
   being voice activity controlled. Here the differences are both in
   actual media rate and in choices for what feedback messages might be
   needed. Other examples of differences that can exist are due to the
   intended usage of a media source. A media source carrying the video

of the speaker in a conference is different from a document camera. Basic parameters that can differ in this case are frame-rate, acceptable end-to-end delay, and the SNR fidelity of the image. These differences affect not only the needed bit-rates, but also possible transmission behaviours, usable repair mechanisms, what feedback messages the control and repair requires, the transmission requirements on those feedback messages, and monitoring of the RTP stream delivery. Other similar scenarios can also exist.

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.

7.1. Timing out SSRCs

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

7.1.1. Problems with the RTP/AVPF T_rr_interval Parameter

   The RTP/AVPF profile includes a method to prevent regular 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.

7.1.2. 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 7.1.1. 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 7.1.4.

7.1.3. 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 7.1.4 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 7.1.1 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.

7.1.4. 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:

o 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.

o For the RTP/AVP, RTP/SAVP, RTP/AVPF, and RTP/SAVPF profiles,
   calculation of Td, for the purpose of calculating the participant
   timeout only, 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. Specifically, the first paragraph of
Section 3.5.4 of [RFC4585] is updated to use Tmin instead of
T_rr_interval in the timeout calculation for RTP/AVPF entities.

7.2. 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].

7.2.1. 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 without
changing the scaled minimum interval, we need to increase the RTCP
bandwidth fraction from 5%; changing the session bandwidth to a
higher value would reduce the Tmin. However, if using the default 5%
allocation of RTCP bandwidth, an increase will result in more SSRCs
being supported given a fixed Td target.

Based on the above, when using the RTP/AVP profile or the RTP/SAVP
profile, the key limitation for rapid RTCP reporting in small unicast
sessions is going to be the Tmin value. 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.

7.2.2. 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 but smaller than Tmin 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 7.1.1. 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 profiles, using T_rr_interval = 0 has
utility, and results in a behaviour where the RTCP transmission is
only limited by the bandwidth, i.e., no Tmin limitations at all.
This allows more frequent regular RTCP reporting than can be achieved
using the RTP/AVP profile. Many configurations of RTCP will not
consume all the bandwidth that they have been configured to use, but
this configuration will consume what it has been given. Note that
the same behaviour will be achieved as long as T_rr_interval is
smaller than 1/3 of Td as that prevents T_rr_interval from affecting
the transmission.

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 each type or stream.

8. 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.

9. IANA Considerations

   No IANA actions are needed.

10. Acknowledgments

   The authors like to thank Harald Alvestrand and everyone else who has
   been involved in the development of this document.

11. References

11.1. Normative References

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

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

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

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

   [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for
              Real-time Transport Control Protocol (RTCP)-Based Feedback
              (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February
              2008, <http://www.rfc-editor.org/info/rfc5124>.

   [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size
              Real-Time Transport Control Protocol (RTCP): Opportunities
              and Consequences", RFC 5506, DOI 10.17487/RFC5506, April
              2009, <http://www.rfc-editor.org/info/rfc5506>.

11.2. 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-11 (work in
              progress), November 2015.

   [I-D.ietf-avtcore-rtp-multi-stream-optimisation]
              Lennox, J., Westerlund, M., Wu, W., 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-08 (work
              in progress), October 2015.

   [I-D.ietf-avtcore-rtp-topologies-update]
              Westerlund, M. and S. Wenger, "RTP Topologies", draft-
              ietf-avtcore-rtp-topologies-update-10 (work in progress),
              July 2015.

   [I-D.ietf-clue-framework]
              Duckworth, M., Pepperell, A., and S. Wenger, "Framework
              for Telepresence Multi-Streams", draft-ietf-clue-
              framework-24 (work in progress), November 2015.

   [I-D.ietf-mmusic-sdp-bundle-negotiation]
              Holmberg, C., Alvestrand, H., and C. Jennings,
              "Negotiating Media Multiplexing Using the Session
              Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-
              negotiation-23 (work in progress), July 2015.

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

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

   [RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K.
              Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830,
              DOI 10.17487/RFC3830, August 2004,
              <http://www.rfc-editor.org/info/rfc3830>.

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

   [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman,
              "Codec Control Messages in the RTP Audio-Visual Profile
              with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104,
              February 2008, <http://www.rfc-editor.org/info/rfc5104>.

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

   [RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis,
              "RTP Payload Format for Scalable Video Coding", RFC 6190,
              DOI 10.17487/RFC6190, May 2011,
              <http://www.rfc-editor.org/info/rfc6190>.

   [RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla,
              "Guidelines for Choosing RTP Control Protocol (RTCP)
              Canonical Names (CNAMEs)", RFC 7022, DOI 10.17487/RFC7022,
              September 2013, <http://www.rfc-editor.org/info/rfc7022>.

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

Authors' Addresses

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

Email: jonathan@vidyo.com

Magnus Westerlund
Ericsson
Farogatan 2
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