AVTCORE WG   M. Westerlund
Internet-Draft   Ericsson
Updates: 3550, 3551 (if approved)   C. Perkins
Intended status: Standards Track   University of Glasgow
Expires: September 10, 2015   J. Lennox
                                                             Vidyo
                                                     March 9, 2015

Sending Multiple Types of Media in a Single RTP Session
draft-ietf-avtcore-multi-media-rtp-session-07

Abstract

   This document specifies how an RTP session can contain RTP Streams
   with media from multiple media types such as audio, video, and text.
   This has been restricted by the RTP Specification, and thus this
   document updates RFC 3550 and RFC 3551 to enable this behaviour for
   applications that satisfy the applicability for using multiple media
   types in a single RTP session.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

When the Real-time Transport Protocol (RTP) [RFC3550] was designed,
close to 20 years ago, IP networks were different to those deployed
at the time of this writing. The virtually ubiquitous deployment of
Network Address Translators (NAT) and Firewalls has since increased
the cost and likely-hood of communication failure when using many
different transport flows. Hence, there is pressure to reduce the
number of concurrent transport flows used by RTP applications.

The RTP specification recommends against sending several different
types of media, for example audio and video, in a single RTP session.

The RTP profile for Audio and Video Conferences with Minimal Control
(RTP/AVP) [RFC3551] mandates a similar restriction. The motivation
for these limitations is partly to allow lower layer Quality of
Service (QoS) mechanisms to be used, and partly due to limitations of
the RTCP timing rules that assumes all media in a session to have
similar bandwidth. The Session Description Protocol (SDP) [RFC4566]
is one of the dominant signalling methods for establishing RTP
sessions, and has enforced this rule by not allowing multiple media
types for a given destination or set of ICE candidates.

The fact that these limitations have been in place for so long, in
addition to RFC 3550 being written without fully considering the use
of multiple media types in an RTP session, results in a number of
issues when allowing this behaviour. This memo updates [RFC3550] and
[RFC3551] with important considerations regarding applicability and
functionality when using multiple types of media in an RTP session,
including normative specification of behaviour. This memo makes no
changes to RTP behaviour when using multiple RTP streams with media
of the same type (e.g., multiple audio streams or multiple video
streams) in a single RTP session. Instead it relies on the
clarifications in [I-D.ietf-avtcore-rtp-multi-stream].

This memo is structured as follows. First, some basic definitions
are provided. This is followed by a background that discusses the
motivation in more detail. A overview of the solution of how to
provide multiple media types in one RTP session is then presented.
Next is the formal applicability this specification have followed by
the normative specification. This is followed by a discussion how
some RTP/RTCP Extensions are expected to function in the case of
multiple media types in one RTP session. A specification of the
requirements on signalling from this specification and a look how
this is realized in SDP using Bundle
[I-D.ietf-mmusic-sdp-bundle-negotiation]. The memo ends with the
security considerations.

2. Definitions

Media Type: The general type of media data used by a real-time
   application. The media type corresponds to the value used in the
   <media> field of an SDP m= line. The media types defined at the
   time of this writing are "audio", "video", "text", "application",
   and "message".

Quality of Service (QoS): Network mechanisms that are intended to
   ensure that the packets within a flow or with a specific marking
   are transported with certain properties.

The terms Encoded Stream, Endpoint, Media Source, RTP Session, and
RTP Stream are used as defined in
[I-D.ietf-avtext-rtp-grouping-taxonomy].

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

3. Motivation

The existence of NATs and Firewalls at almost all Internet access has
had implications on protocols like RTP that were designed to use
multiple transport flows. First of all, the NAT/FW traversal
solution needs to ensure that all these transport flows are
established. This has three consequences:

1. Increased delay to perform the transport flow establishment

2. The more transport flows, the more state and the more resource
    consumption in the NAT and Firewalls. When the resource
    consumption in NAT/FWs reaches their limits, unexpected
    behaviours usually occur.

3. More transport flows means a higher risk that some transport flow
    fails to be established, thus preventing the application to
    communicate.

Using fewer transport flows reduces the risk of communication
failure, improved establishment behaviour and less load on NAT and
Firewalls.

Furthermore, we note that many RTP-using applications don't utilize
any network level Quality of Service (QoS) functions. Nor do they
expect or desire any separation in network treatment of its media
packets, independent of whether they are audio, video or text. When
an application has no such desire, it doesn't need to provide a
transport flow structure that simplifies flow based QoS.

For applications that don't require different lower-layer QoS for
different media types, and that have no special requirements for RTP
extensions or RTCP reporting, the requirement to separate different
media into different RTP sessions might seem unnecessary. Provided
the application accepts that all media flows will get similar RTCP
reporting, using the same RTP session for several types of media at
once appears a reasonable choice. The architecture ought to be
agnostic about the type of media being carried in an RTP session to
the extent possible given the constraints of the protocol.

4. Overview of Solution

   The goal of the solution is to enable each RTP session to contain
   more than just one media type. This includes having multiple RTP
   sessions containing a given media type, for example having three
   sessions containing both video and audio.

   The solution is quite straightforward. The first step is to override
   the SHOULD and SHOULD NOT language of the RTP specification
   [RFC3550]. Similar change is needed to a sentence in Section 6 of
   [RFC3551] that states that "different media types SHALL NOT be
   interleaved or multiplexed within a single RTP Session". This is
   resolved by appropriate exception clauses given that this
   specification and its applicability is followed.

   Within an RTP session where multiple media types have been configured
   for use, an SSRC can only send one type of media during its lifetime
   (i.e., it can switch between different audio codecs, since those are
   both the same type of media, but cannot switch between audio and
   video). Different SSRCs MUST be used for the different media
   sources, the same way multiple media sources of the same media type
   already have to do. The payload type will inform a receiver which
   media type the SSRC is being used for. Thus the payload type MUST be
   unique across all of the payload configurations independent of media
   type that is used in the RTP session.

   Some few extra considerations within the RTP sessions also needs to
   be considered. RTCP bandwidth and regular reporting suppression
   (RTP/AVPF and RTP/SAVPF) SHOULD be configured to reduce the impact
   for bit-rate variations between RTP streams and media types. It is
   also clarified how timeout calculations are to be done to avoid any
   issues. Certain payload types like FEC also need additional rules.

   The final important part of the solution to this is to use signalling
   and ensure that agreement on using multiple media types in an RTP
   session exists, and how that then is configured. This memo describes
   some existing requirements, while an external reference defines how
   this is accomplished in SDP.

5. Applicability

   This specification has limited applicability, and anyone intending to
   use it needs to ensure that their application and usage meets the
   below criteria.

5.1. Usage of the RTP session

   Before choosing to use this specification, an application implementer
   needs to ensure that they don't have a need for different RTP
   sessions between the media types for some reason. The main rule is
   that if one expects to have equal treatment of all media packets,
   then this specification might be suitable. The equal treatment
   include anything from network level up to RTCP reporting and
   feedback. The document Guidelines for using the Multiplexing
   Features of RTP [I-D.ietf-avtcore-multiplex-guidelines] gives more
   detailed guidance on aspects to consider when choosing how to use RTP
   and specifically sessions.

      There is some work in progress
      [I-D.westerlund-avtcore-transport-multiplexing] that attempt to
      address a solution for RTP-using applications that need or would
      prefer multiple RTP sessions, but do not require the
      functionalities or behaviours that multiple transport flows give.

   The second important consideration is the resulting behaviour when
   media flows to be sent within a single RTP session does not have
   similar RTCP requirements. There are limitations in the RTCP timing
   rules, and this implies a common RTCP reporting interval across all
   participants in a session. If an RTP session contains flows with
   very different RTCP requirements, for example due to RTP Streams
   bandwidth consumption and packet rate, for example low-rate audio
   coupled with high-quality video, this can result in either excessive
   or insufficient RTCP for some flows, depending how the RTCP session
   bandwidth, and hence reporting interval, is configured. This is
   discussed further in Section 6.4.

5.2. Signalled Support

   Usage of this specification is not compatible with anyone following
   RFC 3550 and intending to have different RTP sessions for each media
   type. Therefore there needs to be mutual agreement to use multiple
   media types in one RTP session by all participants within that RTP
   session. This agreement has to be determined using signalling in
   most cases.

   This requirement can be a problem for signalling solutions that can't
   negotiate with all participants. For declarative signalling
   solutions, mandating that the session is using multiple media types
   in one RTP session can be a way of attempting to ensure that all
   participants in the RTP session follow the requirement. However, for
   signalling solutions that lack methods for enforcing that a receiver
   supports a specific feature, this can still cause issues.

5.3. **Homogeneous Multi-party**

   In multiparty communication scenarios it is important to separate two
   different cases. One case is where the RTP session contains multiple
   participants in a common RTP session. This occurs for example in Any
   Source Multicast (ASM) and Relay (Transport Translator) topologies as
   defined in RTP Topologies [I-D.ietf-avtcore-rtp-topologies-update].
   It can also occur in some implementations of RTP mixers that share
   the same SSRC/CSRC space across all participants. The second case is
   when the RTP session is terminated in a middlebox and the other
   participants sources are projected or switched into each RTP session
   and rewritten on RTP header level including SSRC mappings.

   For the first case, with a common RTP session or at least shared
   SSRC/CSRC values, all participants in multiparty communication are
   REQUIRED to support multiple media types in an RTP session. An
   participant using two or more RTP sessions towards a multiparty
   session can't be collapsed into a single session with multiple media
   types. The reason is that in case of multiple RTP sessions, the same
   SSRC value can be use in both RTP sessions without any issues, but
   when collapsed to a single session there is an SSRC collision. In
   addition some collisions can't be represented in the multiple
   separate RTP sessions. For example, in a session with audio and
   video, an SSRC value used for video will not show up in the Audio RTP
   session at the participant using multiple RTP sessions, and thus not
   trigger any collision handling. Thus any application using this type
   of RTP session structure MUST have a homogeneous support for multiple
   media types in one RTP session, or be forced to insert a translator
   node between that participant and the rest of the RTP session.

   For the second case of separate RTP sessions for each multiparty
   participant and a central node it is possible to have a mix of single
   RTP session users and multiple RTP session users as long as one is
   willing to remap the SSRCs used by a participant with multiple RTP
   sessions into non-used values in the single RTP session SSRC space
   for each of the participants using a single RTP session with multiple
   media types. It can be noted that this type of implementation has to
   understand all types of RTP/RTCP extension being used in the RTP
   sessions to correctly be able to translate them between the RTP
   sessions. It might also suffer issues due to differencies in
   configured RTCP bandwidth and other parameters between the RTP
   sessions. It can also negatively impact the possibility for loop
   detection, as SSRC/CSRC can't be used to detect the loops, instead
   some other RTP stream or media source identity name space that is
   common across all interconnect parts are needed.

5.4. Reduced number of Payload Types

   An RTP session with multiple media types in it have only a single
   7-bit Payload Type range for all its payload types. Within the 128
   available values, only 96 or less if "Multiplexing RTP Data and
   Control Packets on a Single Port" [RFC5761] is used, all the
   different RTP payload configurations for all the media types need to
   fit in the available space. For most applications this will not be a
   real problem, but the limitation exists and could be encountered.

5.5. Stream Differentiation

   If network level differentiation of the RTP streams with different
   media types is desired, using this specification can cause severe
   limitations. All RTP streams in an RTP session, independent of the
   media type, will be sent over the same underlying transport flow.
   Any flow-based Quality of Service (QoS) mechanism will be unable to
   provide differentiated treatment between different media types, e.g.
   to prioritize audio over video. If differentiated treatment is
   desired using flow-based QoS, separate RTP sessions over different
   underlying transport flows needs to be used.

   Marking-based QoS schemes like DiffServ can be affected if a network
   ingress is the one that performs, markings based on flows. Endpoint
   marking where the network API supports marking on individual packet
   level will be unaffected by this specification. However, there exist
   limitations, as discussed in [I-D.ietf-dart-dscp-rtp], on how
   different traffic classes can be applied on different packets or RTP
   streams within a single transport flow.

5.6. Non-compatible Extensions

   There exist some RTP and RTCP extensions that rely on the existence
   of multiple RTP sessions. If the goal of using an RTP session with
   multiple media types is to have only a single RTP session, then these
   extensions can't be used. If one has no need to have different RTP
   sessions for the media types but is willing to have multiple RTP
   sessions, one for the main media transmission and one for the
   extension, they can be used. It is to be noted that this assumes
   that it is possible to get the extension working when the related RTP
   session contains multiple media types.

   Identified RTP/RTCP extensions that require multiple RTP Sessions
   are:

   RTP Retransmission: RTP Retransmission [RFC4588] has a session
      multiplexed mode. It also has a SSRC multiplexed mode that can be

used instead. So use the mode that is suitable for the RTP
application.

XOR-Based FEC: The RTP Payload Format for Generic Forward Error
Correction [RFC5109] and its predecessor [RFC2733] requires a
separate RTP session unless the FEC data is carried in RTP Payload
for Redundant Audio Data [RFC2198]. However, using the Generic
FEC with the Redundancy payload has another set of restrictions,
see Section 7.2.

Note that the Source-Specific Media Attributes [RFC5576]
specification defines an SDP syntax (the "FEC" semantic of the
"ssrc-group" attribute) to signal FEC relationships between
multiple RTP streams within a single RTP session. However, this
can't be used as the FEC repair packets need to have the same SSRC
value as the source packets being protected. [RFC5576] does not
normatively update and resolve that restriction. There is ongoing
work on an ULP extension to allow it be use FEC RTP streams within
the same RTP Session as the source stream
[I-D.lennox-payload-ulp-ssrc-mux].

6. RTP Session Specification

This section defines what needs to be done or avoided to make an RTP
session with multiple media types function without issues.

6.1. RTP Session

Section 5.2 of "RTP: A Transport Protocol for Real-Time Applications"
[RFC3550] states:

For example, in a teleconference composed of audio and video media
encoded separately, each medium SHOULD be carried in a separate
RTP session with its own destination transport address.

Separate audio and video streams SHOULD NOT be carried in a single
RTP session and demultiplexed based on the payload type or SSRC
fields.

This specification changes both of these sentences. The first
sentence is changed to:

For example, in a teleconference composed of audio and video media
encoded separately, each medium SHOULD be carried in a separate
RTP session with its own destination transport address, unless
specification [RFCXXXX] is followed and the application meets the
applicability constraints.

The second sentence is changed to:

> Separate audio and video media sources SHOULD NOT be carried in a
> single RTP session and demultiplexed based on the payload type or
> SSRC fields, unless multiplexed based on both SSRC and payload
> type and usage meets what Multiple Media Types in an RTP Session
> [RFCXXXX] specifies.

Second paragraph of Section 6 in RTP Profile for Audio and Video
Conferences with Minimal Control [RFC3551] says:

> The payload types currently defined in this profile are assigned
> to exactly one of three categories or media types: audio only,
> video only and those combining audio and video. The media types
> are marked in Tables 4 and 5 as "A", "V" and "AV", respectively.
> Payload types of different media types SHALL NOT be interleaved or
> multiplexed within a single RTP session, but multiple RTP sessions
> MAY be used in parallel to send multiple media types. An RTP
> source MAY change payload types within the same media type during
> a session. See the section "Multiplexing RTP Sessions" of RFC
> 3550 for additional explanation.

This specifications purpose is to violate that existing SHALL NOT
under certain conditions. Thus also this sentence has to be changed
to allow for multiple media type's payload types in the same session.
The above sentence is changed to:

> Payload types of different media types SHALL NOT be interleaved or
> multiplexed within a single RTP session unless as specified and
> under the restriction in Multiple Media Types in an RTP Session
> [RFCXXXX]. Multiple RTP sessions MAY be used in parallel to send
> multiple media types.

RFC-Editor Note: Please replace RFCXXXX with the RFC number of this
specification when assigned.

We can now go on and discuss the five bullets that are motivating the
previous in Section 5.2 of the RTP Specification [RFC3550]. They are
repeated here for the reader's convenience:

1. If, say, two audio streams shared the same RTP session and the
    same SSRC value, and one were to change encodings and thus
    acquire a different RTP payload type, there would be no general
    way of identifying which stream had changed encodings.

2. An SSRC is defined to identify a single timing and sequence
    number space. Interleaving multiple payload types would require
    different timing spaces if the media clock rates differ and would

require different sequence number spaces to tell which payload
type suffered packet loss.

3. The RTCP sender and receiver reports (see Section 6.4 of RFC
    3550) can only describe one timing and sequence number space per
    SSRC and do not carry a payload type field.

4. An RTP mixer would not be able to combine interleaved streams of
    incompatible media into one stream.

5. Carrying multiple media in one RTP session precludes: the use of
    different network paths or network resource allocations if
    appropriate; reception of a subset of the media if desired, for
    example just audio if video would exceed the available bandwidth;
    and receiver implementations that use separate processes for the
    different media, whereas using separate RTP sessions permits
    either single- or multiple-process implementations.

Bullets 1 to 3 are all related to that each media source has to use
one or more unique SSRCs to avoid these issues as mandated below
(Section 6.2). Bullet 4 can be served by two arguments, first of all
each SSRC will be associated with a specific media type, communicated
through the RTP payload type, allowing a middlebox to do media type
specific operations. The second argument is that in many contexts
blind combining without additional contexts are anyway not suitable.
Regarding bullet 5 this is a understood and explicitly stated
applicability limitations for the method described in this document.

6.2. Sender Source Restrictions

A SSRC in the RTP session MUST only send one media type (audio,
video, text etc.) during the SSRC's lifetime. The main motivation is
that a given SSRC has its own RTP timestamp and sequence number
spaces. The same way that you can't send two encoded streams of
audio on the same SSRC, you can't send one encoded audio and one
encoded video stream on the same SSRC. Each encoded stream when made
into an RTP stream needs to have the sole control over the sequence
number and timestamp space. If not, one would not be able to detect
packet loss for that particular encoded stream. Nor can one easily
determine which clock rate a particular SSRCs timestamp will increase
with. For additional arguments why RTP payload type based
multiplexing of multiple media sources doesn't work see
[I-D.ietf-avtcore-multiplex-guidelines].

6.3. **Payload Type Applicability**

   Most Payload Types have a native media type, like an audio codec is
   natural belonging to the audio media type. However, there exist a
   number of RTP payload types that don't have a native media type. For
   example, transport robustness mechanisms like RTP Retransmission
   [RFC4588] and Generic FEC [RFC5109] inherit their media type from
   what they protect. RTP Retransmission is explicitly bound to the
   payload type it is protecting, and thus will inherit it. However
   Generic FEC is a excellent example of an RTP payload type that has no
   natural media type. The media type for what it protects is not
   relevant as it is the recovered RTP packets that have a particular
   media type, and thus Generic FEC is best categorized as an
   application media type.

   The above discussion is relevant to what limitations exist for RTP
   payload type usage within an RTP session that has multiple media
   types. In fact this document (Section 7.2) suggest that for usage of
   Generic FEC (XOR-based) as defined in RFC 5109 can actually use a
   single media type when used with independent RTP sessions for source
   and repair data.

      Note a particular SSRC carrying Generic FEC will clearly only
      protect a specific SSRC and thus that instance is bound to the
      SSRC's media type. For this specific case, it is possible to have
      one be applicable to both. However, in cases when the signalling
      is setup to enable fall back to using separate RTP sessions, then
      using a different media type, e.g. application, than the media
      being protected can create issues.

6.4. **RTCP Considerations**

   Guidelines for handling RTCP when sending multiple RTP streams with
   disparate rates in a single RTP session are outlined in
   [I-D.ietf-avtcore-rtp-multi-stream]. These guidelines apply when
   sending multiple types of media in a single RTP session if the
   different types of media have different rates.

7. **Extension Considerations**

   This section discusses the impact on some RTP/RTCP extensions due to
   usage of multiple media types in on RTP session. Only extensions
   where something worth noting has been included.

7.1. RTP Retransmission

   SSRC-multiplexed RTP retransmission [RFC4588] is actually very
   straightforward. Each retransmission RTP payload type is explicitly
   connected to an associated payload type. If retransmission is only
   to be used with a subset of all payload types, this is not a problem,
   as it will be evident from the retransmission payload types which
   payload types have retransmission enabled for them.

   Session-multiplexed RTP retransmission is also possible to use where
   an retransmission session contains the retransmissions of the
   associated payload types in the source RTP session. The only
   difference to the previous case is if the source RTP session is one
   which contains multiple media types. This results in the
   retransmission streams in the RTP session for the retransmission
   having multiple associated media types.

   When using SDP signalling for a multiple media type RTP session, i.e.
   BUNDLE [I-D.ietf-mmusic-sdp-bundle-negotiation], the session
   multiplexed case do require some recommendations on how to signal
   this. To avoid breaking the semantics of the FID grouping as defined
   by [RFC5888] each media line can only be included in one FID group.
   FID is used by RTP retransmission to indicate the SDP media lines
   that is a source and retransmission pair. Thus, for SDP using
   BUNDLE, each original media source (m= line) that is retransmitted
   needs a corresponding media line in the retransmission RTP session.
   In case there are multiple media lines for retransmission, these
   media lines will form a independent BUNDLE group from the BUNDLE
   group with the source streams.

   Below is an SDP example (Figure 1) which shows the grouping
   structures. This example is not legal SDP and only the most
   important attributes has been left in place. Note that this SDP is
   not an initial BUNDLE offer. As can be seen there are two bundle
   groups, one for the source RTP session and one for the
   retransmissions. Then each of the media sources are grouped with its
   retransmission flow using FID, resulting in three more groupings.

```
         a=group:BUNDLE foo bar fiz
         a=group:BUNDLE zoo kelp glo
         a=group:FID foo zoo
         a=group:FID bar kelp
         a=group:FID fiz glo
         m=audio 10000 RTP/AVP 0
         a=mid:foo
         a=rtpmap:0 PCMU/8000
         m=video 10000 RTP/AVP 31
         a=mid:bar
         a=rtpmap:31 H261/90000
         m=video 10000 RTP/AVP 31
         a=mid:fiz
         a=rtpmap:31 H261/90000
         m=audio 40000 RTP/AVPF 99
         a=rtpmap:99 rtx/90000
         a=fmtp:99 apt=0;rtx-time=3000
         a=mid:zoo
         m=video 40000 RTP/AVPF 100
         a=rtpmap:100 rtx/90000
         a=fmtp:199 apt=31;rtx-time=3000
         a=mid:kelp
         m=video 40000 RTP/AVPF 100
         a=rtpmap:100 rtx/90000
         a=fmtp:199 apt=31;rtx-time=3000
         a=mid:glo
```

      Figure 1: SDP example of Session Multiplexed RTP Retransmission

7.2. Generic FEC

   The RTP Payload Format for Generic Forward Error Correction
   [RFC5109], and also its predecessor [RFC2733], requires some
   considerations, and they are different depending on what type of
   configuration of usage one has.

   Independent RTP Sessions, i.e. where source and repair data are sent
   in different RTP sessions. As this mode of configuration requires
   different RTP session, there has to be at least one RTP session for
   source data, this session can be one using multiple media types. The
   repair session only needs one RTP Payload type indicating repair
   data, i.e. x/ulpfec or x/parityfec depending if RFC 5109 or RFC 2733
   is used. The media type in this session is not relevant and can in
   theory be any of the defined ones. It is RECOMMENDED that one uses
   "Application".

   If one uses SDP signalling with BUNDLE
   [I-D.ietf-mmusic-sdp-bundle-negotiation], then the RTP session

carrying the FEC streams will be its own BUNDLE group. The media
line with the source stream for the FEC and the FEC stream's media
line will be grouped using media line grouping using the FEC or FEC-
FR [RFC5956] grouping. This is very similar to the situation that
arise for RTP retransmission with session multiplexing discussed
above in Section 7.1.

In stream, using RTP Payload for Redundant Audio Data [RFC2198]
combining repair and source data in the same packets. This is
possible to use within a single RTP session. However, the usage and
configuration of the payload types can create an issue. First of all
it might be necessary to have one payload type per media type for the
FEC repair data payload format, i.e. one for audio/ulpfec and one for
text/ulpfec if audio and text are combined in an RTP session.
Secondly each combination of source payload and its FEC repair data
has to be an explicit configured payload type. This has potential
for making the limitation of RTP payload types available into a real
issue.

8. Signalling

The Signalling requirements

Establishing an RTP session with multiple media types requires
signalling. This signalling needs to fulfil the following
requirements:

1.   Ensure that any participant in the RTP session is aware that this
     is an RTP session with multiple media types.

2.   Ensure that the payload types in use in the RTP session are using
     unique values, with no overlap between the media types.

3.   Configure the RTP session level parameters, such as RTCP RR and
     RS bandwidth, AVPF trr-int, underlying transport, the RTCP
     extensions in use, and security parameters, commonly for the RTP
     session.

4.   RTP and RTCP functions that can be bound to a particular media
     type SHOULD be reused when possible also for other media types,
     instead of having to be configured for multiple code-points.
     Note: In some cases one will not have a choice but to use
     multiple configurations.

8.1. SDP-Based Signalling

The signalling of multiple media types in one RTP session in SDP is
specified in "Multiplexing Negotiation Using Session Description
Protocol (SDP) Port Numbers"
[I-D.ietf-mmusic-sdp-bundle-negotiation].

9. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section is to be removed on publication as
an RFC.

10. Security Considerations

Having an RTP session with multiple media types doesn't change the
methods for securing a particular RTP session. One possible
difference is that the different media have often had different
security requirements. When combining multiple media types in one
session, their security requirements also have to be combined by
selecting the most demanding for each property. Thus having multiple
media types can result in increased overhead for security for some
media types to ensure that all requirements are meet.

Otherwise, the recommendations for how to configure and RTP session
do not add any additional requirements compared to normal RTP, except
for the need to be able to ensure that the participants are aware
that it is a multiple media type session. If not that is ensured it
can cause issues in the RTP session for both the unaware and the
aware one. Similar issues can also be produced in an normal RTP
session by creating configurations for different end-points that
doesn't match each other.

11. Acknowledgements

The authors would like to thank Christer Holmberg, Gunnar Hellstroem,
and Charles Eckel for the feedback on the document.

12. References

12.1. Normative References

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

[I-D.ietf-mmusic-sdp-bundle-negotiation]
          Holmberg, C., Alvestrand, H., and C. Jennings,
          "Negotiating Media Multiplexing Using the Session
          Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-
          negotiation-17 (work in progress), March 2015.

[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.

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

12.2. Informative References

[I-D.ietf-avtcore-multiplex-guidelines]
          Westerlund, M., Perkins, C., and H. Alvestrand,
          "Guidelines for using the Multiplexing Features of RTP to
          Support Multiple Media Streams", draft-ietf-avtcore-
          multiplex-guidelines-03 (work in progress), October 2014.

[I-D.ietf-avtcore-rtp-topologies-update]
          Westerlund, M. and S. Wenger, "RTP Topologies", draft-
          ietf-avtcore-rtp-topologies-update-06 (work in progress),
          March 2015.

[I-D.ietf-avtext-rtp-grouping-taxonomy]
          Lennox, J., Gross, K., Nandakumar, S., and G. Salgueiro,
          "A Taxonomy of Grouping Semantics and Mechanisms for Real-
          Time Transport Protocol (RTP) Sources", draft-ietf-avtext-
          rtp-grouping-taxonomy-06 (work in progress), March 2015.

[I-D.ietf-dart-dscp-rtp]
          Black, D. and P. Jones, "Differentiated Services
          (DiffServ) and Real-time Communication", draft-ietf-dart-
          dscp-rtp-10 (work in progress), November 2014.

[I-D.lennox-payload-ulp-ssrc-mux]
          Lennox, J., "Supporting Source-Multiplexing of the Real-
          Time Transport Protocol (RTP) Payload for Generic Forward
          Error Correction", draft-lennox-payload-ulp-ssrc-mux-00
          (work in progress), February 2013.

[I-D.westerlund-avtcore-transport-multiplexing]
          Westerlund, M. and C. Perkins, "Multiplexing Multiple RTP
          Sessions onto a Single Lower-Layer Transport", draft-
          westerlund-avtcore-transport-multiplexing-07 (work in
          progress), October 2013.

[RFC2198] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V.,
          Handley, M., Bolot, J., Vega-Garcia, A., and S. Fosse-
          Parisis, "RTP Payload for Redundant Audio Data", RFC 2198,
          September 1997.

[RFC2733] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format
          for Generic Forward Error Correction", RFC 2733, December
          1999.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session
          Description Protocol", RFC 4566, July 2006.

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

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error
          Correction", RFC 5109, December 2007.

[RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific
          Media Attributes in the Session Description Protocol
          (SDP)", RFC 5576, June 2009.

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and
          Control Packets on a Single Port", RFC 5761, April 2010.

[RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description
          Protocol (SDP) Grouping Framework", RFC 5888, June 2010.

[RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in
          the Session Description Protocol", RFC 5956, September
          2010.

Authors' Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

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

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

Email: csp@csperkins.org

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

Email: jonathan@vidyo.com