Network Working Group
Internet-Draft
Updates: RFC3550
(if approved)
Intended status: Standards Track
Expires: September 10, 2009

C. Perkins
University of Glasgow
T. Schierl
Fraunhofer HHI
March 9, 2009

Rapid Synchronisation of RTP Flows
draft-perkins-avt-rapid-rtp-sync-03.txt

Status of this Memo

Copyright Notice

Abstract

This memo outlines how RTP multimedia sessions are synchronised, and discusses how rapidly such synchronisation can occur. We show that most RTP sessions can be synchronised immediately, but that the use of video switching multipoint conference units (MCUs) or large source specific multicast (SSM) groups can greatly increase the initial synchronisation delay. This increase in delay can be unacceptable to some applications that use layered and/or multi-description codecs.

This memo updates the RTP Control Protocol (RTCP) timing rules to reduce the initial synchronisation delay for SSM sessions. A new feedback packet is defined for use with the Extended RTP Profile for RTCP-based Feedback (RTP/AVPF), allowing video switching MCUs to rapidly request resynchronisation. Two new RTP header extensions are defined to allow rapid synchronisation of late joiners, and guarantee correct timestamp based decoding order recovery for layered codecs in the presence of clock skew.

Table of Contents

1. Introduction

When using RTP to deliver multimedia content it's often necessary to synchronise playout of audio and video components of a presentation. This is achieved using information contained in RTP Control Protocol (RTCP) Sender Report (SR) packets [1]. These are sent periodically, and the components of a multimedia session cannot be synchronised until sufficient RTCP SR packets have been received for each flow to allow the receiver to establish mappings between the media clock used for each flow, and the common (NTP-format) clock used to establish synchronisation.

Recently, concern has been expressed that this synchronisation delay is problematic for some applications, for example those using layered or multi-description video coding. This memo reviews the operations of RTP synchronisation, and describes the synchronisation delay that can be expected. Two backwards compatible extensions to the basic RTP synchronisation mechanism are proposed:

- An enhancement to the Extended RTP Profile for RTCP-based Feedback (RTP/AVPF) [2] is defined to allow receivers to request additional RTCP SR packets, providing the metadata needed to synchronise RTP flows. This can reduce the synchronisation delay when joining sessions with large RTCP reporting intervals, or in the presence of packet loss.

- Two RTP header extensions are defined, to deliver synchronisation metadata in-band with RTP data packets. These extensions provide synchronisation metadata that is aligned with RTP data packets, and so eliminate the need to estimate clock-skew between flows before synchronisation. They can also reduce the need to receive RTCP SR packets before synchronising flows.

The immediate use-case for these extensions is to reduce the delay due to synchronisation when joining a layered video session (e.g. an H.264/SVC session in NI-T mode [7]). The extensions are not specific to layered coding, however, and can be used in any environment when synchronisation latency is an issue.

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 RFC 2119 [3].

2. Synchronisation of RTP Flows

RTP flows are synchronised by receivers based on information that is contained in RTCP SR packets generated by senders (specifically, the

NTP and RTP timestamps). Each type of media (e.g. audio or video) is sent in a separate RTP session, and the receiver associates RTP flows to be synchronised by means of the canonical end-point identifier (CNAME) item included in the RTCP Source Description (SDES) packets generated by the sender. To ensure synchronisation, an RTP sender MUST therefore send periodic compound RTCP packets following Section 6 of RFC 3550 [1].

The timing of these periodic compound RTCP packets will depend on the number of members in each RTP session, the fraction of those that are sending data, the session bandwidth, the configured RTCP bandwidth fraction, and whether the session is multicast or unicast (see RFC 3550 Section 6.2 for details). In summary, RTCP control traffic is allocated a small fraction, generally 5%, of the session bandwidth, and of that fraction, one quarter is allocated to active RTP senders, while receivers use the remaining three quarters (these fractions can be configured via SDP [8]). Each member of an RTP session derives an RTCP reporting interval based on these fractions, whether the session is multicast or unicast, the number of members it has observed, and whether it is actively sending data or not. It then sends a compound RTCP packet on average once per reporting interval (the actual packet transmission time is randomised in the range [0.5 ... 1.5] times the reporting interval to avoid synchronisation of reports).

A minimum reporting interval of 5 seconds is RECOMMENDED, except that the delay before sending the initial report "MAY be set to half the minimum interval to allow quicker notification that the new participant is present" [1]. Also, for unicast sessions, "the delay before sending the initial compound RTCP packet MAY be zero" [1]. In addition, for unicast sessions, and for active senders in a multicast session, the fixed minimum reporting interval MAY be scaled to "360 divided by the session bandwidth in kilobits/second. This minimum is smaller than 5 seconds for bandwidths greater than 72 kb/s." [1]

2.1. Initial Synchronisation Delay

A multimedia session comprises a set of concurrent RTP sessions among a common group of participants, using one RTP session for each media type. For example, a videoconference (which is a multimedia session) might contain an audio RTP session and a video RTP session. To allow a receiver to synchronise the components of a multimedia session, a compound RTCP packet containing an RTCP SR packet and an RTCP SDES packet with a CNAME item MUST be sent to each of the RTP sessions in the multimedia session. A receiver cannot synchronise playout across the multimedia session until such RTCP packets have been received on all of the component RTP sessions. If there is no packet loss, this gives an expected initial synchronisation delay equal to the average time taken to receive the first RTCP packet in the RTP session with

the longest RTCP reporting interval. This will vary between unicast and multicast RTP sessions.

2.1.1. Unicast Sessions

For unicast multimedia sessions, senders SHOULD transmit an initial compound RTCP packet (containing an RTCP SR packet and an RTCP SDES packet with a CNAME item) immediately on joining each RTP session in the multimedia session. The individual RTP sessions are considered to be joined once any in-band signalling for NAT traversal (e.g. [9]) and/or security keying (e.g. [10],[11]) has concluded, and the media path is open. This implies that the initial RTCP packet is sent in parallel with the first data packet following the guidance in RFC 3550 that "the delay before sending the initial compound RTCP packet MAY be zero" and, in the absence of any packet loss, flows can be synchronised immediately.

Note that NAT pinholes, firewall holes, quality-of-service, and media security keys should have been negotiated as part of the signalling, whether in-band or out-of-band, before the first RTCP packet is sent. This should ensure that any middleboxes are ready to accept traffic, and reduce the likelihood that the initial RTCP packet will be lost.

2.1.2. Source Specific Multicast (SSM) Sessions

For multicast sessions, the delay before sending the initial RTCP packet, and hence the synchronisation delay, varies with the session bandwidth and the number of members in the session. For a multicast multimedia session, the average synchronisation delay will depend on the slowest of the component RTP sessions; this will generally be the session with the lowest bandwidth (assuming all the RTP sessions have the same number of members).

When sending to a multicast group, the reduced minimum RTCP reporting interval of 360 seconds divided by the session bandwidth in kilobits per second [1] should be used when synchronisation latency is likely to be an issue. Also, as usual, the reporting interval is halved for the first RTCP packet. Depending on the session bandwidth and the number of members, this gives the following average synchronisation delays:

Session Bandwidth	Number of receivers (single sender assumed): 2	3	4	5	10	100	1000	10000
8 kbps	2.73	4.10	5.47	5.47	5.47	5.47	5.47	5.47
16 kbps	2.50	2.50	2.73	2.73	2.73	2.73	2.73	2.73
32 kbps	2.50	2.50	2.50	2.50	2.50	2.50	2.50	2.50
64 kbps	2.50	2.50	2.50	2.50	2.50	2.50	2.50	2.50
128 kbps	1.41	1.41	1.41	1.41	1.41	1.41	1.41	1.41
256 kbps	0.70	0.07	0.07	0.07	0.07	0.07	0.07	0.07
512 kbps	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35
1 Mbps	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18
2 Mbps	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09
4 Mbps	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04

Figure 1: Average RTCP Reporting Interval (seconds)

These numbers assume a single-source multicast channel with a single active sender, which the rules in RFC 3550 section 6.3 give a fixed fraction of the RTCP bandwidth irrespective of the number of receivers. It can be seen that they are sufficient for lip-synchronisation without excessive delay, but might be viewed as having too much latency for synchronising parts of a layered video stream.

The RTCP interval is randomised in the usual manner, so the minimum synchronisation delay will be half these intervals, and the maximum delay will be 1.5 times these intervals. Note also that these RTCP intervals are calculated assuming perfect knowledge of the number of members in the session. In practice, an implementation will have only limited knowledge of the size of the session when joining, and will likely send its initial report early compared to these values, following the RTCP reconsideration rules.

2.1.3. Any Source Multicast (ASM) Sessions

(tbd)

For ASM sessions, the fraction of members that are senders plays an important role, and imply more variation in average RTCP reporting interval.

2.1.4. Discussion

For unicast sessions, the existing RTCP SR-based mechanism allows for immediate synchronisation, provided the initial RTCP packet is not lost.

For SSM sessions, the initial synchronisation delay is sufficient for

lip-synchronisation, but may be larger than desired for some layered codecs. The rationale for not sending immediate RTCP packets for multicast groups is to avoid implosion of requests when large numbers of members simultaneously join the group ("flash crowd"). This is not an issue for SSM senders, since there can be at most one sender, so it might be desirable to allow SSM senders to send an immediate RTCP SR on joining a session (as is currently allowed for unicast sessions, which also don't suffer from the implosion problem). SSM receivers using unicast feedback would not be allowed to send immediate RTCP. This would be a change to RFC 3550, if accepted.

For ASM session... (tbd)

In all cases, it is possible that the initial RTCP SR packet is lost. In this case, the receiver will not be able to synchronise the media until the reporting interval has passed, and the next RTCP SR packet is sent. This is undesirable. Section 3.1 defines a new RTP/AVPF transport layer feedback message to request an RTCP SR be generated, allowing rapid resynchronisation in the case of packet loss.

2.2. Synchronisation for Late Joiners

Synchronisation between RTP sessions is potentially slower for late joiners, than for participants present at the start of the session. The reasons for this are two-fold:

1. Many of the optimisations that allow rapid transmission of RTCP SR packets apply only at the start of a session. This implies that a new participant may have to wait a complete RTCP reporting interval for each session before receiving the necessary data to synchronise media streams. This might potentially take several seconds, depending on the configured session bandwidth and the number of participants.

2. Additional synchronisation delay comes from the nature of the RTCP timing rules. Packets are generated on average once per reporting interval but with the exact transmission times being randomised +/- 50% to avoid synchronisation of reports. This is important to avoid network congestion in multicast sessions, but does mean that the timing of RTCP SR reports for different RTP sessions aren't synchronised. Accordingly, a receiver must estimate the skew on the NTP-format clock in order to align RTP timestamps across sessions. This estimation is an essential part of an RTP synchronisation implementation, and can be done exactly given sufficient reports. Collecting sufficient RTCP SR data to perform this estimation, however, may require several reports, further increasing the synchronisation delay.

These delays are likely an issue for tuning in to an ongoing multicast RTP session, or for video switching MCUs.

3. Reducing RTP Synchronisation Delays

Two backwards compatible RTP extensions are defined to reduce the possible synchronisation delay: a rapid resynchronisation request message, and RTP header extensions that can convey synchronisation metadata in-band.

3.1. Rapid Resynchronisation Request

The general format of an RTP/AVPF transport layer feedback message is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|   FMT   | PT=RTPFB=205  |          length               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of packet sender                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of media source                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:            Feedback Control Information (FCI)                 :
:                                                               :
```

A new feedback message type, RTCP-SR-REQ, is defined with FMT = XXX. (the next available FMT is 5?) This MAY be sent to indicate that a receiver is unable to synchronise media streams, and desires that the media source send an RTCP SR packet as soon as possible (within the constraints of the RTCP early feedback rules). On receipt of this, the media source SHOULD generate an RTCP SR packet as soon as possible within the RTCP early feedback rules. That RTCP SR packet MAY be sent as a non-compound RTCP packet, if this has been negotiated.

The Feedback Control Information (FCI) part of the packet is emtpy. The SSRC of packet sender indicates the member that is unable to synchronise media streams, while the SSRC of media source indicates the sender of the media it is unable to synchronise. The lenght MUST equal 2.

(tbd: discuss what happens if the feedback target is not co-located with the sender)

3.2. In-band Delivery of Synchronisation Metadata

The RTP header extension mechanism defined in [4] can be adopted to carry an OPTIONAL NTP format wall clock timestamp in RTP data packets. If such a timestamp is included, it MUST correspond to the same time instant as the RTP timestamp in the packet's header, and MUST be derived from the same clock used to generate the NTP format timestamps included in RTCP SR packets. The receiver can use the information provided as input to the synchronisation algorithm, as-if an RTCP SR packet had been received for the flow.

Two variants are defined for this header extension. The first variant extends the RTP header with a 64 bit NTP timestamp format timestamp as defined in [5]. The second variant carries the lower 24 bit part of the Seconds of a NTP timestamp format timestamp and the 32 bit of the Fraction of a NTP timestamp format timestamp. The formats of the two variants are shown below.

```
    Variant A (16 byte) of the NTP header extension:

    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |V=2|P|1|  CC   |M|     PT      |       sequence number         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+R
   |                           timestamp                           |T
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+P
   |           synchronization source (SSRC) identifier            |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |       0xBE    |    0xDE       |           length=3            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+E
   |  ID-A | L=7   |   NTP timestamp format - Seconds (bit 0-23)   |x
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+t
   |NTP Sec.(24-31)|   NTP timestamp format - Fraction(bit 0-23)   |n
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |NTP Frc.(24-31)|    0 (pad)    |    0 (pad)    |    0 (pad)    |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |                          payload data                         |
   |                              ....                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Variant B (12 byte) of the NTP header extension:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|1|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+R
|                           timestamp                           |T
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+P
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|       0xBE    |    0xDE       |           length=2            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+E
|  ID-B | L=6   |  NTP timestamp format - Seconds (bit 8-31)    |x
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+t
|           NTP timestamp format - Fraction (bit 0-31)          |n
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                          payload data                         |
|                              ....                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

An NTP timestamp format timestamp MAY be included on any RTP packets the sender chooses, but it is RECOMMENDED when performing timestamp based decoding order recovery for layered codecs transported in multiple RTP flows, as discussed in Section 4. This header extension MAY be also sent on the RTP packets corresponding to a video random access point, and on the associated audio packets, to allow rapid synchronisation for late joiners and in video switching scenarios. Note that the inclusion of an RTP header extension will reduce the efficiency of RTP header compression, if it is used.

In all cases, irrespective of whether in-band NTP timestamp format timestamps are included or not, regular RTCP SR packets MUST be sent to provide backwards compatibility with receivers that synchronize RTP flows according to [1]. The sender reports are also required to receive the upper 8 bit of the Seconds of the NTP timestamp format timestamp not included in the NTP header extension.

3.3. Signalling

The signaling of using the NTP header extension defined in Section 3.2 MUST be applied as defined in [4].

(tbd - URI, ID-A and ID-B for the NTP header extension need to be defined, e.g. URI: "a=extmap:ID-A urn:ietf:params:rtp-hdrext:ntp-64"] "a=extmap:ID-B urn:ietf:params:rtp-hdrext:ntp-56"]

4. Application to Decoding Order Recovery in Layered Codecs

Based on the timestamp contained each RTP data packet, and the mapping to an NTP-format wall-clock time, a decoding order recovery process is applied if a media as result of a layered coding process is transported in multiple RTP flows. This recovers the decoding order of media frames or samples at the receiver. Especially when transporting layered video, the decoding order recovery process is not straight forward. In this section, we provide guidance on how to use RTP/NTP timing information for decoding order recovery.

4.1. Problem description

One option for decoding order recovery in layered codecs is to use the NTP (sample presentation) timestamps to reorder data of the same layered media transported in different RTP flows. For a timestamp-based decoding order recovery process, it is crucial to allow exact alignment of media frames respectively samples using the NTP timing information. In the presence of clock skew, it may not be possible to derive exact matching NTP timestamps using the NTP wallclock in each RTP flow's RTCP sender reports. This is due to the fact that RTCP sender reports are not sent at the same point of time in the multiple RTP flows transporting data of the same layered media. If the RTCP SR packets are not send at the same time, they therefore do not contain the same NTP wallclock timestamp. If there is a skew present in the clock used for NTP wallclock timestamp generation, using different wallclock timestamps for the same sampling instance in the RTP flow inevitably leads to non-matching NTP timestamps generated from RTP timestamps and wallclock timestamp in the multiple RTP flows. In order to allow a common and straight forward timestamp-based decoding order recovery process, it is important to guarantee exact matching of NTP timestamps. Thus in the presence of non-perfect clocks, which should be the normal case, an additional mechanism shall be used. An exact inter-flow alignment of NTP timestamps can be guaranteed, if the NTP header extension is always inserted at the same timing position in all the RTP flows in question and if those NTP header extensions are used to update the NTP-RTP relation in all RTP flows at the same point of time.

4.2. Use of RTP Header Extensions for Synchronisation

The NTP header extension SHOULD be used with a layered, multi-description, or multi-view codec, to provide exact matching of NTP timestamps between layers, descriptions, or views transported in different RTP flows to allow timestamp-based decoding order recovery. If the NTP header extension is inserted for RTP flows transporting samples or parts of samples of the same layered media, the NTP header extension SHALL be included at least once in each of the RTP flows of

the same media for the sampling time instance of an insertion of a NTP header extension and such synchronously inserted NTP header extensions SHALL contain the same NTP timestamp. The frequency of inserting NTP header extensions in the RTP flows is up to the sender.

Note: If the decoding order of RTP flows is given by any means (as e.g., by mechanism defined in [6]), the NTP timestamp provided by the header extension allows to collect data of the same sample from the RTP flows, forming the sample decoding order.

It is RECOMMENDED that the receiver uses for timestamp-based decoding order recovery the NTP timestamps provided in the RTP NTP header extensions only, if such extensions are present for the RTP flows. Section 4 gives further details about the timestamp-based decoding order recovery.

Note: Using the RTP header extensions described above allows the receiver to find the corresponding sample of the layered media, or parts thereof, in all RTP flows at the instant the RTP header extension is inserted into the flows. This guarantees that any clock skew present in the NTP timestamp generation process based on RTCP sender reports is avoided, and so allows direct comparison of NTP timestamps across multiple RTP flows. Furthermore, this approach solves the possible problem of clock skews identified for the NI-T mode as defined in [7]. To ensure the absence of clock skew, a header extension containing the NTP timestamp MUST be inserted into the RTP flows comprising a layered media stream at the same instant in each RTP flow. This may require the insertion of extra packets in some of the RTP flows, since in layered video codecs not all sampling instances may be present in all the flows. If such a header extension is included in all flows at a sampling time instance, the NTP timestamps for samples following in decoding order the NTP header insertion point can be constructed using the RTP timestamps and identical reference NTP timestamps in the NTP header extension in all RTP flows. It should be noted that the frequency of inserting the NTP header extension is crucial in presence of clock skew, since the points of insertion may be the only points for a receiver to start the decoding order recovery.

4.3. Timestamp based decoding order recovery

If parts or complete samples as result of a layered coding process are transported as different RTP flows as different RTP streams and/or as different RTP sessions, a decoding order recovery process is required to reorder the samples or parts of samples received. Such mechanism may be based on the NTP presentation timestamp which can be derived from the RTP timestamp using the NTP wallclock provided in the RTCP sender report packets. In order to guarantee

the exact NTP alignment, the RTP NTP header extension as defined in this memo in Section 3.2 allows the receiver to tune in before the reception of such a sender report if the header extension is earlier provided in the RTP flow or it may be the only way to allow correct decoding order recovery based on exact matching of NTP timestamps in case of the presence of clock skew in the clock used for generating the NTP wallclock.

Since typically for layered video codecs as, e.g. SVC [7], the decoding order is not equal to the presentation order of the media samples, media samples or parts of media samples cannot be simply ordered according to the presentation timestamp order. For this reason, if transporting media samples or parts of media samples of a layered, multi-view or multi description codec in different RTP flows, the following rules SHOULD be kept for sending such flows:

Note: The following rules are typically kept for layered audio codecs, which allows using the same algorithm for decoding order recovery of audio samples.

Terminology: Following the decoding order of RTP flows as described above, an RTP flow containing sample data which is required to be accessed and/or decoded before decoding a second sample data of another RTP flow is called a lower RTP flow with respect to the second RTP flow. A second RTP flow, which requires for the decoding process accessing and/or decoding the sample data of the lower RTP flow is called the higher RTP flow. The lowest RTP flow is the flow, which does not require the presence of any other data.

- o The decoding order of media samples or part of the media samples transported in different RTP flows SHOULD be derivable by any means. This can be accomplished, e.g. by using the mechanisms defined in [6] if the sample data or parts of the sample data are transported in different RTP sessions or by any other means.

- o For each two RTP flows the following rules SHOULD be true in order to allow decoding order recovery based on matching NTP timestamps present in the different RTP flows:

    1. The order of the RTP samples within an RTP flow is equal to the decoding order.

    2. A higher RTP flow contains all the same sampling instances of the lower RTP flow. A higher RTP flow may contain additional sampling instances.

Note: In some cases, it may be required to add packets in higher RTP flows in order to satisfy the second rule above. This may be

achieved by placing empty RTP packets (containing padding data only) or by other payload means as, e.g. the Empty NAL unit packet as defined in [7].

If a packet must be inserted for satisfying the above rule, the NTP timestamp of such an inserted packet MUST be set equal to the NTP timestamp of a packet of the access unit present in any lower RTP flow and the lowest RTP flow. This is easy to accomplish if the packet can be inserted at the time of the RTP flow generation, since the NTP timestamp must be the same for the inserted packet and the packet of the corresponding sample.

The above rules allow the receiver to process the data of the RTP flows as follows:

o Go through all received RTP flows starting with the highest RTP flow and aggregate the sample data or parts of the sample data with the same NTP timestamp in the order of RTP flows, starting from the lowest RTP flow up to the highest RTP flow received, to the sample with the NTP timestamp present in the highest RTP flow. The NTP timestamps MAY be derived using RTCP sender reports or MAY be directly taken from the NTP header extension. The order of RTP flows may e.g. be indicated by mechanisms as defined in [6] or any other implicit or explicit means. Repeat the aforementioned process for each different NTP timestamp present in the highest RTP flow.

Informative example: The example shown in Figure 3 refers to three RTP flows A, B and C containing a layered, a multi-view or a multi-description media stream. In the example, the dependency signalling as defined in [6] indicates that flow A is the lowest RTP flow, B is the first higher RTP flow and depends on A, and C is the second higher RTP flow corresponding to flow A and depends on A and B. A picture coding prediction structure is used that results in samples present in higher flows but not present in all lower flows. Flow A has the lowest frame rate and Flow B and C have the same but higher frame rate. The figure shows parts of video samples contained in RTP packets which are stored in the de-jittering buffer at the receiver for de-packetization. The parts of the video samples are already re-ordered according to their RTP sequence number order. The figure indicates for the received sample parts the decoding order within the sessions, as well as the associated media (NTP) timestamps ("TS[..]"). Parts share the same media timestamp TS, which is shown at the bottom of the figure. Note that the timestamps are not in increasing order since, in this example, the decoding order is different from the output/display order.

The process first proceeds to the sample parts associated with the

first media timestamp TS[1] present in the highest flow C and removes/ignores all preceding (in decoding order) sample parts to sample parts with TS[1] in each of the de-jittering buffers of RTP flows A, B, and C. Then, starting from flow C, the first media timestamp available in decoding order (TS [1]) is selected and sample parts starting from RTP flow A, and flow B and C are placed in order of the RTP flow dependency as indicated by mechanisms defined in [6] (in the example for TS[1]: first flow B and then flow C into the video sample AU(TS[1]) associated with media timestamp TS[1]. Then the next media timestamp TS[3] in order of appearance in the highest RTP flow C is processed and the process described above is repeated. Note that there may be video samples with no sample parts present, e.g., in the lowest RTP flow A (see, e.g., TS[1]). With TS[8], the first video sample with sample parts present in all the RTP flows appears in the buffers.

```
C: ------------(1)----(2)---(3)---(4)----(5)--(6)---(7)----(8)----
     |     |     |     |     |     |      |    |     |      |
B: -(1)---(2)--(3)----(4)---(5)---(6)----(7)--(8)---(9)---(10)----
     |     |                 |     |                 |      |
A: -------(1)---------------(2)---(3)---------------(4)----(5)----
-------------------------------------------------decoding order-->

TS: [4]   [2]   [1]   [3]   [8]   [6]   [5]   [7]   [12]   [10]


Key:
A, B, C               - RTP sessions
Integer values in "()"- Video sample/part of video sample decoding
                        order within RTP session
"|"                   - indicates corresponding samples / parts of
                        sample of the same video sample AU(TS[..])
                        in the RTP flows
Integer values in "[]"- media timestamp TS, sampling time
                        as derived from the NTP timestamp associated
                        with the video sample AU(TS[..]), consisting
                        of sample parts in the sessions above.
```

5. Security Considerations

The security considerations of the RTP specification [1] and RTP/AVPF profile [2] apply. No additional security considerations apply due to the RTP/AVPF rapid resynchronisation mechanism defined in Section 3.1.

6. IANA Considerations

(tbd - this needs to register the new RTP/AVPF transport layer feedback packet type)

7. Acknowledgements

This memo has benefitted from discussions with numerous members of the IETF AVT working group, including Magnus Westerlund, Randell Jesup, Jonathan Lennox, Gerard Babonneau, Ingemar Johansson, and Roni Even. The header extension format of Variant A in Section 3.2 was suggested by Dave Singer, matching a similar mechanism specified by ISMA.

8. References

8.1. Normative References

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

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

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

[4] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

[5] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation", RFC 1305, March 1992.

[6] Schierl, T. and S. Wenger, "Signaling media decoding dependency in Session Description Protocol (SDP)", draft-ietf-mmusic-decoding-dependency-05 (work in progress), November 2008.

8.2. Informative References

[7] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for SVC Video", draft-ietf-avt-rtp-svc-16 (work in progress), December 2008.

[8] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, July 2003.

[9] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-mmusic-ice-19 (work in progress), October 2007.

[10] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for Secure Real-time Transport Protocol (SRTP)", draft-ietf-avt-dtls-srtp-05 (work in progress), September 2008.

[11] Zimmermann, P., Johnston, A., and J. Callas, "ZRTP: Media Path Key Agreement for Secure RTP", draft-zimmermann-avt-zrtp-13 (work in progress), January 2009.

Authors' Addresses

Colin Perkins
University of Glasgow
Department of Computing Science
Sir Alwyn Williams Building
Lilybank Gardens
Glasgow G12 8QQ
UK

Email: csp@csperkins.org

Thomas Schierl
Fraunhofer HHI
Einsteinufer 37
D-10587 Berlin
Germany

Phone: +49-30-31002-227
Email: thomas.schierl@hhi.fraunhofer.de