Offer/Answer Considerations for G.723 Annex A and G.729 Annex B

draft-muthu-payload-offer-answer-g723-g729-00

Abstract

[RFC4856] describes the annexa parameter for G723 and the annexb parameter for G729, G729D and G729E. However, the specification does not describe the offerer and answerer behavior when the value of the annexa or annexb parameter does not match in the SDP offer and answer. This document provides the offer/answer considerations for these parameters.

Status of this Memo

This Internet-Draft is submitted to IETF 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 August 27, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ......................................................... 3
2. Terminology .......................................................... 3
3. Offer/Answer Considerations ........................................ 4
    3.1. Offer/Answer Considerations for G723 Annex A .............. 4
    3.2. Offer/Answer Considerations for G.729 Annex B, G.729D
         Annex B and G.729E Annex B .................................. 4
4. Examples ............................................................... 5
    4.1. Offer with G279 annexb=yes and answer with G279
         annexb=no ....................................................... 5
    4.2. Offer with G279 annexb=yes and answer with G729 and no
         annexb parameter .............................................. 6
    4.3. Offer with G279 and no annexb parameter and answer
         with G729 annexb=no ............................................ 6
5. Security Considerations ............................................... 7
6. IANA Considerations .................................................. 7
7. Acknowledgement ..................................................... 7
8. Normative References ................................................ 7
Authors’ Addresses ....................................................... 8
1. Introduction

[RFC4856] describes the annexa parameter for G723 as follows:

annexa: indicates that Annex A, voice activity detection, is used or preferred. Permissible values are "yes" and "no" (without the quotes); "yes" is implied if this parameter is omitted.

Also, [RFC4856] describes the annexb parameter for G729, G729D and G729E as follows:

annexb: indicates that Annex B, voice activity detection, is used or preferred. Permissible values are "yes" and "no" (without the quotes); "yes" is implied if this parameter is omitted.

However, it does not have any normative statement for the case where the value of this parameter does not match in the SDP offer and answer. For example, if the offer has G729 with annexb=yes and the answer has G729 with annexb=no, it can be interpreted in two different ways:
- The offerer and answerer proceed as if G729 is negotiated with annexb=yes.
- The offerer and answerer proceed as if G729 is negotiated with annexb=no.

Since [RFC4856] does not state it clearly, various implementations have interpreted the offer/answer in their own ways, resulting in a different codec parameter being chosen to call failure, when the parameter value does not match in the offer and answer.


This document describes the offer/answer considerations for these parameters and provides the necessary clarifications.

2. Terminology

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. Offer/Answer Considerations

[RFC3551] states that

Receivers MUST accept comfort noise frames if restriction of their use has not been signaled. The MIME registration for G729 in RFC 3555 specifies a parameter that MAY be used with MIME or SDP to restrict the use of comfort noise frames.

Based on the above it is best to not use comfort noise frames if the SDP offer or answer indicates no support for it.

3.1. Offer/Answer Considerations for G723 Annex A

When the offer or answer has G723 and the annexa parameter is absent, it MUST be considered as if the offer or answer has G723 with annexa=yes.

When the offer has G723 with annexa=yes and the answer has G723 with annexa=no, the offerer and answerer MUST proceed as if G723 is negotiated with annexa=no.

When the offer has G723 with annexa=no then the answer MUST NOT have annexa=yes for G723. Thus the annexa parameter can be turned off by the answerer, but cannot be turned on.

Open item: Should the above be restated as follows?

When the offer has G723 with annexa=no then the answer MUST have annexa=no for G723.

This is technically correct, but are there implementations that omit the annexa parameter in answer and expect the least common denominator to be used?

When the offer has G723 with no annexa parameter and the answer has G723 with annexa=yes, the offerer and answerer MUST proceed as if G723 is negotiated with annexa=yes.


In this section G729 represents any of G729 or G729D or G729E.

When the offer or answer has G729 and the annexb parameter is absent, it MUST be considered as if the offer or answer has G729 with annexb=yes.

When the offer has G729 with annexb=yes and the answer has G729 with annexb=no, the offerer and answerer MUST proceed as if G729 is
negotiated with annexb=no.

When the offer has G729 with annexb=no then the answer MUST NOT have annexb=yes for G729. Thus the annexb parameter can be turned off by the answerer, but cannot be turned on.

Open item: Should the above be restated as follows?
When the offer has G729 with annexb=no then the answer MUST have annexb=no for G729.
This is technically correct, but are there implementations that omit the annexb parameter in answer and expect the least common denominator to be used?

When the offer has G.729 with no annexb parameter and the answer has G.729 with annexb=yes, the offerer and answerer MUST proceed as if G.729 is negotiated with annexb=yes.

4.  Examples

4.1.  Offer with G279 annexb=yes and answer with G279 annexb=no

[Offer with G279 annexb=yes]

v=0
o=alice 2890844526 2890844526 IN IP4 host.atlanta.example.com
s=
c=IN IP4 host.atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 18
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=yes

[Answer with G279 annexb=no]

v=0
o=bob 1890844326 1890844326 IN IP4 host.bangalore.example.com
s=
c=IN IP4 host.bangalore.example.com
t=0 0
m=audio 19140 RTP/AVP 18
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=no

In the above example the offerer and answerer proceed as if G729 is negotiated with annexb=no.
4.2. Offer with G279 annexb=yes and answer with G729 and no annexb parameter

[Offer with G279 annexb=yes]

v=0
o=alice 2890844526 2890844526 IN IP4 host.atlanta.example.com
s=
c=IN IP4 host.atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 18
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=yes

[Answer with G729 and no annexb parameter]

v=0
o=bob 1890844326 1890844326 IN IP4 host.bangalore.example.com
s=
c=IN IP4 host.bangalore.example.com
t=0 0
m=audio 19140 RTP/AVP 18
a=rtpmap:18 G729/8000

In the above example the offerer and answerer proceed as if G729 is negotiated with annexb=yes.

4.3. Offer with G279 and no annexb parameter and answer with G729 annexb=no

[Offer with G279 and no annexb parameter]

v=0
o=alice 2890844526 2890844526 IN IP4 host.atlanta.example.com
s=
c=IN IP4 host.atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 18
a=rtpmap:18 G729/8000
In the above example the offerer and answerer proceed as if G729 is negotiated with annexb=no.

5. Security Considerations

There is no extra security required apart from what is described in [RFC4856].

6. IANA Considerations

There is no IANA consideration for this draft.

7. Acknowledgement

Thanks to Flemming Andreasen (Cisco), Paul Kyzivat, Kevin Riley (Sonus), Ashish Sharma (Sonus) for their valuable comments and inputs.

8. Normative References


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

Authors’ Addresses

Muthu Arul Mozhi Perumal
Cisco Systems
Cessna Business Park
Sarjapur-Marathahalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: mperumal@cisco.com

Parthasarathi Ravindran
Sonus Networks
Prestige Shantiniketan – Business Precinct
Whitefield Road
Bangalore, Karnataka 560066
India

Email: pravindran@sonusnet.com
Abstract

This document describes use cases for ITU-T Recommendation G.711.0 compression of ITU-T Recommendation G.711 payloads when deployed in transport system segments using the Real-Time Transport Protocol (RTP).

ITU-T Rec. G.711.0 defines a lossless and stateless compression for G.711 packet payloads typically used in IP networks. Although the use of ITU-T Rec. G.711.0 can be negotiated end-to-end, being lossless and stateless it can also be applied as a compression mechanism "in-the-middle" of an end-to-end ITU-T G.711 negotiated session. These "in-the-middle" applications of ITU-T Rec. G.711.0 are called "G.711.0 Compression Segments" in this document.

This document outlines considerations and best practices (a.k.a. use cases) for these "G.711.0 compression segments".

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 September 1, 2012.
Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction . . . . . . . . . . . . . . . . . . . . . . . . . 3
2. Requirements Language . . . . . . . . . . . . . . . . . . . . 4
3. ITU-T Rec. G.711.0 Background and Use in RTP . . . . . . . . . 5
   3.1. G.711.0 Codec Background . . . . . . . . . . . . . . . . . 5
   3.2. G.711.0 RTP Background . . . . . . . . . . . . . . . . . . 6
4. G.711.0 "In The Middle" - Media Issues . . . . . . . . . . . . 8
   4.1. G.711.0 "In The Middle" - No RTP Header Compression . . . 8
   4.2. G.711.0 "In The Middle" - With RTP Header Compression . 11
   4.3. G.711.0 "In The Middle" - Implications for Voice Quality and Added Delay . . . . . . . . . . . . . . . . . . . . . . . 12
   4.4. G.711.0 "In The Middle" - Multiplexing Multiple G.711 Flows . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
5. G.711.0 "In The Middle" - Signaling Issues . . . . . . . . . . . 14
6. Acknowledgements . . . . . . . . . . . . . . . . . . . . . . . 17
7. Contributors . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
8. IANA Considerations . . . . . . . . . . . . . . . . . . . . . . . 19
9. Security Considerations . . . . . . . . . . . . . . . . . . . . . 20
10. References . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21
    10.1. Normative References . . . . . . . . . . . . . . . . . . . 21
    10.2. Informative References . . . . . . . . . . . . . . . . . . 22
Authors’ Addresses . . . . . . . . . . . . . . . . . . . . . . . . . . 23
1. Introduction


This document outlines considerations and best practices for when ITU-T Rec. G.711.0 is used as a compression mechanism on one or more segments of an end-to-end ITU-T Rec. G.711 [G.711] negotiated session using the Real-Time Transport Protocol (RTP, [RFC3550]). Because RTP payload types (PT) for G.711 PCMU (0) and PCMA (8) are static PTs and because G.711.0 is both lossless and stateless, G.711.0-based compression can often times be employed on intermediate segments without access to session signaling. These properties allow G.711.0-based bandwidth savings without modifications to G.711 endpoints or G.711 call processing systems. Additionally, due to the lossless property of G.711.0, it may be employed multiple times on an end-to-end G.711 session with no loss of voice quality relative to G.711.

ITU-T Rec. G.711.0 and ITU-T Rec. G.711 may be referred to in this document simply as G.711.0 and G.711, respectively.
2. Requirements Language

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 [RFC2119].
3. ITU-T Rec. G.711.0 Background and Use in RTP

This document describes the use of ITU-T Rec. G.711.0 when it is employed as a lossless compression mechanism somewhere in between end systems where:

1) at least one of the media processing end systems has negotiated use of G.711, and

2) RTP (see RFC 3550 [RFC3550] and RFC 3551 [RFC3551]) is employed.

When used in this way, the G.711 payloads resulting from decompression and the corresponding G.711 RTP headers should "appear" to the media processing end systems that have negotiated G.711 as having been transported transparently (more detail later in Section 4.2 (Section 4.2)). This use case is referred to herein as "G.711.0 in-the-middle".

This section briefly describes ITU-T Rec. G.711.0 and its use in RTP.

3.1. G.711.0 Codec Background

ITU-T Rec. G.711.0 is a lossless and stateless compression mechanism for ITU-T Recommendation G.711 [G.711] and thus is not a "codec" in the sense of "lossy" codecs typically carried by RTP. When ITU-T Rec. G.711.0 is negotiated end-to-end as if it were a codec, the understanding is that ITU-T Rec. G.711.0 is losslessly encoding the underlying (lossy) ITU-T Rec. G.711 pulse code modulation (PCM) sample representation of an audio signal. For this reason ITU-T Rec. G.711.0 will be interchangeably referred to in this document as a "lossless data compression algorithm" or a "codec", depending on context. ITU-T Rec. G.711 and ITU-T Rec. G.711.0 will be referred to as G.711 and G.711.0, respectively. Likewise, within this document, individual G.711 PCM samples will be referred to as "G.711 symbols" or just "symbols" or "samples" for brevity.

When ITU-T Rec. G.711.0 is negotiated end-to-end as a codec, it is negotiated similarly to ITU-T Rec. G.711 and its RTP payload format specification is nearly identical to ITU-T Rec. G.711. This end-to-end use of ITU-T Rec. G.711.0 and the payload format for it is documented in the G.711.0 RTP payload format Internet Draft [I-D.ramalho-payload-g7110].

The fundamental design of G.711.0 resulted from the desire to losslessly encode and compress frames of G.711 symbols independent of what types of signals those G.711 frames contained. The primary G.711.0 use case is for G.711 encoded, zero-mean, acoustic signals...
(such as speech and music), and for virtually every real-world signal significant compression is obtained. However, a significant property for G.711.0 is that it is lossless for any valid G.711 payload — even if the payload consisted of random G.711 symbols (many G.711-encoded modem or FAX payloads appear random). For this reason G.711.0 can be applied as a compression mechanism for any VoIP payload containing G.711 symbols without special consideration if those G.711 symbols came from FAX, modem, TTY or other "non-voice" applications. For detailed properties of the G.711.0 codec, see Section 3.2 of [I-D.ramalho-payload-g7110].

G.711.0, being both lossless and stateless, can be employed multiple times (e.g., on multiple, individual hops or series of hops) of a given flow with no degradation of quality relative to end-to-end G.711. Stated another way, multiple "lossless transcodes" from/to G.711.0/G.711 do not negatively affect voice quality as usually occurs with lossy transcodes to/from dissimilar codecs.

3.2. G.711.0 RTP Background

We note here that ITU-T Rec. G.711 is virtually always negotiated in RTP with a Payload Type (PT) of 0 (PCMU) or 8 (PCMA) because G.711 has a static payload type assignment and use of that static assignment is generally preferred. Payload types 0 and 8 should not be dynamically assigned to other codecs unless all dynamic payload types and unassigned payload types are already in use. Therefore, we have not seen RTP payload type 0 or 8 used in any network administrative domain for carrying other than a G.711 payload.

For this reason RTP packets with payload type 0 or 8 can usually be assumed to be PCMU or PCMA in most RTP settings. Thus compression from a G.711 payload to a G.711.0 payload can occur by:

1) noting payload type 0 or 8 is in use (input is assumed to be G.711), and

2) checking the input payload size in octets to ensure that it is an integer multiple of 40 (G.711.0 compression requires an integer multiple of 40 G.711 samples).

Because the lossless property of G.711.0, even if the payload presented to the G.711.0 encoder was not G.711, the G.711.0 lossless decoding process would produce the same exact payload as the payload input to the G.711.0 encoder.

Thus, due to the lossless property of G.711.0, network elements MAY apply G.711.0 compression to RTP payloads with payload type 0 or 8 (after check #2 above) and transport the payload as a G.711.0 payload.
- knowing that upon decompression the same G.711 input payload would be output from the G.711.0 decoder.

Because G.711.0 may be employed (as a payload compression mechanism) on any hop or hops of an end-to-end G.711 flow and payload types of 0 or 8 can reasonably be assumed to be G.711, neither Session Description Protocol (SDP, RFC 4566) [RFC4566] signaling elements nor specific G.711.0 negotiation mechanisms will be mandated by this RFC. While this is true, SDP descriptions in the G.711.0 RTP Payload Format Internet Draft [I-D.ramalho-payload-g7110] MAY be used for a G.711.0 "in-the-middle" negotiation such as may occur in Session Border Controllers (SBCs) and the like; these cases are described below.

The following section describes media issues for these "G.711.0 in-the-middle" use cases. The section following that section, Section 5 (Section 5), describes signaling implications for these "G.711.0 in-the-middle" use cases.
4. G.711.0 "In The Middle" - Media Issues

When G.711 has been negotiated end-to-end, G.711.0 can be employed by entities in the middle of the end-to-end G.711 flow as a compression mechanism. When used in this manner, this payload compression may be used with or without compression of the RTP header (e.g., cRTP [RFC2508], [RFC5795]). In either case, the G.711 payloads AND the corresponding G.711 RTP headers MUST appear to the end systems as having been transported transparently.

4.1. G.711.0 "In The Middle" - No RTP Header Compression

This figure below illustrates how the compression could be accomplished without RTP header compression.
Figure 1 depicts G.711.0 compression in Box C and G.711.0 decompression back to G.711 in Box E. This figure depicts the case where only compression of the G.711 payload is desired; the RTP header (including any extensions) is simply copied with the exception that the G.711 payload type (the usual static PT of 0 or 8 is shown)
is replaced by a PT negotiated between Box C and Box E (depicted above as PT = Q).

Note that if there are no hops between Box C and E (i.e., no Box D), Figure 1 is equivalent to compression over a single link.

The compression segment represented by Box C, Box E and Box F is labeled a "G.711.0 compression segment" in the above figure.

Since G.711.0 is a lossless and stateless compression, there can be multiple such segments between the sending and receiving endpoints (not shown).

The G.711.0 compression and decompression (Box C and E) may reside in a variety of network elements such as, but not limited to, switches, routers, middleboxes (NATs/PATs, firewalls, session border controllers, transport acceleration devices) and is purposely not specified here.

In IP routed networks one cannot guarantee that the same physical element represented by Box C (the compressor) or Box E (the decompressor) will stay in the same IP routed path between Sending Endpoints A and Receiving Endpoint G. While this is true, we note that G.711.0 is a stateless compression and that as long as it is assured that some element in the topology can provide the functionality represented by Box C and Box E, the identical physical element need not be in the path. For example, if all ingress and egress routers in an enterprise WAN administrative domain had such functionality, it need not be the case that the same ingress or egress routers be traversed for every packet in the flow due to the statelessness of G.711.0-based compression.

In one possible design the payload type isn’t changed at all (i.e., Q = original PCUM or PCM A PT) because a G.711.0 payload in number of octets is guaranteed to be different than the original G.711 payload size; this allows a RTP decoding process knowing the number of G.711 symbols to expect in the payload to infer that a G.711.0 representation of a G.711 payload is present. The interested reader will note that while a G.711.0 payload representation is usually much smaller than the uncompressed G.711 payload, an "uncompressible payload" is actually one octet larger (see ITU-T Rec. G.711.0 [G.711.0] specification).

If the RTP payload type to use (Q) is negotiated via session signaling, the method by which G.711.0 compression segment endpoints negotiate that payload type is not mandated in this document. However, the SDP descriptions in the G.711.0 RTP Payload Format Internet Draft [I-D.ramalho-payload-g7110] MAY be used for such a
negotiation and this point is addressed in Section 5.

In designs where Q is other than the original PCMU or PCMA PT and is not negotiated via session signaling, Q MUST be outside of the range of dynamic PT assignment and it is RECOMMENDED that Q be chosen from a static PT that is known to never be assigned within the scope of the G.711.0 compression segment or from the range of unassigned PTs [RFC3551] that are otherwise known to be free from conflict within the system design. We note here that the PT Q should never be seen by the end systems nor by any element outside of the G.711.0 compression segment; the specification for the choice of Q here reflects an abundance of caution for the case where a rogue RTP packet is not successfully processed by Box E functionality.

If the RTP payload type to use (Q) is configured, the method by which the G.711.0 compression segment endpoints are configured is outside the scope of this document.

In another possible design, Box C and Box E might be configured to be tunnel endpoints in a design where functionality represented by Box C (and possibly Box E) are known to be in the end-to-end path. Such functionality is also outside the scope of this document.

There may be many "potential G.711.0 compression/decompression points" along the end-to-end G.711 flow; the mechanisms by which certain entities determine that they should perform G.711.0-based compression and decompression are outside the scope of this document.

G.711.0 payloads, like G.711 payloads, may be encrypted by encryption protocols such as the Secure Real-time Transport Protocol (SRTP) [RFC3711], however the mechanisms by which the keys are exchanged or negotiated are outside the scope of this document. Mechanisms such as those used when G.711 encryption is employed MAY be used. Additionally, the security considerations of using G.711.0 SHOULD be considered in these "G.711.0 in-the-middle" applications (see Internet Draft [I-D.ramalho-payload-g7110]).

Network elements such as firewalls, NATs, SBCs, etc. that may exist in the path of the G.711.0 packets (Box D). These elements may drop packets containing unexpected payload types; therefore these elements may need additional configuration and/or signaling knowledge to let the compressed G.711.0 packets through. Mechanisms to do this are also outside the scope of this document.

4.2. G.711.0 "In The Middle" - With RTP Header Compression

When it is desired to compress the G.711 header as well, the G.711.0 compression segment endpoints of the previous section have further
functionality by which they also compress the headers. However, this functionality and the negotiation of same is outside of the scope of this document.

We note here that if such RTP header compression functionality is employed, that the G.711 payloads AND the corresponding G.711 RTP headers MUST appear to the end systems as having been transported transparently. That is, RTP header fields such as sequence numbers and timestamps need not necessarily be identical, but the differences between the input and output fields should be such that the receiving end system "cannot tell" that they were modified (differences by a constant in modulo send timestamp units for example).

Any RTP header compression functionality SHOULD be stateless so as to minimize error propagation for lost packets to be consistent with the G.711.0 design goal of no error propagation due to lost packets (see G.711.0 RTP Payload Format Internet Draft [I-D.ramalho-payload-g7110] Attribute A3).

4.3. G.711.0 "In The Middle" - Implications for Voice Quality and Added Delay

G.711.0, being both lossless and stateless, can be employed multiple times on an end-to-end G.711 flow (e.g., on multiple, individual hops or series of hops). If RTP headers are not compressed or stateless RTP header compression is employed (as recommended in Section 4.2 (Section 4.2)), then there is no error propagation owing to a loss of a G.711.0 packet. That is, the impact of an individual packet drop of a G.711.0 RTP packet is identical to the impact of a drop of the corresponding G.711 RTP packet.

Stated another way, multiple "lossless transcodes" from/to G.711.0/ G.711 do not negatively affect voice quality as may occur with lossy transcodes to/from dissimilar codecs.

G.711.0 provides over 50% reduction in average payload size with exactly 0.0000% quality loss relative to G.711 [ICASSP].

For completeness we note that a G.711.0 encode/decode average complexity is 1 WMOPS (see Internet Draft [I-D.ramalho-payload-g7110] Section 3.2, Attribute A8). Given such low complexity, less than 1 ms of compression/decompression of additional delay per each G.711.0 compression segment is expected in most implementations.

4.4. G.711.0 "In The Middle" - Multiplexing Multiple G.711 Flows

G.711.0 may also be desired to multiplex the payloads of many G.711 channels into one "G.711.0 payload" in a multiplex RTP packet.
If all the G.711 channels to be multiplexed have the same number of G.711 symbols in each individual source G.711 payload, as is the case in many "G.711 VoIP trunks", a straightforward way to parse the G.711.0 payload into individual G.711 payloads would be the methodology described in Section 4.2.2 in the G.711.0 Payload Format Internet Draft [I-D.ramalho-payload-g7110]. While this is possible, there are subtleties to such an approach such as what to do when the ith channel is unavailable due to an input packet drop. A straightforward way to address this issue is to have a dynamic mapping carried in side information, such as a RTP header extension, which has the capability to add or drop channels "on-the-fly".

Alternatively, specialized tunneling mechanisms, such as WAN optimization tunneling, can be used to convey such dynamic mapping between input and output G.711 channels.

Owing to these architectural options, the specification of such mechanisms is outside the scope of this document.

Similarly to Section 4.2 (Section 4.2) above, the de-multiplexing process producing individual G.711 output RTP packets MUST produce G.711 RTP headers that appear to the end systems as having been transported transparently. The mechanisms used are also outside the scope of this document.

Any RTP header multiplexing functionality SHOULD be stateless so as to minimize error propagation for lost packets to be consistent with the G.711.0 design goal of no error propagation due to lost packets (G.711.0 RTP Payload Format Internet Draft [I-D.ramalho-payload-g7110] Attribute A3).
5. G.711.0 "In The Middle" - Signaling Issues

This section describes a G.711.0 use case in which G.711 is negotiated end-to-end and:

1) there exists one or more places in the end-to-end path where the media is terminated and re-initiated, and

2) one or more of these "media segments" contained in the end to end path desires to compress the G.711 payloads to G.711.0 format (or vice versa).

Session Border Controllers (SBCs) and Media Termination Points (MTPs) are two examples of places where this could occur.

Figure 2 provides an illustration for such a case where SBCs are used as an example. This figure also assumes, as an example, that the Session Initiation Protocol (SIP, [RFC3261]) is used to set up the session and SDP was employed to negotiate the end-to-end codec.
This figure represents an end-to-end session between the originating and terminating endpoints where there are two SBCs in the end-to-end path. The end-to-end path is depicted to be in three segments, labeled A, B and C, and without loss of generality the IP Networks in these segments are labeled administrative domain A, B and C (although A, B or C may be part of the same administrative domain).

Here we assume that G.711 is the preferred codec end-to-end. If all
points where media could be terminated had G.711.0 capability, it is
highly likely that G.711.0 would have been negotiated from the
originating endpoint to the terminating endpoint. The reason for
this is that G.711.0 losslessly conveys G.711 and G.711.0 would
likely have been preferred over G.711 in the original negotiation.

However, if one or more points where media could be terminated did
not have G.711.0 capability, the end-to-end call would likely have
been negotiated as G.711. This is the case depicted here. In this
case there is an opportunity for any segment to renegotiate G.711.0
on its segment, and compressing to/from G.711 on the segments that
remain G.711.

The RTP payload format for G.711.0 is in Internet Draft
[I-D.ramalho-payload-g7110]. Specifically we note that the payload
format for G.711.0 is nearly identical to G.711 in that the timestamp
units are identical and most other elements are also identically
assigned (the major exception is the PT). Thus, the G.711.0 payload
format makes it trivial simply to change the PT and the payload of a
G.711 RTP packet to a G.711.0 packet and the converse. In other
words, the compression of the payload and the translation of the RTP
headers to/from G.711/G.711.0 may be performed "on-the-fly" in the
middle of an end-to-end G.711 session with no voice quality
degradation (relative to G.711) and concurrently obtaining all the
compression benefits of G.711.0.

For this case, the SDP parameters defined in the G.711 Payload Format
specification MAY be used for renegotiation to G.711.0 by any SIP UA
facing one of these segments without signaling these changes end-to-
end. We note here that SBCs also have signaling functionality and
are typically implemented as SIP Back-to-Back User Agents (B2BUAs).
From the perspective of the Originating Endpoint, the SIP signaling
termination is the UA at the first SBC (i.e., not the Terminating
Endpoint). Thus, for the case where G.711.0 is renegotiated only on
Segment A, the signaling for that codec change is not propagated to
the Terminating Endpoint. The end result is that the terminating
endpoint "believes" it is participating in an end-to-end G.711
session and the resulting voice quality is identical to that of an
end-to-end G.711 session (i.e., there is no need for it to know that
a lossless compression had taken place).
6.  Acknowledgements

There have been many people contributing to G.711.0 in the course of its development. The people listed here deserve special mention: Takehiro Moriya, Claude Lamblin, Herve Taddei, Simao Campos, Yusuke Hiwasaki, Jacek Stachurski, Lorin Netsch, Paul Coverdale, Patrick Luthi, Lei Miao, Paul Barrett, Jari Hagqvist, Pengjun (Jeff) Huang, and Jon Gibbs.
7. Contributors

The authors thank everyone who have contributed to this document. The people listed here deserve special mention: Paul, Jones, Ali Begen, and Roni Even.
8. IANA Considerations

This RFC is informational and contains no elements requiring IANA registration.
9. Security Considerations

This RFC is informational. The security considerations surrounding the use of G.711.0 (including uses described in this document) are described in the security considerations section of the G.711.0 RTP Payload Format Internet Draft [I-D.ramalho-payload-g7110].
10. References

10.1. Normative References


[I-D.ramalho-payload-g7110] Ramalho, M., Jones, P., Harada, N., Perumal, M., and M. Lei, "RTP Payload Format for G.711.0", draft-ramalho-payload-g7110-00 (work in progress), June 2011.


10.2. Informative References


Authors’ Addresses

Michael A. Ramalho (editor)
Cisco Systems, Inc.
4563 Tuscana Drive
Sarasota, FL  34241
USA

Phone:  +1 919 476 2038
Email:  mramalho@cisco.com

Dan Wing
Cisco Systems, Inc.
821 Alder Drive
Milpitas, CA  95035
USA

Phone:  +1 408 853 4197
Email:  dwing@cisco.com

Muthu Arul Mozhi Perumal
Cisco Systems, Inc.
Cessna Business Park
Sarjapur-Marathahalli Outer Ring Road
Bangalore, Karnataka  560103
India

Phone:  +91 9449288768
Email:  mperumal@cisco.com

Noboru Harada
NTT Communications Science Labs
3-1 Morinosato-Wakamiya
Atsugi, Kanagawa  243-0198
JAPAN

Email:  harada.noboru@lab.ntt.co.jp
Hadriel Kaplan
Acme Packet
71 Thirs Avenue
Burlington, VT  01803
USA

Email:  hkaplan@acme.com
RTP Payload Format for High Efficiency Video Coding

draft-schierl-payload-rtp-h265-00.txt

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on August 27, 2012.

Copyright and License Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.
Abstract

This memo describes an RTP payload format for High Efficiency Video Coding (HEVC) [HEVC], which is currently being developed by the Joint Collaborative Team on Video Coding (JCT-VC). The RTP payload format allows for packetization of one or more Network Abstraction Layer (NAL) units in each RTP packet payload, as well as fragmentation of a NAL unit into multiple RTP packets. Furthermore, it supports transmission of an HEVC stream over a single as well as multiple RTP flows. The payload format has wide applicability in videoconferencing, Internet video streaming, and high bit-rate entertainment-quality video, among others.
6. De-Packetization Process ........................................ 29
   6.1 Non-Interleaved Mode ........................................ 30
   6.2 Interleaved Mode ............................................ 30
       6.2.1 Size of the De-interleaving Buffer .................. 30
       6.2.2 De-interleaving Process ................................ 31
   6.3 Additional De-Packetization Guidelines .................... 33
7. Payload Format Parameters ........................................ 33
   7.1 Media Type Registration ..................................... 34
   7.2 SDP Parameters .............................................. 39
       7.2.1 Mapping of Payload Type Parameters to SDP .......... 39
       7.2.2 Usage with the SDP Offer/Answer Model .............. 39
       7.2.3 Usage with SDP Offer/Answer Model .................... 40
       7.2.4 Usage in Declarative Session Descriptions .......... 40
       7.2.5 Signaling of Parallel Processing ..................... 40
   7.3 Examples .................................................... 41
   7.4 Parameter Set Considerations ................................ 41
8. Security Considerations ........................................... 41
9. Congestion Control ................................................ 41
10. IANA Consideration .............................................. 41
11. Informative Appendix: Application Examples .................. 41
   11.1 Introduction .............................................. 41
   11.2 Streaming ................................................ 41
   11.3 Videoconferencing (Unicast to MANE, Unicast to Endpoints) 41
   11.4 Mobile TV (Multicast to MANE, Unicast to Endpoint) ....... 41
12. Acknowledgements ................................................ 41
13. References ..................................................... 42
   13.1 Normative References ....................................... 42
   13.2 Informative References ..................................... 42
14. Authors’ Addresses................................................. 43
1. Introduction

1.1. The HEVC Codec

1.1.1 Overview

High Efficiency Video Coding [HEVC] is a forthcoming video coding standard under development by the Joint Collaborative Team on Video Coding (JCT-VC) formed by the ITU-T and ISO/IEC. It is reported to provide significantly coding efficiency gains over H.264 [H.264]. The standard will be found under ISO/IEC as ISO/IEC 23008-2, informally as MPEG H Part 2. ITU-T may decide soon on the final recommendation number.

H.264 and HEVC share a similar hybrid video codec design. Conceptually, both technologies include a video coding layer (VCL), and a network abstraction layer (NAL).

The VCL of HEVC includes a prediction stage that involves motion compensation and spatial intra-prediction, integer transforms applied to prediction residuals, and an entropy coding stage that uses an arithmetic coding. As in H.264, in-loop debllocking filtering is applied to the reconstructed picture.

An important difference of HEVC compared to H.264 is the coding structure within a picture. In HEVC each picture is divided into treeblocks of up to 64x64 luma samples. Treeblocks can be recursively split into smaller Coding Units (CUs) using a generic quad-tree segmentation structure. CUs can be further split into Prediction Units (PUs) used for intra- and inter-prediction and Transform Units (TUs) defined for transform and quantization. HEVC includes integer transforms for a number of TU sizes. HEVC also includes two new in-loop filters that may be applied after the debllocking filtering: Sample Adaptive Offset (SAO) and Adaptive Loop Filter (ALF).

On random accessibility provisioning, HEVC introduces besides Instantaneous Decoder Refresh (IDR) pictures a Clean Random Access (CRA) picture, which is similar to what has been conventionally called open Group-of-Pictures (GOP) intra picture. Compared to H.264 wherein a CRA picture may be signalled using a recovery point Supplemental Enhancement Information (SEI) message, in HEVC a distinct NAL unit type is used for indication of a CRA picture. Furthermore, HEVC specifies that a conforming bitstream may start with a CRA picture, compared to in H.264 a conforming must start with an IDR picture.
Temporal layer access (TLA) pictures were introduced in HEVC to indicate temporal layer switching points.

Predictively coded pictures can include uni-predicted and bi-predicted slices. The flexibility in creating picture coding structures is roughly comparable to H.264.

The VCL generates and consumes syntax structures designed to be adaptable to MTU sizes commonly found in IP networks, irrespective of the size of a coded picture. Picture segmentation is achieved through slices. A concept of "fine granularity slices" (FGS) is included that allows to create slice boundaries within a treeblock.

The Network Adaptation Layer (NAL) is responsible for information required to the decoding process of more than one slice, which are collected in parameter sets. A number of data structures not strictly required for the decoding process, but potentially helpful in decoding systems can be conveyed in data structures such as Supplementary Enhancement Information (SEI) messages, Access unit delimiters, and so on.

All the aforementioned MTU-sized (or smaller) data structures are available in the form of Network Adaptation Layer Units.

The single distinguishing difference between HEVC and H.264 with respect to the RTP payload format design is the availability of VCL-based coding tools that are specifically designed to enable processing on high-level parallel architectures. These tools are described below in sufficient detail to provide motivation for the parallel processing signaling support that is described in section 7.2.5.

1.1.2 Parallel Processing Support

The reportedly significantly higher computational demand of HEVC over H.264, in conjunction with the ever increasing video resolution (both spatially and temporally) required by the market, led to the adoption of VCL coding tools specifically targeted to allow for parallelization on the sub-picture level. That is, parallelization occurs, at the minimum, at the granularity of an integer number of treeblocks. The targets for this type of high-level parallelization are multicore CPUs and DSPs as well as multiprocessor systems. In a system design, to be useful, these tools require signaling support, which is provided in section 7.2.5 of this memo. This section provides a brief overview of the tools available in [HEVC]. This section is expected to be updated frequently as the HEVC draft evolves.
For parallelization, four picture partition strategies are available.

Regular slices are segments of the bitstream that can be reconstructed independently from other regular slices within the same picture (though there may still be interdependencies through loop filtering operations). Regular slices are the only tool that can be used for parallelization that is also available, in virtually identical form, in H.264. Regular slices based parallelization does not require much inter-processor or inter-core communication for motion compensation when decoding a predictively coded picture, which is typically much heavier than inter-processor or inter-core data sharing due to in-picture prediction), as slices are designed to be independently decodable. However, for the same reason, regular slices can require some coding overhead. Further, regular slices (in contrast to some of the other tools mentioned below) also serve as the key mechanism for bitstream partitioning to match MTU size requirements, due to the in-picture independence of regular slices and that each regular slice is encapsulated in its own NAL unit. In many cases, the goal of parallelization and the goal of MTU size matching can place contradicting demands to the slice layout in a picture. The realization of this situation led to the development of the more advanced tools mentioned below. This payload format does not contain any specific mechanisms aiding parallelization through regular slices.

Entropy slices, like regular slices, break entropy decoding dependencies but allow prediction (and filtering) to cross slice boundaries. Insofar, they can be used as a lightweight mechanism to parallelize the entropy decoding, without having impact on other decoding steps. The lightweightness comes from that though each entropy slice is encapsulated into its own NAL unit, it has a much shorter slice header as most of the slice header syntax elements are not present and must be inherited from the preceding full slice header. Due to the allowance of in-picture prediction between neighboring entropy slices within a picture, the required inter-processor/inter-core communication to enable in-picture prediction can be substantial. Due to the same reason, entropy slices cannot be used for MTU size matching. Entropy slices appear to be only useful for system architectures that execute the entropy decoding process on a multicore/multi-CPU architecture, but execute the remaining decoding functionality on dedicated signal processing hardware. At the time of writing, entropy slices are not included in any profile defined in draft HEVC. No support of entropy slices is included in this memo.
In Wavefront Parallel Processing, the picture is partitioned into rows of treeblocks. Entropy decoding and prediction are allowed to use data from treeblocks in other partitions. Parallel processing is possible through parallel decoding of rows of treeblocks, where the start of the decoding of a row is delayed by two treeblocks, so to ensure that data related to a treeblock above and to the right of the subject treeblock is available before the subject treeblock is being decoded. Using this staggered start (which appears like a wavefront when represented graphically), parallelization is possible with up to as many processors/cores as the picture contains treeblock rows. At the time of writing, the draft HEVC includes a mechanism to organize the coded bits of different treeblock rows to be friendly to a particular number of parallel processors/cores. For example, it is possible that coded bits of even numbers of treeblock rows (treeblock rows 0, 2, 4, ...) all come before coded bits of odd numbers of treeblock rows (treeblock rows 1, 3, 5, ...), such that the bitstream is friendly to two parallel processors/cores, though decoding of an earlier-coming treeblock row (e.g. treeblock row 2) refers to an later-coming treeblock row (e.g. treeblock row 1). Similarly as entropy slices, due to the allowance of in-picture prediction between neighboring treeblock rows within a picture, the required inter-processor/inter-core communication to enable in-picture prediction can be substantial. The wavefront parallel processing partitioning does not result into more NAL units compared to when it is not applied, thus wavefront parallel processing cannot be used for MTU size matching. At the time of writing, wavefront parallel processing is not included in any profile of draft HEVC. This memo does not specify support for it.

Tiles define horizontal and vertical boundaries that partition a picture into tile columns and rows. The scan order of treeblocks is changed to be local within a tile (in the order of a treeblock raster can of a tile), before decoding the top-left treeblock of the next tile in the order of tile raster scan of a picture. Similar to regular slices, tiles break in-picture prediction dependencies (including entropy decoding dependencies). However, they do not need to be included into individual NAL units (same as wavefront parallel processing in this regard), hence tiles cannot be used for MTU size matching. Each tile can be processed by one processor/core, and the inter-processor/inter-core communication required for in-picture prediction between processing units decoding neighboring tiles is limited to conveying the shared slice header in cases a slice is spanning more than one tile, and loop filtering related sharing of reconstructed samples and metadata. Insofar, tiles are less demanding in terms of memory bandwidth compared to WPP due to the in-picture independence between two neighboring partitions. Tiles are included in the (single) existing profile of
[EHVC] and the support in the context of this memo will be specified in section 7 of this memo.

The interaction between regular slices and tiles is simplified by constraints of the HEVC draft. Specifically, for each slice and tile, either or both of the following conditions must be fulfilled: 1) all coded blocks in a slice belong to the same tile; 2) all coded blocks in a tile belong to the same slice.

1.1.3 Parameter Sets

The parameter set concept is borrowed from [H.264]. In addition to Sequence Parameter Sets (SPS), carrying data valid to the whole video sequence, and Picture Parameter Sets (PPS), carrying information valid on a picture by picture base, the new Adaption Parameters Sets (APS) carries picture-adaptive information that is also valid on a picture by picture base but is expected to change (typically much) more frequently than the information in PPS.

1.1.4 NAL Unit Header

HEVC maintains the NAL unit concept of H.264 with modifications. HEVC uses a two-byte NAL unit header. Table 1 lists the allocation of NAL unit types for VCL NAL units and non-VCL NAL units.
Table 1. NAL unit types in HEVC

<table>
<thead>
<tr>
<th>Type</th>
<th>NAL Unit Name</th>
<th>NAL unit type class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unspecified</td>
<td>non-VCL</td>
</tr>
<tr>
<td>1</td>
<td>Coded slice of a non-IDR, non-CRA and non-TLA pictures</td>
<td>VCL</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>Coded slice of a TLA picture</td>
<td>VCL</td>
</tr>
<tr>
<td>4</td>
<td>Coded slice of a CRA picture</td>
<td>VCL</td>
</tr>
<tr>
<td>5</td>
<td>Coded slice of an IDR picture</td>
<td>VCL</td>
</tr>
<tr>
<td>6</td>
<td>Supplemental enhancement information (SEI)</td>
<td>non-VCL</td>
</tr>
<tr>
<td>7</td>
<td>Sequence parameter set</td>
<td>non-VCL</td>
</tr>
<tr>
<td>8</td>
<td>Picture parameter set</td>
<td>non-VCL</td>
</tr>
<tr>
<td>9</td>
<td>Access unit delimiter</td>
<td>non-VCL</td>
</tr>
<tr>
<td>10..11</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>Filler data</td>
<td>non-VCL</td>
</tr>
<tr>
<td>13</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>Adaptation parameter set</td>
<td>non-VCL</td>
</tr>
<tr>
<td>15..23</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>24..63</td>
<td>unspecified</td>
<td>non-VCL</td>
</tr>
</tbody>
</table>

The syntax and semantics of the NAL unit header are specified in [HEVC], but the essential properties of the NAL unit header are summarized below for convenience.

The first byte of the NAL unit header has the following format:

```
+---------------+
|0|1|2|3|4|5|6|7|
+---------------+
|F|N|   Type    |
+---------------+

```

The semantics of the components of the NAL unit type octets, as specified in [HEVC], are described briefly below. In addition to the name and size of each field, the corresponding syntax element name in [HEVC] is also provided.

F: 1 bit
forbidden_zero_bit. HEVC declares a value of 1 as a syntax violation. Note: the bit is wasted for compatibility with MPEG-2 transport systems.

N: 1 bit
nal_ref_flag. A value of 0 indicates that the content of the NAL unit is not used to reconstruct reference pictures for future
prediction. Such NAL units can be discarded without potentially damaging the integrity of the reference pictures. A value of 1 indicates that the decoding of the NAL unit is required to maintain the integrity of reference pictures or that the NAL unit contains a parameter set.

Type: 6 bits

nal_unit_type. This component specifies the NAL unit type as defined in Table 7-1 of [HEVC], and in Table 1 in this memo. For a reference of all currently defined NAL unit types and their semantics, please refer to Section 7.4.1 in [HEVC].

In NAL units specified by HEVC, the second octet in the NAL unit header is shown below.

```
+---------------+
|0|1|2|3|4|5|6|7|
+---------------+
| TID |    R    |
+---------------+
```

TID: 3 bits
temporal_id. This component indicates the temporal identifier of the NAL unit in the coded sequence. For IDR pictures or CRA pictures the value is 0. For TLA pictures the value of temporal_id must be greater than 0.

R: 5 bits
reserved_5 bits. Reserved bits for future extension (such as scalability and three-dimension video extensions). R MUST be equal to "00001" (in binary form). Decoders must ignore (i.e. remove from the bitstream and discard) NAL units with values of reserved_one_5bits not equal to ‘00001’.

This memo extends the semantics of F, N, and TID, as described in Section 4.2.

1.2. Overview of the Payload Format

This payload format defines the following processes required for transport of HEVC coded data over RTP [RFC3550]:

- Usage of RTP header with this payload format
- Packetization of HEVC coded NAL units into RTP packets
o Transmission of HEVC NAL units of the same bitstream within a
single RTP session or within multiple RTP sessions

o Payload format parameters to be used within the Session
Description Protocol (SDP) [RFC4566].

2. Conventions

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 BCP 14, RFC 2119
[RFC2119].

This specification uses the notion of setting and clearing a bit
when bit fields are handled. Setting a bit is the same as assigning
that bit the value of 1 (On). Clearing a bit is the same as
assigning that bit the value of 0 (Off).

3. Definitions and Abbreviations

3.1 Definitions

This document uses the terms and definitions of [HEVC]. Section
3.1.1 lists relevant definitions copied from [HEVC] for convenience.
Section 3.1.2 gives definitions specific to this memo.

3.1.1 Definitions from the HEVC Specification

access unit: A set of NAL units that are consecutive in decoding
order and contain exactly one coded picture. In addition to the
coded slice NAL units of the coded picture, the access unit may
also contain other NAL units not containing slices of the coded
picture. The decoding of an access unit always results in a
decoded picture.

coded video sequence: A sequence of access units that consists,
in decoding order, of an IDR access unit followed by zero or more
non-IDR access units including all subsequent access units up to
but not including any subsequent IDR access unit.

CRA access unit: An access unit in which the coded picture is a
CRA picture.

CRA picture: A coded picture containing only I slices and for
which each slice has nal_unit_type equal to 4; all coded pictures
that follow the Clean Random Access (CRA) picture both in
decoding order and output order shall not use inter prediction
from any picture that precedes the CRA picture either in decoding order or output order; and any picture that precedes the CRA picture in decoding order also precedes the CRA picture in output order.

IDR access unit: An access unit in which the coded picture is an IDR picture.

IDR picture: A coded picture for which the variable IdrPicFlag is equal to 1. An IDR picture causes the decoding process to mark all reference pictures as "unused for reference". All coded pictures that follow an IDR picture in decoding order can be decoded without inter prediction from any picture that precedes the IDR picture in decoding order. The first picture of each coded video sequence in decoding order is an IDR picture.

Random Access: The act of starting the decoding process for a bitstream at a point other than the beginning of the stream.

Tile: An integer number of treeblocks co-occurring in one column and one row (each of which comprising one or more columns or rows of treeblocks), ordered consecutively in treeblock raster scan of the tile. The division of each picture into tiles is a partitioning. Tiles in a picture are ordered consecutively in tile raster scan of the picture. Although a slice contains treeblocks that are consecutive in treeblock raster scan of a tile, these treeblocks are not necessarily consecutive in treeblock raster scan of the picture.

3.1.2 Definitions Specific to This Memo

media aware network element (MANE): A network element, such as a middlebox or application layer gateway that is capable of parsing certain aspects of the RTP payload headers or the RTP payload and reacting to their contents.

Informative note: The concept of a MANE goes beyond normal routers or gateways in that a MANE has to be aware of the signaling (e.g., to learn about the payload type mappings of the media streams), and in that it has to be trusted when working with SRTP. The advantage of using MANEs is that they allow packets to be dropped according to the needs of the media coding. For example, if a MANE has to drop packets due to congestion on a certain link, it can identify and remove those packets whose elimination produces the least adverse effect on the user experience. After dropping packets, MANEs...
must rewrite RTCP packets to match the changes to the RTP packet stream as specified in Section 7 of [RFC3550].

NAL unit decoding order: A NAL unit order that conforms to the constraints on NAL unit order given in Section 7.4.1.2.3 in [HEVC].

NALU-time: The value that the RTP timestamp would have if the NAL unit would be transported in its own RTP packet.

RTP packet stream: A sequence of RTP packets with increasing sequence numbers (except for wrap-around), identical PT and identical SSRC (Synchronization Source), carried in one RTP session. Within the scope of this memo, one RTP packet stream is utilized to transport one or more layers.

transmission order: The order of packets in ascending RTP sequence number order (in modulo arithmetic). Within an aggregation packet, the NAL unit transmission order is the same as the order of appearance of NAL units in the packet.

3.2 Abbreviations

TBD

4. RTP Payload Format

4.1 RTP Header Usage

The format of the RTP header is specified in [RFC3550] and reprinted in Figure 1 for convenience. This payload format uses the fields of the header in a manner consistent with that specification.

The RTP payload (and the settings for some RTP header bits) for aggregation packets and fragmentation units are specified in Sections 4.6 and 4.8, respectively.
The RTP header information to be set according to this RTP payload format is set as follows:

Marker bit (M): 1 bit

Set for the very last packet of the access unit indicated by the RTP timestamp, in line with the normal use of the M bit in video formats, to allow an efficient playout buffer handling. For aggregation packets (STAP), the marker bit in the RTP header MUST be set to the value that the marker bit of the last NAL unit of the aggregation packet would have been if it were transported in its own RTP packet. Decoders MAY use this bit as an early indication of the last packet of an access unit but MUST NOT rely on this property.

Informative note: Only one M bit is associated with an aggregation packet carrying multiple NAL units. Thus, if a gateway has re-packetized an aggregation packet into several packets, it cannot reliably set the M bit of those packets.

Payload type (PT): 7 bits

The assignment of an RTP payload type for this new packet format is outside the scope of this document and will not be specified here. The assignment of a payload type has to be performed either through the profile used or in a dynamic way.
Sequence number (SN): 16 bits

Set and used in accordance with RFC 3550. In some packetization modes (list TBD), the sequence number is used to determine decoding order for the NALUs.

Timestamp: 32 bits

The RTP timestamp is set to the sampling timestamp of the content. A 90 kHz clock rate MUST be used.

If the NAL unit has no timing properties of its own (e.g., parameter set and SEI NAL units), the RTP timestamp is set to the RTP timestamp of the coded picture of the access unit in which the NAL unit is included, according to Section 7.4.1.2.3 of [HEVC].

Receivers SHOULD ignore any picture timing SEI messages included in access units that have only one display timestamp. Instead, receivers SHOULD use the RTP timestamp for synchronizing the display process. If one access unit has more than one display timestamp carried in a picture timing SEI message, then the information in the SEI message SHOULD be treated as relative to the RTP timestamp, with the earliest event occurring at the time given by the RTP timestamp and subsequent events later, as given by the difference in picture time values carried in the picture timing SEI message. Let $t_{SEI1}$, $t_{SEI2}$, ..., $t_{SEIn}$ be the display timestamps carried in the SEI message of an access unit, where $t_{SEI1}$ is the earliest of all such timestamps. Let $tmadjst()$ be a function that adjusts the SEI messages time scale to a 90-kHz time scale. Let $TS$ be the RTP timestamp. Then, the display time for the event associated with $t_{SEI1}$ is $TS$. The display time for the event with $t_{SEIx}$, where $x$ is $[2..n]$, is $TS + tmadjst(t_{SEIx} - t_{SEI1})$.

4.2 NAL Unit Header Usage

The structure and semantics of the NAL unit header according to the HEVC specification [HEVC] were introduced in Section 1.1.4. This section specifies the extended semantics of the NAL unit header fields.

4.3 Payload Structures

The NAL unit structure is central to HEVC [HEVC], all HEVC coded bits for representing a video signal are encapsulated in NAL units. Therefore each RTP packet payload is structured as a NAL unit, which
contains one or a part of one NAL unit specified in HEVC, or aggregates one or more NAL units specified in HEVC.

4.4 Transmission Modes

This memo enables transmission of an HEVC bitstream over a single RTP session or multiple RTP sessions.

TBD: SSRC Muxing for video conf. + TV broadcast/multicast.

4.5 Packetization Modes

This memo specifies the following packetization modes:

- Non-interleaved mode
- Interleaved mode

In the non-interleaved mode, NAL units are transmitted in NAL unit decoding order. The interleaved mode allows transmission of NAL units out of NAL unit decoding order.

The packetization mode in use MAY be signaled by the value of the OPTIONAL packetization-mode media type parameter. The used packetization mode governs which NAL unit types are allowed in RTP payloads. Table 2 summarizes the allowed packet payload types for each packetization mode. Packetization modes are explained in more detail in section 6.

Table 2. Summary of allowed NAL unit types for each packetization mode (yes = allowed, no = disallowed, ig = ignore)

<table>
<thead>
<tr>
<th>Payload Type</th>
<th>Packet Type</th>
<th>Non-Interleaved Mode</th>
<th>Interleaved Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>reserved</td>
<td>ig</td>
<td>ig</td>
</tr>
<tr>
<td>1-23</td>
<td>NAL unit</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>24</td>
<td>STAP-A</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>25</td>
<td>STAP-B</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>26</td>
<td>FU-A</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>27</td>
<td>FU-B</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>28-63</td>
<td>reserved</td>
<td>ig</td>
<td>ig</td>
</tr>
</tbody>
</table>

Some NAL unit or payload type values (indicated as reserved in Table 2) are reserved for future extensions. NAL units of those types SHOULD NOT be sent by a sender (direct as packet payloads, or as aggregation units in aggregation packets, or as fragmented units...
in FU packets) and MUST be ignored by a receiver. For example, the payload types 1-23, with the associated packet type "NAL unit", are allowed in "Non-Interleaved Mode", but disallowed in "Interleaved Mode". However, NAL units of NAL unit types 1-23 can be used in "Interleaved Mode" as aggregation units in STAP-B packets as well as fragmented units in FU-A and FU-B packets. Similarly, NAL units of NAL unit types 1-23 can also be used in the "Non-Interleaved Mode" as aggregation units in STAP-A packets or fragmented units in FU-A packets, in addition to being directly used as packet payloads.

4.6 Decoding Order

In the interleaved packetization mode, the transmission order of NAL units is allowed to differ from the decoding order of the NAL units. Decoding order number (DON) is a field in the payload structure or a derived variable that indicates the NAL unit decoding order. Rationale and examples of use cases for transmission out of decoding order and for the use of DON are given in section 13.

The coupling of transmission and decoding order is controlled by the OPTIONAL sprop-interleaving-depth media type parameter as follows. When the value of the OPTIONAL sprop-interleaving-depth media type parameter is equal to 0 (explicitly or per default), the transmission order of NAL units MUST conform to the NAL unit decoding order. When the value of the OPTIONAL sprop-interleaving-depth media type parameter is greater than 0,

o the order of NAL units generated by de-packetizing STAP-Bs, and

FUs in two consecutive packets is NOT REQUIRED to be the NAL unit decoding order.

The RTP payload structures for an STAP-A, and an FU-A do not include DON. STAP-B and FU-B structures include DON.

Informative note: When an FU-A occurs in interleaved mode, it always follows an FU-B, which sets its DON.

Informative note: If a transmitter wants to encapsulate a single NAL unit per packet and transmit packets out of their decoding order, STAP-B packet type can be used.

In the non-interleaved packetization mode, the transmission order of NAL units in single NAL unit packets, STAP-As, and FU-As MUST be the same as their NAL unit decoding order. The NAL units within an STAP MUST appear in the NAL unit decoding order. Thus, the decoding order is first provided through the implicit order within a STAP,
and second provided through the RTP sequence number for the order between STAPs, FUs, and single NAL unit packets.

Signaling of the value of DON for NAL units carried in STAP-B, and a series of fragmentation units starting with an FU-B is specified in sections 4.7.1, and 4.8, respectively. The DON value of the first NAL unit in transmission order MAY be set to any value. Values of DON are in the range of 0 to 65535, inclusive. After reaching the maximum value, the value of DON wraps around to 0.

The decoding order of two NAL units contained in any STAP-B, or a series of fragmentation units starting with an FU-B is determined as follows. Let DON(i) be the decoding order number of the NAL unit having index i in the transmission order. Function don_diff(m,n) is specified as follows:

\[
\begin{align*}
\text{If } \text{DON}(m) &= \text{DON}(n), \text{don}_\text{diff}(m,n) = 0 \\
\text{If } (\text{DON}(m) < \text{DON}(n) \text{ and } \text{DON}(n) - \text{DON}(m) < 32768), \text{don}_\text{diff}(m,n) &= \text{DON}(n) - \text{DON}(m) \\
\text{If } (\text{DON}(m) > \text{DON}(n) \text{ and } \text{DON}(m) - \text{DON}(n) >= 32768), \text{don}_\text{diff}(m,n) &= 65536 - \text{DON}(m) + \text{DON}(n) \\
\text{If } (\text{DON}(m) < \text{DON}(n) \text{ and } \text{DON}(n) - \text{DON}(m) >= 32768), \text{don}_\text{diff}(m,n) &= -(\text{DON}(m) + 65536 - \text{DON}(n)) \\
\text{If } (\text{DON}(m) > \text{DON}(n) \text{ and } \text{DON}(m) - \text{DON}(n) < 32768), \text{don}_\text{diff}(m,n) &= -(\text{DON}(m) - \text{DON}(n))
\end{align*}
\]

A positive value of don_diff(m,n) indicates that the NAL unit having transmission order index n follows, in decoding order, the NAL unit having transmission order index m. When don_diff(m,n) is equal to 0, then the NAL unit decoding order of the two NAL units can be in either order. A negative value of don_diff(m,n) indicates that the NAL unit having transmission order index n precedes, in decoding order, the NAL unit having transmission order index m.

Values of the DON field MUST be such that the decoding order determined by the values of DON, as specified above, conforms to the NAL unit decoding order. If the order of two NAL units in NAL unit decoding order is switched and the new order does not conform to the NAL unit decoding order, the NAL units MUST NOT have the same value of DON. If the order of two consecutive NAL units in the NAL unit stream is switched and the new order still conforms to the NAL unit decoding order, the NAL units MAY have the same value of DON. Consequently, NAL units having the same value of DON can be decoded.
in any order, and two NAL units having a different value of DON should be passed to the decoder in the order specified above. When two consecutive NAL units in the NAL unit decoding order have a different value of DON, the value of DON for the second NAL unit in decoding order SHOULD be the value of DON for the first, incremented by one.

An example of the de-packetization process to recover the NAL unit decoding order is given in section 7.

Informative note: Receivers should not expect that the absolute difference of values of DON for two consecutive NAL units in the NAL unit decoding order will be equal to one, even in error-free transmission. An increment by one is not required, as at the time of associating values of DON to NAL units, it may not be known whether all NAL units are delivered to the receiver. For example, a gateway may not forward coded slice NAL units of non-reference pictures or SEI NAL units when there is a shortage of bit rate in the network to which the packets are forwarded. In another example, a live broadcast is interrupted by pre-encoded content, such as commercials, from time to time. The first intra picture of a pre-encoded clip is transmitted in advance to ensure that it is readily available in the receiver. When transmitting the first intra picture, the originator does not exactly know how many NAL units will be encoded before the first intra picture of the pre-encoded clip follows in decoding order. Thus, the values of DON for the NAL units of the first intra picture of the pre-encoded clip have to be estimated when they are transmitted, and gaps in values of DON may occur.

4.7 Aggregation Packets

Aggregation packets are the NAL unit aggregation scheme of this payload specification. The scheme is introduced to reflect the dramatically different MTU sizes of two key target networks: wireline IP networks (with an MTU size that is often limited by the Ethernet MTU size; roughly 1500 bytes), and IP or non-IP (e.g., ITU-T H.324/M) based wireless communication systems with preferred transmission unit sizes of 254 bytes or less. To prevent media transcoding between the two worlds, and to avoid undesirable packetization overhead, a NAL unit aggregation scheme is introduced.

The Single-time aggregation packet (STAP) is defined by this specification:
o Single-time aggregation packet (STAP): aggregates NAL units with identical NALU-time. Two types of STAPs are defined, one without DON (STAP-A) and another including DON (STAP-B).

Each NAL unit to be carried in an aggregation packet is encapsulated in an aggregation unit. The structure of the RTP payload format for aggregation packets is presented in Figure 2.

![RTP payload format for aggregation packets](image)

STAPs do have the following packetization rules: The type field of the NAL unit type octet MUST be set to the appropriate value for STAP, as indicated in Table 2. The F bit MUST be cleared if all F bits of the aggregated NAL units are zero; otherwise, it MUST be set. The value of NRI MUST be the maximum of all the NAL units carried in the aggregation packet.

The marker bit in the RTP header is set to the value that the marker bit of the last NAL unit of the aggregated packet would have if it were transported in its own RTP packet.

The payload of an aggregation packet consists of one or more aggregation units. See sections 4.7.1 for the single time aggregation unit. An aggregation packet can carry as many aggregation units as necessary; however, the total amount of data in an aggregation packet obviously MUST fit into an IP packet, and the size SHOULD be chosen so that the resulting IP packet is smaller than the MTU size. An aggregation packet MUST NOT contain fragmentation units specified in section 4.8. Aggregation packets MUST NOT be nested; i.e., an aggregation packet MUST NOT contain another aggregation packet.
4.7.1 Single Time Aggregation Packet (STAP)

Single-time aggregation packet (STAP) SHOULD be used whenever NAL units are aggregated that all share the same NALU-time. The payload of an STAP consists of at least one single-time aggregation unit, as presented in Figure 3. The payload of an STAP-B consists of a 16-bit unsigned decoding order number (DON) (in network byte order) followed by at least one single-time aggregation unit, as presented in Figure 4.

The DON field specifies the value of DON for the first NAL unit in an STAP-B in transmission order. For each successive NAL unit in appearance order in an STAP-B, the value of DON is equal to (the value of DON of the previous NAL unit in the STAP-B + 1) % 65536, in which ‘%’ stands for the modulo operation.
A single-time aggregation unit consists of 16-bit unsigned size information (in network byte order) that indicates the size of the following NAL unit in bytes (excluding these two octets, but including the NAL unit type octet of the NAL unit), followed by the NAL unit itself, including its NAL unit type byte. A single-time aggregation unit is byte aligned within the RTP payload, but it may not be aligned on a 32-bit word boundary. Figure 5 presents the structure of the single-time aggregation unit.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NAL unit size |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NAL unit |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Figure 5 Structure for single-time aggregation unit (STAU)
```

Figure 6 presents an example of an RTP packet that contains an STAP-A. The STAP-A contains two single-time aggregation units, labeled as 1 and 2 in the figure.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RTP Header |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| STAP NAL HDR | NALU 1 Size | NALU 1 HDR |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NALU 1 HDR | NALU 1 Data |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NALU 2 Size | NALU 2 HDR |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NALU 2 HDR | NALU 2 Data |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| : |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| :...OPTIONAL RTP padding |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
Figure 6 An example of an RTP packet including an STAP-A containing two single-time aggregation units

Figure 7 presents an example of an RTP packet that contains an STAP-B. The STAP contains two single-time aggregation units, labeled as 1 and 2 in the figure.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          RTP Header                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|STAP-B NAL HDR | DON                           | NALU 1 Size   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NALU 1 Size  | NALU 1 HDR                    | NALU 1 Data   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               | NALU 2 Size                   | NALU 2 HDR    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| NALU 2 HDR    |        NALU 2 Data                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               :...OPTIONAL RTP padding        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7 An example of an RTP packet including an STAP-B containing two single-time aggregation units

4.8 Fragmentation Units (FUs)

This payload type allows fragmenting a NAL unit into several RTP packets. Doing so on the application layer instead of relying on lower layer fragmentation (e.g., by IP) may have the following use cases:

- The payload format is capable of transporting NAL units bigger than 64 kbytes over an IPv4 network that may be present in pre-recorded video, particularly in High Definition formats (there is a limit of the number of slices per picture, which results in a limit of NAL units per picture, which may result in big NAL units).

- The fragmentation mechanism allows fragmenting a single NAL unit and applying generic forward error correction.
Fragmentation is defined only for a single NAL unit and not for any aggregation packets. A fragment of a NAL unit consists of an integer number of consecutive octets of that NAL unit. Each octet of the NAL unit MUST be part of exactly one fragment of that NAL unit. Fragments of the same NAL unit MUST be sent in consecutive order with ascending RTP sequence numbers (with no other RTP packets within the same RTP packet stream being sent between the first and last fragment). Similarly, a NAL unit MUST be reassembled in RTP sequence number order.

When a NAL unit is fragmented and conveyed within fragmentation units (FUs), it is referred to as a fragmented NAL unit. STAPs MUST NOT be fragmented. FUs MUST NOT be nested; i.e., an FU MUST NOT contain another FU.

The RTP timestamp of an RTP packet carrying an FU is set to the NALU-time of the fragmented NAL unit.

Figure 8 presents the RTP payload format for FU-A. An FU-A consists of a fragmentation unit indicator of one octet, a fragmentation unit header of one octet, and a fragmentation unit payload.

```
+---------------+---------------+---------------+---------------+
<table>
<thead>
<tr>
<th>FU    NAL HDR</th>
<th>FU header</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FU payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|                                           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                           :...OPTIONAL RTP padding |
|                                           +-----------------------------
```

Figure 8   RTP payload format for FU-A

Figure 9 presents the RTP payload format for FU-Bs. An FU-B consists of a fragmentation unit indicator of one octet, a fragmentation unit header of one octet, a decoding order number (DON) (in network byte order), and a fragmentation unit payload. In other words, the structure of FU-B is the same as the structure of FU-A, except for the additional DON field.
NAL unit type FU-B MUST be used in the interleaved packetization mode for the first fragmentation unit of a fragmented NAL unit. NAL unit type FU-B MUST NOT be used in any other case. In other words, in the interleaved packetization mode, each NALU that is fragmented has an FU-B as the first fragment, followed by one or more FU-A fragments.

The FU NAL HDR octet has the following format:

```
+---------------+
|0|1|2|3|4|5|6|7|
+---------------+

F|N|    Type   |
+---------------+
```

A value equal to 26 in the Type field of the FU indicator octet identifies an FU-A packet and a value of 27 identifies an FU-B packet. The use of the F bit is described in section 5. The value of the N field MUST be set according to the value of the N field in the fragmented NAL unit.

The FU header has the following format:

```
+---------------+
|0|1|2|3|4|5|6|7|
+---------------+

S|E|    Type   |
+---------------+
```

S: 1 bit
When set to one, the Start bit indicates the start of a
fragmented NAL unit. When the following FU payload is not the start of a fragmented NAL unit payload, the Start bit is set to zero.

E: 1 bit
When set to one, the End bit indicates the end of a fragmented NAL unit, i.e., the last byte of the payload is also the last byte of the fragmented NAL unit. When the following FU payload is not the last fragment of a fragmented NAL unit, the End bit is set to zero.

Type: 6 bits
The NAL unit payload type as defined in Table 7-1 of [HEVC].

The value of DON in FU-Bs is selected as described in section 4.6.

Informative note: The DON field in FU-Bs allows gateways to fragment NAL units to FU-Bs without organizing the incoming NAL units to the NAL unit decoding order.

A fragmented NAL unit MUST NOT be transmitted in one FU; i.e., the Start bit and End bit MUST NOT both be set to one in the same FU header.

The FU payload consists of fragments of the payload of the fragmented NAL unit so that if the fragmentation unit payloads of consecutive FUs are sequentially concatenated, the payload of the fragmented NAL unit can be reconstructed. The NAL unit type octet of the fragmented NAL unit is not included as such in the fragmentation unit payload, but rather the information of the NAL unit type octet of the fragmented NAL unit is conveyed in F and N fields of the FU indicator octet of the fragmentation unit and in the type field of the FU header. An FU payload MAY have any number of octets and MAY be empty.

If a fragmentation unit is lost, the receiver SHOULD discard all following fragmentation units in transmission order corresponding to the same fragmented NAL unit.

A receiver in an endpoint or in a MANE MAY aggregate the first n-1 fragments of a NAL unit to an (incomplete) NAL unit, even if fragment n of that NAL unit is not received. In this case, the forbidden_zero_bit of the NAL unit MUST be set to one to indicate a syntax violation.
5. Packetization Rules

The packetization modes are introduced in section 4.5. The packetization rules common to more than one of the packetization modes are specified in section 5.1. The packetization rules for the non-interleaved mode are specified in section 5.2, and the packetization rules for the interleaved mode are specified in sections 5.3.

5.1 Common Packetization Rules

All senders MUST enforce the following packetization rules regardless of the packetization mode in use:

- VCL NAL units belonging to the same coded picture (and thus sharing the same RTP timestamp value) SHOULD be sent in their original decoding order to minimize the delay. Note that the decoding order is the order of the NAL units in the bitstream.

- Parameter sets are handled in accordance with the rules and recommendations given in section 7.4.

- MANEs MUST NOT duplicate any NAL unit except for sequence or picture parameter set NAL units, as neither this memo nor the HEVC specification provides means to identify duplicated NAL units. Sequence and picture parameter set NAL units MAY be duplicated to make their correct reception more probable, but any such duplication MUST NOT affect the contents of any active sequence or picture parameter set. Duplication SHOULD be performed on the application layer and not by duplicating RTP packets (with identical sequence numbers).

Senders using the non-interleaved mode and the interleaved mode MUST enforce the following packetization rule:

- MANEs MAY convert single NAL unit packets into one aggregation packet, convert an aggregation packet into several single NAL unit packets, or mix both concepts, in an RTP translator. The RTP translator SHOULD take into account at least the following parameters: path MTU size, unequal protection mechanisms (e.g., through packet-based FEC according to [RFC5109], especially for sequence and picture parameter set NAL units and coded slice data partition A NAL units), bearable latency of the system, and buffering capabilities of the receiver.
Informative note: An RTP translator is required to handle RTCP as per [RFC3550].

5.2 Non-Interleaved mode

This mode MUST be supported. This mode is in use when the value of the OPTIONAL packetization-mode media type parameter is equal to 1. It is primarily intended for low-delay applications. Only single NAL unit packets, STAPs, and FUs MAY be used in this mode. The transmission order of NAL units MUST comply with the NAL unit decoding order.

5.3 Interleaved mode

This mode is in use when the value of the OPTIONAL packetization-mode media type parameter is equal to 2. Some receivers MAY support this mode. STAP-Bs, FU-As, and FU-Bs MAY be used. STAP-As and single NAL unit packets MUST NOT be used. The transmission order of packets and NAL units is constrained as specified in section 4.6.

6. De-Packetization Process

The de-packetization process is implementation dependent. Therefore, the following description should be seen as an example of a suitable implementation. Other schemes may be used as well as long as the output for the same input is the same as the process described below. The output is the same meaning that the number of NAL units and their order are both the identical. Optimizations relative to the described algorithms are likely possible. Section 6.1 presents the de-packetization process for the non-interleaved packetization mode and section 6.2 presents the de-packetization process for the interleaved packetization mode.

All normal RTP mechanisms related to buffer management apply. In particular, duplicated or outdated RTP packets (as indicated by the RTP sequences number and the RTP timestamp) are removed. To determine the exact time for decoding, factors such as a possible intentional delay to allow for proper inter-stream synchronization must be factored in.
6.1 Non-Interleaved Mode

The receiver includes a receiver buffer to compensate for transmission delay jitter. The receiver stores incoming packets in reception order into the receiver buffer. Packets are de-packetized in RTP sequence number order. If a de-packetized packet is a single NAL unit packet, the NAL unit contained in the packet is passed directly to the decoder. If a de-packetized packet is an STAP-A, the NAL units contained in the packet are passed to the decoder in the order in which they are encapsulated in the packet. For all the FU-A packets containing fragments of a single NAL unit, the de-packetized fragments are concatenated in their sending order to recover the NAL unit, which is then passed to the decoder.

6.2 Interleaved Mode

The general concept behind these de-packetization rules is to reorder NAL units from transmission order to the NAL unit decoding order.

The receiver includes a receiver buffer, which is used to compensate for transmission delay jitter and to reorder NAL units from transmission order to the NAL unit decoding order. In this section, the receiver operation is described under the assumption that there is no transmission delay jitter. To make a difference from a practical receiver buffer that is also used for compensation of transmission delay jitter, the receiver buffer is hereafter called the de-interleaving buffer in this section. Receivers SHOULD also prepare for transmission delay jitter; i.e., either reserve separate buffers for transmission delay jitter buffering and de-interleaving buffering or use a receiver buffer for both transmission delay jitter and de-interleaving. Moreover, receivers SHOULD take transmission delay jitter into account in the buffering operation; e.g., by additional initial buffering before starting of decoding and playback.

This section is organized as follows: subsection 6.2.1 presents how to calculate the size of the de-interleaving buffer. Subsection 6.2.2 specifies the receiver process how to organize received NAL units to the NAL unit decoding order.

6.2.1 Size of the De-interleaving Buffer

When the SDP Offer/Answer model or any other capability exchange procedure is used in session setup, the properties of the received stream SHOULD be such that the receiver capabilities are not exceeded. In the SDP Offer/Answer model, the receiver can indicate
its capabilities to allocate a de-interleaving buffer with the deint-buf-cap media type parameter. The sender indicates the requirement for the de-interleaving buffer size with the sprop-deint-buf-req media type parameter. It is therefore RECOMMENDED to set the de-interleaving buffer size, in terms of number of bytes, equal to or greater than the value of sprop-deint-buf-req media type parameter. See section 8.1 for further information on deint-buf-cap and sprop-deint-buf-req media type parameters and section 8.2.2 for further information on their use in the SDP Offer/Answer model.

When a declarative session description is used in session setup, the sprop-deint-buf-req media type parameter signals the requirement for the de-interleaving buffer size. It is therefore RECOMMENDED to set the de-interleaving buffer size, in terms of number of bytes, equal to or greater than the value of sprop-deint-buf-req media type parameter.

6.2.2 De-interleaving Process

There are two buffering states in the receiver: initial buffering and buffering while playing. Initial buffering occurs when the RTP session is initialized. After initial buffering, decoding and playback are started, and the buffering-while-playing mode is used.

Regardless of the buffering state, the receiver stores incoming NAL units, in reception order, in the de-interleaving buffer as follows. NAL units of aggregation packets are stored in the de-interleaving buffer individually. The value of DON is calculated and stored for each NAL unit.

The receiver operation is described below with the help of the following functions and constants:

- Function AbsDON is specified in section 7.1.
- Function don_diff is specified in section 4.6.
- Constant N is the value of the OPTIONAL sprop-interleaving-depth media type type parameter (see section 7.1) incremented by 1.

Initial buffering lasts until one of the following conditions is fulfilled:

- There are N or more VCL NAL units in the de-interleaving buffer.
If sprop-max-don-diff is present, don_diff(m,n) is greater than the value of sprop-max-don-diff, in which n corresponds to the NAL unit having the greatest value of AbsDON among the received NAL units and m corresponds to the NAL unit having the smallest value of AbsDON among the received NAL units.

Initial buffering has lasted for the duration equal to or greater than the value of the OPTIONAL sprop-init-buf-time media type parameter.

The NAL units to be removed from the de-interleaving buffer are determined as follows:

- If the de-interleaving buffer contains at least N VCL NAL units, NAL units are removed from the de-interleaving buffer and passed to the decoder in the order specified below until the buffer contains N-1 VCL NAL units.

- If sprop-max-don-diff is present, all NAL units m for which don_diff(m,n) is greater than sprop-max-don-diff are removed from the de-interleaving buffer and passed to the decoder in the order specified below. Herein, n corresponds to the NAL unit having the greatest value of AbsDON among the NAL units in the de-interleaving buffer.

The order in which NAL units are passed to the decoder is specified as follows:

- Let PDON be a variable that is initialized to 0 at the beginning of the RTP session.

- For each NAL unit associated with a value of DON, a DON distance is calculated as follows. If the value of DON of the NAL unit is larger than the value of PDON, the DON distance is equal to DON - PDON. Otherwise, the DON distance is equal to 65535 - PDON + DON + 1.

- NAL units are delivered to the decoder in ascending order of DON distance. If several NAL units share the same value of DON distance, they can be passed to the decoder in any order.

- When a desired number of NAL units have been passed to the decoder, the value of PDON is set to the value of DON for the last NAL unit passed to the decoder.
6.3 Additional De-Packetization Guidelines

The following additional de-packetization rules may be used to implement an operational HEVC de-packetizer:

- Intelligent RTP receivers (e.g., in gateways) may identify lost FUs. If a lost FU is found, a gateway may decide not to send the following FUs of the same fragmented NAL unit, as their information is meaningless for HEVC decoders. In this way a MANE can reduce network load by discarding useless packets without parsing a complex bitstream.

- Intelligent receivers having to discard packets or NALUs should first discard all packets/NALUs in which the value of the NRI field of the NAL unit type octet is equal to 0. This will minimize the impact on user experience and keep the reference pictures intact. If more packets have to be discarded, then packets with a NRI value equal to zero may be discarded before packets with a a higher NRI value. However, discarding any packets with an NRI not equal to zero very likely leads to decoder drift and SHOULD be avoided.

7. Payload Format Parameters

This section specifies the parameters that MAY be used to select optional features of the payload format and certain features of the bitstream. The parameters are specified here as part of the media type registration for the HEVC codec. A mapping of the parameters into the Session Description Protocol (SDP) [RFC4566] is also provided for applications that use SDP. Equivalent parameters could be defined elsewhere for use with control protocols that do not use SDP.

Some parameters provide a receiver with the properties of the stream that will be sent. The names of all these parameters start with "sprop" for stream properties. Some of these "sprop" parameters are limited by other payload or codec configuration parameters. For example, the sprop-parameter-sets parameter is constrained by the profile-level-id parameter. The media sender selects all "sprop" parameters rather than the receiver. This uncommon characteristic of the "sprop" parameters may be incompatible with some signaling protocol concepts, in which case the use of these parameters SHOULD be avoided.
7.1 Media Type Registration

The media subtype for the HEVC codec is allocated from the IETF tree.

The receiver MUST ignore any unspecified parameter.

Media Type name: video

Media subtype name: H265

Required parameters: none

OPTIONAL parameters:

In the following definitions of parameters, "the stream" or "the NAL unit stream" refers to all NAL units conveyed in the current RTP session in SST, and all NAL units conveyed in the current RTP session and all NAL units conveyed in other RTP sessions that the current RTP session depends on in MST.

profile-level-id:

TBD

sprop-parameter-sets:

TBD

max-mbps, max-smbps, max-fs, max-cpb, max-dpb, and max-br:

TBD

max-mbps:

TBD

max-smbps:

TBD

max-fs:

TBD

max-cpb:

TBD

max-dpb:

TBD
max-br:
  TBD

redundant-pic-cap:
  TBD

sprop-level-parameter-sets:
  TBD

use-level-src-parameter-sets:
  TBD

packetization-mode:
  This parameter signals the properties of an RTP payload type
  or the capabilities of a receiver implementation. Only a
  single configuration point can be indicated; thus, when
  capabilities to support more than one packetization-mode are
  declared, multiple configuration points (RTP payload types)
  must be used.

  When the value of packetization-mode is equal to 1, the non-
  interleaved mode, as defined in section 5.2 MUST be used.
  When the value of packetization-mode is equal to 2, the
  interleaved mode, as defined in section 5.3, MUST be used.
  The value of packetization-mode MUST be an integer in the
  range of 1 to 2, inclusive.

sprop-interleaving-depth:
  This parameter MUST NOT be present when packetization-mode is
  not present or the value of packetization-mode is equal to 0
  or 1. This parameter MUST be present when the value of
  packetization-mode is equal to 2.

  This parameter signals the properties of an RTP packet stream.
  It specifies the maximum number of VCL NAL units that precede
  any VCL NAL unit in the RTP packet stream in transmission
  order and follow the VCL NAL unit in decoding order.
  Consequently, it is guaranteed that receivers can reconstruct
  NAL unit decoding order when the buffer size for NAL unit
  decoding order recovery is at least the value of sprop-
  interleaving-depth + 1 in terms of VCL NAL units.

  The value of sprop-interleaving-depth MUST be an integer in
  the range of 0 to 32767, inclusive.

sprop-deint-buf-req:
  This parameter MUST NOT be present when packetization-mode is
not present or the value of packetization-mode is not equal to 2. It MUST be present when the value of packetization-mode is equal to 2.

sprop-deint-buf-req signals the required size of the de-interleaving buffer for the RTP packet stream. The value of the parameter MUST be greater than or equal to the maximum buffer occupancy (in units of bytes) required in such a de-interleaving buffer that is specified in section 6.2. It is guaranteed that receivers can perform the de-interleaving of interleaved NAL units into NAL unit decoding order, when the de-interleaving buffer size is at least the value of sprop-deint-buf-req in terms of bytes.

The value of sprop-deint-buf-req MUST be an integer in the range of 0 to 4294967295, inclusive.

Informative note: sprop-deint-buf-req indicates the required size of the de-interleaving buffer only. When network jitter can occur, an appropriately sized jitter buffer has to be provisioned for as well.

deint-buf-cap:
This parameter signals the capabilities of a receiver implementation and indicates the amount of de-interleaving buffer space in units of bytes that the receiver has available for reconstructing the NAL unit decoding order. A receiver is able to handle any stream for which the value of the sprop-deint-buf-req parameter is smaller than or equal to this parameter.

If the parameter is not present, then a value of 0 MUST be used for deint-buf-cap. The value of deint-buf-cap MUST be an integer in the range of 0 to 4294967295, inclusive.

Informative note: deint-buf-cap indicates the maximum possible size of the de-interleaving buffer of the receiver only. When network jitter can occur, an appropriately sized jitter buffer has to be provisioned for as well.

sprop-init-buf-time:
This parameter MAY be used to signal the properties of an RTP packet stream. The parameter MUST NOT be present, if the value of packetization-mode is equal to 1.

The parameter signals the initial buffering time that a receiver MUST wait before starting decoding to recover the NAL
unit decoding order from the transmission order. The parameter is the maximum value of (decoding time of the NAL unit - transmission time of a NAL unit), assuming reliable and instantaneous transmission, the same timeline for transmission and decoding, and that decoding starts when the first packet arrives.

An example of specifying the value of sprop-init-buf-time follows. A NAL unit stream is sent in the following interleaved order, in which the value corresponds to the decoding time and the transmission order is from left to right:

0  2  1  3  5  4  6  8  7 ...

Assuming a steady transmission rate of NAL units, the transmission times are:

0  1  2  3  4  5  6  7  8 ...

Subtracting the decoding time from the transmission time column-wise results in the following series:

0 -1  1  0 -1  1  0 -1  1 ...

Thus, in terms of intervals of NAL unit transmission times, the value of sprop-init-buf-time in this example is 1. The parameter is coded as a non-negative base-10 integer representation in clock ticks of a 90-kHz clock. If the parameter is not present, then no initial buffering time value is defined. Otherwise the value of sprop-init-buf-time MUST be an integer in the range of 0 to 4294967295, inclusive.

In addition to the signaled sprop-init-buf-time, receivers SHOULD take into account the transmission delay jitter buffering, including buffering for the delay jitter caused by mixers, translators, gateways, proxies, traffic-shapers, and other network elements.

sprop-max-don-diff:
This parameter MAY be used to signal the properties of an RTP packet stream. It MUST NOT be used to signal transmitter or receiver or codec capabilities. The parameter MUST NOT be present if the value of packetization-mode is equal to 1. sprop-max-don-diff is an integer in the range of 0 to 32767, inclusive. If sprop-max-don-diff is not present, the value of
the parameter is unspecified. sprop-max-don-diff is calculated as follows:

\[ sprop-max-don-diff = \max\{\text{AbsDON}(i) - \text{AbsDON}(j)\}, \]
for any \( i \) and any \( j > i \),

where \( i \) and \( j \) indicate the index of the NAL unit in the transmission order and AbsDON denotes a decoding order number of the NAL unit that does not wrap around to 0 after 65535. In other words, AbsDON is calculated as follows: Let \( m \) and \( n \) be consecutive NAL units in transmission order. For the very first NAL unit in transmission order (whose index is 0), \( \text{AbsDON}(0) = \text{DON}(0) \). For other NAL units, AbsDON is calculated as follows:

If \( \text{DON}(m) = \text{DON}(n) \), \( \text{AbsDON}(n) = \text{AbsDON}(m) \)

If \( \text{DON}(m) < \text{DON}(n) \) and \( \text{DON}(n) - \text{DON}(m) < 32768 \),
\( \text{AbsDON}(n) = \text{AbsDON}(m) + \text{DON}(n) - \text{DON}(m) \)

If \( \text{DON}(m) > \text{DON}(n) \) and \( \text{DON}(m) - \text{DON}(n) >= 32768 \),
\( \text{AbsDON}(n) = \text{AbsDON}(m) + 65536 - \text{DON}(m) + \text{DON}(n) \)

If \( \text{DON}(m) < \text{DON}(n) \) and \( \text{DON}(m) - \text{DON}(n) >= 32768 \),
\( \text{AbsDON}(n) = \text{AbsDON}(m) - (\text{DON}(m) + 65536 - \text{DON}(n)) \)

If \( \text{DON}(m) > \text{DON}(n) \) and \( \text{DON}(m) - \text{DON}(n) < 32768 \),
\( \text{AbsDON}(n) = \text{AbsDON}(m) - (\text{DON}(m) - \text{DON}(n)) \)

where \( \text{DON}(i) \) is the decoding order number of the NAL unit having index \( i \) in the transmission order. The decoding order number is specified in section 4.6.

Informative note: Receivers may use sprop-max-don-diff to trigger which NAL units in the receiver buffer can be passed to the decoder.

max-rcmd-nalu-size:
\[ \text{TBD} \]

sar-understood:
\[ \text{TBD} \]

sar-supported:
\[ \text{TBD} \]
Encoding considerations:
This type is only defined for transfer via RTP (RFC 3550).

Security considerations:
See Section 8 of RFC XXXX.

Public specification:
Please refer to Section 13 of RFC XXXX.

Additional information:
None

File extensions: none
Macintosh file type code: none
Object identifier or OID: none
Person & email address to contact for further information:
Thomas Schierl, ts@thomas-schierl.de

Intended usage: COMMON

Author:
Thomas Schierl, ts@thomas-schierl.de

Change controller:
IETF Audio/Video Transport Payloads working group delegated from the IESG.

7.2 SDP Parameters

7.2.1 Mapping of Payload Type Parameters to SDP
TBD

7.2.2 Usage with the SDP Offer/Answer Model

The media type video/H265 string is mapped to fields in the Session Description Protocol (SDP) [RFC4566] as follows:

- The media name in the "m=" line of SDP MUST be video.
- The encoding name in the "a=rtpmap" line of SDP MUST be H265 (the media subtype).
o The clock rate in the "a=rtpmap" line MUST be 90000.

o The OPTIONAL parameters "profile-level-id", "packetization-mode", when present, MUST be included in the "a=fmtp" line of SDP. These parameters are expressed as a media type string, in the form of a semicolon separated list of parameter=value pairs.

o The OPTIONAL parameters "sprop-parameter-sets" and "sprop-level-parameter-sets", when present, MUST be included in the "a=fmtp" line of SDP or conveyed using the "fmt" source attribute as specified in section 6.3 of [RFC5576]. For a particular media format (i.e., RTP payload type), a "sprop-parameter-sets" or "sprop-level-parameter-sets" MUST NOT be both included in the "a=fmtp" line of SDP and conveyed using the "fmt" source attribute. When included in the "a=fmtp" line of SDP, these parameters are expressed as a media type string, in the form of a semicolon separated list of parameter=value pairs. When conveyed using the "fmt" source attribute, these parameters are only associated with the given source and payload type as parts of the "fmt" source attribute.

Informative note: Conveyance of "sprop-parameter-sets" and "sprop-level-parameter-sets" using the "fmt" source attribute allows for out-of-band transport of parameter sets in topologies like Topo-Video-switch-MCU [TBD].

An example of media representation in SDP is as follows:

m=video 49170 RTP/AVP 98
a=rtpmap:98 H265/90000
a=fmtp:98 profile-level-id=UVWXYZ;
   packetization-mode=1;
   sprop-parameter-sets=<parameter sets data>

7.2.3 Usage with SDP Offer/Answer Model

TBD

7.2.4 Usage in Declarative Session Descriptions

TBD

7.2.5 Signaling of Parallel Processing

TBD
7.3 Examples

TBD.

7.4 Parameter Set Considerations

TBD

8. Security Considerations

TBD

9. Congestion Control

TBD

10. IANA Consideration

A new media type, as specified in Section 7.1 of this memo, should be registered with IANA.

11. Informative Appendix: Application Examples

11.1 Introduction

TBD

11.2 Streaming

TBD

11.3 Videoconferencing (Unicast to MANE, Unicast to Endpoints)

TBD

11.4 Mobile TV (Multicast to MANE, Unicast to Endpoint)

TBD

12. Acknowledgements

TBD

This document was prepared using 2-Word-v2.0.template.dot.
13. References

13.1 Normative References


13.2 Informative References

14. Authors’ Addresses

Thomas Schierl
Fraunhofer HHI
Einsteinufer 37
D-10587 Berlin
Germany
Phone: +49-30-31002-227
EMail: ts@thomas-schierl.de

Stephan Wenger
Vidyo, Inc.
433 Hackensack Ave., 7th floor
Hackensack, N.J. 07601
USA
Phone: +1-415-713-5473
EMail: stewe@stewe.org

Ye-Kui Wang
Qualcomm Incorporated
5775 Morehouse Drive
San Diego, CA 92121
USA
Phone: +1-858-651-8345
EMail: yekuiw@qualcomm.com

Miska M. Hannuksela
Nokia Corporation
P.O. Box 1000
33721 Tampere
Finland
Phone: +358-7180-08000
EMail: miska.hannuksela@nokia.com
Codec Operation Point RTCP Extension
draft-westerlund-avtext-codec-operation-point-00

Abstract

The Audio-Visual Profile with Feedback (AVPF) specification defines a framework and messages for fast feedback and media control over RTCP. The Codec Control Messages (CCM) specification defines an extension to AVPF, by specifying additional messages for codec control and feedback. This specification extends CCM, by specifying messages that let participants dynamically communicate a set of codec configuration parameters, which enables better optimization of resource efficiency and quality of media transmission.

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 September 6, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ............................................. 4
2. Definitions ............................................. 5
   2.1. Terminology ......................................... 5
   2.2. Abbreviations ....................................... 6
   2.3. Requirements Language ................................ 7
3. Motivation ............................................... 7
   3.1. Problem Description .................................. 7
   3.2. Legacy Methods ..................................... 10
      3.2.1. Relation to SDP .................................. 10
      3.2.2. Relation to RTCP ................................ 10
4. Use Cases for COP ....................................... 11
   4.1. Point to Point ....................................... 11
   4.2. Media Receiver to RTP Mixer ......................... 12
   4.3. RTP Mixer to Media Sender ........................... 13
   4.4. Media Receiver in Multicast or with RTP Transport  15
5. Requirements ............................................. 18
6. Solution Overview ....................................... 19
   6.1. Message Structure ................................... 21
   6.2. Codec Configuration Parameter Use .................... 22
   6.3. Operation Point ...................................... 22
   6.4. Request ............................................ 24
   6.5. Notification ........................................ 25
   6.6. Status Report ....................................... 26
   6.7. Adding and Removing Operation Points ................ 27
7. Codec Control Message Extension ......................... 27
   7.1. COP Message ......................................... 28
   7.2. FCI Format .......................................... 28
      7.2.1. Message Item Format .............................. 28
      7.2.2. Message Item Types .............................. 29
      7.2.3. Operation Point Identification .................. 29
   7.3. Codec Operation Point Notification ................... 30
      7.3.1. Message Format .................................. 30
      7.3.2. Semantics ...................................... 31
      7.3.3. Timing Rules .................................... 34
      7.3.4. Handling in Mixers and Translators ............. 34
   7.4. Codec Operation Point Request ....................... 35
      7.4.1. Message Format .................................. 35
      7.4.2. Semantics ...................................... 36
      7.4.3. Timing Rules .................................... 38
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.4.4.</td>
<td>Handling in Mixers and Translators</td>
<td>38</td>
</tr>
<tr>
<td>7.5.</td>
<td>Codec Operation Point Status</td>
<td>39</td>
</tr>
<tr>
<td>7.5.1.</td>
<td>Message Format</td>
<td>39</td>
</tr>
<tr>
<td>7.5.2.</td>
<td>Semantics</td>
<td>40</td>
</tr>
<tr>
<td>7.5.3.</td>
<td>Timing Rules</td>
<td>42</td>
</tr>
<tr>
<td>7.5.4.</td>
<td>Handling in Mixers and Translators</td>
<td>42</td>
</tr>
<tr>
<td>8.</td>
<td>Parameter Types</td>
<td>42</td>
</tr>
<tr>
<td>8.1.</td>
<td>Parameter Format</td>
<td>42</td>
</tr>
<tr>
<td>8.2.</td>
<td>ALT</td>
<td>46</td>
</tr>
<tr>
<td>8.3.</td>
<td>ID</td>
<td>46</td>
</tr>
<tr>
<td>8.4.</td>
<td>Payload Type</td>
<td>47</td>
</tr>
<tr>
<td>8.5.</td>
<td>Bitrate</td>
<td>48</td>
</tr>
<tr>
<td>8.6.</td>
<td>Token Bucket Size</td>
<td>49</td>
</tr>
<tr>
<td>8.7.</td>
<td>Framerate</td>
<td>50</td>
</tr>
<tr>
<td>8.8.</td>
<td>Horizontal Pixels</td>
<td>51</td>
</tr>
<tr>
<td>8.9.</td>
<td>Vertical Pixels</td>
<td>51</td>
</tr>
<tr>
<td>8.10.</td>
<td>Channels</td>
<td>52</td>
</tr>
<tr>
<td>8.11.</td>
<td>Sampling Rate</td>
<td>52</td>
</tr>
<tr>
<td>8.12.</td>
<td>Maximum RTP Packet Size</td>
<td>54</td>
</tr>
<tr>
<td>8.13.</td>
<td>Maximum RTP Packet Rate</td>
<td>54</td>
</tr>
<tr>
<td>8.14.</td>
<td>Application Data Unit Aggregation</td>
<td>55</td>
</tr>
<tr>
<td>9.</td>
<td>SDP Extensions</td>
<td>56</td>
</tr>
<tr>
<td>9.1.</td>
<td>Extension of the rtcp-fb Attribute</td>
<td>56</td>
</tr>
<tr>
<td>9.2.</td>
<td>Offer/Answer Usage</td>
<td>57</td>
</tr>
<tr>
<td>9.3.</td>
<td>Declarative Usage</td>
<td>58</td>
</tr>
<tr>
<td>10.</td>
<td>Codec Sub-Stream Identification</td>
<td>58</td>
</tr>
<tr>
<td>10.1.</td>
<td>H.264 AVC</td>
<td>59</td>
</tr>
<tr>
<td>10.2.</td>
<td>H.264 SVC</td>
<td>59</td>
</tr>
<tr>
<td>11.</td>
<td>Examples</td>
<td>60</td>
</tr>
<tr>
<td>11.1.</td>
<td>SDP Offer/Answer</td>
<td>60</td>
</tr>
<tr>
<td>11.2.</td>
<td>Dynamic Video Re-sizing</td>
<td>62</td>
</tr>
<tr>
<td>11.3.</td>
<td>Illegal Request</td>
<td>64</td>
</tr>
<tr>
<td>11.4.</td>
<td>Reference Response to Modification of Scalable Layer</td>
<td>65</td>
</tr>
<tr>
<td>11.5.</td>
<td>Successful Request to Add Codec Operation Point</td>
<td>67</td>
</tr>
<tr>
<td>12.</td>
<td>IANA Considerations</td>
<td>69</td>
</tr>
<tr>
<td>13.</td>
<td>Security Considerations</td>
<td>69</td>
</tr>
<tr>
<td>14.</td>
<td>Open Issues</td>
<td>70</td>
</tr>
<tr>
<td>15.</td>
<td>Acknowledgements</td>
<td>70</td>
</tr>
<tr>
<td>16.</td>
<td>References</td>
<td>70</td>
</tr>
<tr>
<td>16.1.</td>
<td>Normative References</td>
<td>70</td>
</tr>
<tr>
<td>16.2.</td>
<td>Informative References</td>
<td>71</td>
</tr>
<tr>
<td>Authors’ Addresses</td>
<td>72</td>
<td></td>
</tr>
</tbody>
</table>
1. Introduction

Multimedia real-time communication services, such as video telephony and videoconferencing, use the real-time transport (RTP/RTCP) [RFC3550] protocol to transmit media streams, such as audio and video. A session establishment protocol, such as SIP [RFC3261], in combination with a capability negotiation protocol, such as SDP offer/answer [RFC3264] is normally used to establish the session and negotiate media capabilities. In some cases, a set of codec parameters is negotiated that does not express any specific limit or capability, but just describes a certain codec configuration.

During session establishment, the participating endpoints normally have limited knowledge about the session environment, e.g. whether the session will be point-to-point or contain some multi-party scenario, how users will interact with the application, how network conditions will vary during the session, etc. To take those variations into account, the participants can re-negotiate session parameters to better suit the communication environment. At times, when variations or changes are frequent in nature, it will require the needed reaction time to be short, which may make repeated session re-negotiation inefficient and/or too slow. In addition, variations may not even affect negotiated session parameters, if the variations occur within the negotiated boundaries.

The above scenario can become critical especially in cases where a given media stream is transmitted towards, and received by, multiple receivers. In multi-party environments, scalable encoding or simulcast can be used to make the system more efficient and provide better quality to participants that are capable of receiving and utilizing the higher quality. These use cases results in that a sending party is requested to deliver multiple encoder operation points.

The Audio-Visual Profile with Feedback (AVPF) specification [RFC4585] defines a framework and messages for fast feedback and media control over RTCP. The Codec Control Messages (CCM) specification [RFC5104] defines an extension to AVPF, by specifying additional messages for codec control and feedback. This specification extends CCM, by specifying messages that let participants dynamically communicate a set of codec configuration parameters, which enables better optimization of resource efficiency and quality of media transmission.

The codec configuration parameters specified in this document focus on some basic audio and video properties, such as video resolution, video frame rate, media stream bit-rate, audio sampling rate, number of audio channels, maximum RTP packet size and rate. Additional

parameters can be standardized in the future.

The codec control messages are not meant to replace configuration performed using e.g. SDP. Instead, the messages can be used to communicate dynamic and frequent changes that take place within boundaries that have been negotiated as part of the session establishment.

2. Definitions

2.1. Terminology

The following terms and abbreviations are used in this document:

Bandwidth: The network resource needed to transport a certain bitrate and any transport overhead, measured in bits per second. There will be spare network bandwidth when the (media) data bitrate and overhead is less than the available bandwidth. Similarly, data will have to be buffered when the available bandwidth excluding transport overhead is less than the bitrate used by the sender, or the excess data will be lost. The available bandwidth typically varies dynamically over time.

Bitrate: The amount of (media) data transmitted per time unit, measured in bits per second, utilizing some amount of the available network bandwidth resource. In the context of this specification and unless otherwise specified, it excludes IP/UDP/RTP overhead. Depending on (media) data source, the bitrate can either be constant or vary dynamically over time.

Codec Configuration Parameter: The configurable value describing a certain codec property, which may impact user-perceived media fidelity, encoded media stream characteristics, or both. The parameter has a type (Codec Parameter Type, see below) and a value, where the type describes what kind of codec property that is controlled, and the value describes the property setting as well as how the value should be used in comparison operations. A single Parameter Value can express one specific value or an open-ended range. A pair of Parameter Values with different comparison types can describe a value range. Such value range can also be combined with a third, target value within that range.

Codec Operation Point: Also denoted just Operation Point. A set of Codec Configuration Parameter values, describing the characteristics of one single encoding. For scalable encoding, it describes the resulting characteristics from combining a set of dependent sub-streams.
Codec Parameter Type: The specific type of a Codec Configuration Parameter. Each parameter type defines what unit the value has. This specification defines a number of generally useful parameter types in Section 8 that can be used to control codec operation.

Encoding: A particular encoding is the resulting media stream from applying a certain choice of Codec Configuration Parameters to the encoder. The media stream will have a certain fidelity (quality) from that encoding through the choice of sampling, bit-rate and other configuration parameters.

Endpoint: A host or node that have a presence in the RTP session with one or more Synchronization Sources (SSRC)s.

Mixer: An RTP session centralized node that generates media streams based on incoming media streams from other endpoints. See Topo-Mixer in RTP Topologies [RFC5117].

RTP Session: An association among a set of participants communicating with RTP. The distinguishing feature of an RTP session (defined in [RFC3550]) is that each RTP session maintains a full, separate space of SSRC identifiers. Each participant in the RTP session can see SSRC or CSRC identifiers from the other participants, either by RTP, RTCP, or both.

Sub-Stream: An individually decodeable part of a scalable media stream, including all dependent sub-streams. The characteristics of a certain sub-stream can be described by a Codec Operation Point.

Translator: An RTP session centralized node that forwards all media streams from other endpoints, modified to some extent, e.g. addressing, encoding, fidelity. See Topo-Translator in RTP Topologies [RFC5117].

2.2. Abbreviations

AVC: Advanced Video Coding

AVPF: Extended RTP Profile for RTCP-Based Feedback

CCP: Codec Configuration Parameter

COP: Codec Operation Point
COPN: Codec Operation Point Notification
COPR: Codec Operation Point Request
COPS: Codec Operation Point Status
CPT: Codec Parameter Type
FCI: Feedback Control Information
FMT: Feedback Message Type
GUI: Graphical User Interface
MST: Multi-Session Transmission
MVC: Multiview Video Coding
OP: Operation Point
OPID: Operation Point Identification number
PPS: Picture Parameter Set
SPS: Sequence Parameter Set
SST: Single-Session Transmission
SVC: Scalable Video Coding
TLV: Type-Length-Value

2.3. Requirements Language

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

3.1. Problem Description

Networks can contain endpoints with different capabilities, including CPU power, capture and render device fidelity (e.g. image resolution), and codecs. In addition, the characteristics and properties of networks can vary, which endpoints have to cope with. For example, in videoconferencing and telepresence services, a large
number of endpoints may participate, and there may be a large number of media streams associated with the session. Such multi-party scenarios typically use entities for media mixing, switching and transcoding. The aim is generally to provide the best possible quality to each endpoint, taking endpoint and network capabilities into consideration.

Many communication services today use codecs that can be configured in a number of different ways. Often, the codecs have multiple properties that can be configured and those properties may also be inter-related, often in complex ways. One example is the H.264 (AVC) [H264] video codec and its scalable (SVC) and multi-view (MVC) versions. Most other video codecs, and codecs for many other types of media, also have multiple configurable properties. Such configurable properties will be referred to as "Codec Configuration Parameters" in this specification.

There can be several reasons to change the media rate or other encoding or packetization properties during an ongoing communication session. One reason can be that the available network bandwidth varies. Another reason can be that other network properties changes, such as effective MTU or packet rate limitations. Other reasons can be that the quality or representation of the media rendered to the end user changes, maybe as a direct result of the user manipulating the GUI (e.g. changing window position or size), the relative importance of the received media stream changes (e.g. active or non-active speaker in a conferencing scenario), or the user selects to show some other content source that is available among the advertised media streams.

The codec changes above can be made directly between endpoints in a point-to-point scenario, or they may involve, and be acted upon, by media aware intermediaries (e.g. RTP mixers). An RTP mixer can do transcoding to provide each receiver with media streams of adapted quality, but transcoding has drawbacks as it always consumes processing power, typically impacts media quality in a negative way, and often introduces additional delays.

In order to avoid separate transcoding towards each endpoint, an RTP Mixer can, by taking the capabilities of the endpoints into account, decide to request specific codec configurations from endpoints, which will minimize the need for transcoding. Also, in scenarios where no RTP Mixers are used and transmitted media reaches multiple endpoints, the sender will have to take into account that each endpoint may have different capabilities. The use cases section (Section 4) shows different use cases, with and without RTP Mixers.

Resource optimization involving bandwidth is expected to be one of...
the major reasons for changing encoding properties, since it is in general desirable to avoid using more bandwidth than absolutely necessary, especially considering that

- the expectation for high media quality will likely continue to increase;

- the bitrate required to transmit the media, despite increasingly efficient media coding, can due to the above also be expected to increase;

- the relation between media bitrate and media codec configuration, the used set of media codec property values, is typically complex and the mapping between each individual codec property and bitrate is in general not linear;

- the used media bitrate does not uniquely identify the media codec configuration, but there are in general multiple codec configurations that can generate the same media bitrate;

- the media receiver preferences how the codec property values should be set for a certain media bitrate will typically vary with the specific end-user service requirements (for example, but not limited to, users with special needs) and the current media stream role in the application;

- the communication scenarios will not be limited to point-to-point, potentially involving multiple and at least partly conflicting constraints from different receivers; and

- the available bandwidth is commonly a scarce and/or costly resource and will likely continue to be so also in the future.

Other resources that may be desirable to optimize include, but is not limited to, endpoint and middle node processing (CPU) utilization, and transport quality (QoS).

A media receiver cannot be assumed to know exactly what codec configuration will be best for the media sender to use, given that the sender needs to take multiple aspects into account, including implementation limitations in the actual encoder. It should be more likely to find a value acceptable to both sender and receiver if the receiver can indicate an acceptable range instead of just a single value.

When an RTP Mixer distributes streams to multiple receivers with different media quality requirements, it is sometimes possible to avoid targeted transcoding for every single receiver. That can be
accomplished if the media sender has the ability to produce multiple media versions, such as for example scalable encoding or simulcast. Thus there is a need to both address specific media versions and describe the fact that multiple media versions with different configurations should be used.

3.2. Legacy Methods

3.2.1. Relation to SDP

The session description protocol (SDP) [RFC4566] is commonly used to negotiate and configure codecs and establish RTP/RTCP session parameters during session establishment, and during sessions, e.g. by using it in conjunction with SIP [RFC3261] and SDP Offer/Answer [RFC3264].

As described Section 3.1 above, many of the underlying reasons that makes media receivers desire certain codec encoding properties are highly dynamic in nature and using SIP/SDP to re-negotiate the session will in many cases be too slow to be useful. SIP messages containing an SDP may become quite large for sessions containing many media, and since there is no defined way to send a partial SDP, even very small changes require sending the entire SDP. Most of the current defined properties in SDP are also oriented to be common for all media streams in the same RTP session, rather than be specific to one media stream.

The mechanism in this specification does not replace SDP, or the SDP Offer/Answer mechanism. It is expected that SDP is used in order to negotiate and configure boundary values for codec properties, and COP can then be used to communicate specific values within those boundaries, as long as there is no impact on the values negotiated using SDP. It is possible to establish communication sessions even if one or more endpoints do not support COP.

3.2.2. Relation to RTCP

As discussed in CCM, regular RTCP reporting or extended reports [RFC3611] can to some extent be used to re-configure an encoder, but the reported measures seldom map directly back to encoding properties and they typically cannot express an unwanted situation in terms of encoding properties and what the receiver would like to receive instead. Communicating codec properties indirectly as a set of network properties will require interpretation by both sender and receiver and will thus risk misinterpretations and ambiguity. Since it is likely that a decoder is able to identify unwanted characteristics of the media stream in terms of encoding properties, the most straightforward approach is to convey those properties
directly to the encoder.

Responsive techniques to control encoding are already available, e.g. Codec Control Messages (CCM) [RFC5104]. Although highly applicable, the possibilities to control encoding is however not explicit enough, both in terms of the amount of available parameters to control, and the fact that they may be inter-related, alternative, or both.

Some codecs define codec-specific methods to enable receiver control of some encoding aspects, but it should be beneficial for interoperability to use codec agnostic signaling instead.

4. Use Cases for COP

This section discusses a number of use cases for Codec Operation Points.

4.1. Point to Point

This set of use cases are all focused on that communication is directly point to point between a media sender and a receiver. There is no need for further forwarding of the media streams. Thus, the goal should be to produce a media stream, transport it to the media receiver, where it is consumed as optimal as possible for the application. Thanks to this one-to-one mapping between encoder and decoder, great flexibility exists to produce a media stream tailored to the receiver’s needs, given the constraints that exist from media sender, transport network and the receiver.

Some constraints will be static (and thus suitable for session configuration signalling), but a number of these are highly dynamical and thus desirable to adapt to during the session:

Video Resolution in GUI: In a video communication application, including WebRTC based ones, the window where the media senders media stream is presented may change, for example due to the user modifying the size of the window. It might also be due to other application related actions, like selecting to show a collaborative work space and thus reducing the area used to show the remote video in. In both of these cases it is the receiver side that knows how big the actual screen area is and what the most suitable resolution would be. It thus appears suitable to let the receiver request the media sender to send a media stream conforming to the displayed video size.
Network Bit-rate Limitations: If the receiver discovers a network bandwidth limitation, it can choose to meet it by requesting media stream bit-rate limitations. Especially in cases where a media sender provides multiple media streams, the relative distribution of available bit-rate could help the application provide the most suitable experience in a constrained situation.

CPU Constraint: A media receiver may become constrained in the amount of available processing resources. This may occur in the middle of a session for example due to the user selecting a power saving mode, or starting additional applications requiring resources. When this occurs, the receiving application can select which codec parameters to constrain and how much constrained they should be to best suit the needs of the application. For example, if lower framerate is somehow a better constraint than lower resolution.

4.2. Media Receiver to RTP Mixer

This section considers a multiparty session with a centralized media intermediary, like an RTP mixer, where the media receiver uses COP to affect the delivered media.

```
+-------------+        +---+
|            |--RTP-->| B |
|            |<--COP--|   |
|            |        +---+
|            |
+---+      |            |        +---+
| A |-RTP->|   Mixer    |--RTP-->| C |
+---+      |            |        +---+
|            |        +---+
|            |--RTP-->| D |
+-------------+        +---+
```

Figure 1: Receiver (B) using COP to adapt media stream

In the above Figure 1 we focus on the possible usages of COP by a media receiver, like B. Here the functional role of the intermediary becomes important. An RTP mixer uses its own SSRC(s) to channel selected media streams to B from other participants like A. If the intermediary is instead a translator, the Receiver B can see A’s SSRC(s) directly instead of possibly showing up as CSRC. We will in this section focus on the Mixer case. The RTP translator case is further discussed in Section 4.4.

The RTP mixer’s usage of its own SSRC allows particular mixer to
receiver media flows to be associated with a particular role or purpose in the application rather than a given media source. When there exist multiple RTP streams from the mixer to a receiver, the receiver can use COP to request an operations point that better suits the receiver needs on each particular stream and possibly role of the media stream. It also allows the receiver to select its desired trade-off in properties and quality between multiple delivered media streams.

There exist some different reasons why B would need to indicate changes in its capabilities to receive a particular media stream;

Network Path: The receiver detects changes in the network that on a mid to long term will result in a new capability regarding the maximum bit-rate that can be supported.

Bandwidth trade-off: In an application receiving multiple media streams, if the receiving application likes to change the relative bit-rate trade-off between the streams.

Presentation or Graphical User Interface Changes: If the presentation or graphical user interface (GUI) changes on the receiving side results in other requirements or needs on the media streams. For example if the application window is re-sized by the user, the amount of screen estate to present the different video elements changes. To optimize the video quality in relation to bit-rate the receiver indicates the new preferred video resolution.

In all the above cases the receiver sends a COP request to the mixer for new codec operation points on mixer controlled media stream(s). It then becomes the mixer’s responsibility to determine if and how the requested COPs can be supported. For example by requesting new operations points from the media source as discussed in Section 4.3. The selection of another media source to deliver in a media stream can result in that the mixer may have to update the receiver on the properties of the operations point.

4.3. RTP Mixer to Media Sender

This section looks at the usage of COP in cases of multiparty with centralized media intermediary, like an RTP mixer, selecting and requesting tailored media stream or streams a media sender delivers to the intermediary for further forwarding or manipulation. This usage can be simplified down to looking at the media streams from one media sender (A), which is currently being delivered to multiple receivers (B-D) as depicted in Figure 2.
The media path from the Mixer to B, C and D are different and thus the available resources may vary between them. In addition B, C and D may have different capabilities when it comes to handling media streams. These limitations can be learned by the Mixer through session configuration signalling, media transmission feedback (e.g. RTCP), or usage of COP by the receivers (See Section 4.2). Limitations are also expected to be updated during the session lifetime.

The media sender (A) has certain capabilities and what is possible to do will depend on A’s capabilities and what has been configured between A and the Mixer. Let’s look at a few different cases of the capabilities A may have and how that influence how the Mixer can use COP to affect the media stream(s) delivered to the Mixer.

**Single Media Encoding:** If A can only provide a single media encoding of a particular media source, then the Mixer has to make a choice on what property it would like to request for that media stream. The most basic choice is to request the lowest common denominator across the receiver population. If the mixer has certain capabilities for media transcoding it could select to request another operation point for the media encoding with higher quality and then transcode to some few receivers. That enables a higher quality to several receivers while still being able to serve endpoints with the least capabilities. In these cases the Mixer has to make COP requests that indicate only a single operation point with parameters that best matches the restrictions.

**Scalable Media Encoding:** If A is capable of producing a scalable media stream encoding, the Mixer can request multiple operation points for the same media stream. For example, if A is capable of producing three different operation points, the Mixer in the above Figure 2 would potentially be able to request scalability layers...
that would allow it to match the capabilities of all the three receivers B, C and D. If several receivers are close in capabilities, the mixer may choose to request fewer operation points. Something that arise in this use case which wasn't present in the single media encoding above is that the mixer must determine which packets or parts of packets that are to be sent to each receiver based on their capabilities. This requires that the mixer is capable of identifying in the media stream which scalability layers that match a given requested operation point. Thus it is desirable that the media sender can indicate to Mixer what layer(s) that match a given operations point.

Simulcast Media: If A and the Mixer has negotiated the usage of simulcasted media encoding of the media source, then the Mixer can adopt several operation points to best suit the receiver set, just like for scalable encoding. When simulcasting, the mixer will however have to send one COP request per media stream it actually wants to affect. Some consideration is necessary to ensure that configuration changes over multiple media streams from the same media source take place. Compared to scalable media, the mixer need not strip away any layers to get at a particular operation point but can forward entirely self-contained media streams.

The use of COP as described above can be triggered by a multitude of reasons. We will here discuss some of them. We already mentioned that bit-rate adaptation (congestion control) on the Mixer to receiver path can indicate a need to change an operation point. Another reason is when a new session participant joins that has certain receiver capabilities (both decoding or other hardware, as well as network path related), thus potentially changing the optimal set of operation points. There also exist a number of different cases where the desired application behavior results in changes in desired operation points, like change of active speakers, reconfiguration of the display layout, etc.

It is also important to remember that Figure 2 only presents the view of a single media sender. In most communication sessions there are multiple media senders, and the mixer will need to take the combination of media streams from multiple media senders into account when choosing what is to be sent to a given receiver. Thus changes at one media sender can result in related changes of the operation points at the other media senders.

4.4. Media Receiver in Multicast or with RTP Transport Translator

This section covers usage of COP in multicast transported RTP sessions, as well as when transport translators [RFC5117] are used. Transport translators can be used to emulate any source multicast...
(ASM) over unicast. Multicast usages also include Source Specific Multicast (SSM) [RFC4607], which according to "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback" [RFC5760] has two main modes; simple mode and summary feedback mode, affecting the usage of functionality that COP provides.

A transport translator [RFC5117], which main purpose is to forward any incoming packets to all the other session participants, emulates an ASM session. As anyone can send to all other in both cases, there are some properties in these sessions that can make use in large scale sessions with many participants require some extra consideration.
In the above Figure 4, the media senders (MS1 .., MSm) send their media streams and RTCP traffic to the distribution source (DS). The DS forwards the RTP and RTCP traffic from the media senders to the SSM group. Using the RTCP extension for unicast RTCP feedback [RFC5760], the receivers (R1...Rn) send their RTCP traffic to their configured feedback target. This sample session has two feedback targets to scale with the amount of receivers. RTCP messages that needs to go to a media sender is forwarded to the FT aggregator part of the distribution source for further forwarding over the unicast paths between the distribution source and the media senders. The feedback target and the feedback aggregator also forwards all RTCP messages from receivers in simple mode, and aggregate it in summary mode. Some RTCP messages from a receiver may still have to be forwarded over the SSM group.

COP needs to support some reasonable functionality over the different
multiparty topologies described above and it is also important that COP does not cause significant issues in any of the environments.

In the basic case, where only a single multicast group exists, there is a well known problem associated with adapting content and bit-rate to the receiver population. The more receivers, the larger the potential for non-matching requirements in requests from the different receivers. One strategy for meeting this is to use the lowest common denominator among the requests from the receiver population. This normally results in sub-optimal quality for a significant part of the session participants, the main benefit being that all participants will be able to receive some content.

Because the above limitations of operation within a single group, usage of COP in larger groups becomes difficult unless the parameters that can be adopted and affected by COP requests are such that a limited set of participants is expected to request them, and the impact for the others are limited or acceptable. The authors therefore expects the usage of COP in large groups to be limited and this specification focuses on operation in smaller groups. However, as it is not possible to define the threshold when a group changes from being small to be too large to work well with COP in the generic case, it is important that COP can operate safely in a large group, although the possibilities to satisfy the request may be severely limited.

There also exist use cases for COP where the media application uses multiple multicast groups to enable multiple operation points and allows each receiver to join the multicast groups that suits the participant’s capabilities. An example of such usage would be Scalable Video Coding (SVC) using the Multi-Session Transport (MST) mode of the SVC RTP payload format [RFC6190]. The SVC MST RTP streams that are sent in each group can still contain multiple scalability layers; one could combine coarse-grained control on the operation points by having the receiver join a particular session with a more fine-grained control using COP to adjust the included scalability layers to suit the receiver’s needs, such as lower CPU load.

5. Requirements

The solution outlined in this specification should fulfill the following requirements:
REQ-1: Enable dynamic control of possibly inter-related codec properties during an ongoing media session.

REQ-2: Be media type agnostic, to the furthest extent possible, and at least cover audio and video media.

REQ-3: Be codec agnostic (within the same media type), to the furthest extent possible.

REQ-4: Work with different media transmission types, i.e. single-stream, simulcast, single-stream scalable, and multi-stream scalable transmission.

REQ-5: Work with un-encrypted as well as encrypted media.

REQ-6: Be extensible, making it simple to add control and description of new codec properties.

REQ-7: Complement rather than conflict with other codec configuration methods such as e.g. other RTCP based techniques and SDP.

REQ-8: Support configurable parameters that are directly visible in the media stream as well as those that are not visible in the media stream.

In addition, Guidelines for Extending RTCP [RFC5968] should be followed to the furthest extent possible.

6. Solution Overview

The mechanism described in this specification especially targets heterogeneous multi-party scenarios where different endpoints require differently encoded media from the same source, but its use in other situations is not precluded, in fact point to point scenarios is considered to be of equal importance but no more demanding that the multiparty case. In the targeted scenario, the media stream from one encoder is sent to multiple decoders, and hence the encoder must possibly provide an encoding with multiple operation points, suitable for the receivers. This is typically only possible with so-called scalable codecs, but some codecs may have inherent scalability features without being generally considered as scalable (e.g. H.264/AVC temporal scalability through non-reference frames). Multi-party services often involve a media mixer (Topo-Mixer) [RFC5117] as a central network node.
Figure 5: Sample Mixer Topology

The solution defined in this specification can be used during an active session to quickly adapt to changes in media receiver available bandwidth and/or preferences for one or more other codec properties, while still conforming to the session configuration, like SDP offer/answer negotiated minimum or maximum limits (depending on individual SDP property semantics). Some needed or wanted codec property changes will also motivate to re-negotiate the SDP, but the scope of this specification intends to cover only changes that lies within the SDP negotiated set and thus do not impact the SDP.

Three message types are defined to support the solution; a request, a notification, and a status report:

**Request**: A media receiver requesting a media sender to adjust one or more of it’s media encoding parameters for a certain media stream. The request is normally based on a specific set of media encoding parameters that the media sender has explicitly notified the media receiver about in a notification.

**Notification**: A media sender notifying a media receiver of the currently used media encoding parameters for a certain (identified) media stream. The notification is initiated by the media sender, typically whenever the media encoding parameters changed significantly from what was previously used. The reason for the change can either be local to the media sender (user, end-point or network), or it can be the result of one or more requests from remote end-points.

**Status Report**: A media sender reporting to a request sender (media receiver) on request reception status; which specific request from the media receiver that was received and considered in setting current media encoding parameters, and the identification of the
media stream that is considered to fulfill the request. The status report can also indicate various error conditions, such as reception of invalid or failing requests.

More details about the individual messages, but still on an overview level, can be found in sub-sections below. To do that, some other aspects need to be described first.

6.1. Message Structure

A COP message is sent from an RTP session participant in it’s role either as media receiver or media sender. Each message can contain one or more message items of one or more message types, all originating from a single media source.

The individual message items each relate only to a single operation point, describing part of an atomic notification or request.

The general structure is outlined below:

```
+--------------------------------------+
| AVPF PSFB FMT="COP"                |
| SSRC of Packet Sender              |
| SSRC of Media Source               |
+----------------------------------+|
| COP Message Item 0                 |
| +----------------------------------+|
| (Codec Configuration Parameters)   |
| +----------------------------------+|
+----------------------------------+|
| COP Message Item 1                 |
| +----------------------------------+|
| (Codec Configuration Parameters)   |
| +----------------------------------+|
| ...                                |
```

Figure 6: COP Message Structure

Note that the Request is the only COP Message Item defined in this specification that is sent in the media receiver role and makes use of "SSRC of Media Source" as the targeted media stream for the Request. Both the Notification and the Status Report Message Items are sent in the media sender role, reporting on the message sender’s own configuration and thus relate only to the "SSRC of Packet Sender", being agnostic to the "SSRC of Media Source" field.
It is thus for example possible to co-locate COPS and COPN messages for the same media source in the same COP FCI. It is also possible to co-locate one or more COPR referring to a single "SSRC of Media Source" with one or more COPN and/or COPS relating to a single "SSRC of Packet Sender" within a single COP message.

Multiple Message Items of the same type in the same COP Message are used to describe a notification, status or request for a media stream containing multiple Operation Points (Section 6.3).

Multiple COP messages are needed to be able to refer to multiple different "SSRC of Packet Sender" and/or "SSRC of Media Source".

6.2. Codec Configuration Parameter Use

The Codec Configuration Parameters that are applicable to a certain codec may be specific to the media type (audio, video, ...), but may also be codec-specific. Some codec properties (described by Codec Configuration Parameters) have to be explicitly enabled by (non-RTCP based) capability signaling to be possible or permitted to use.

An end-point implementing this specification need not support all available Codec Configuration Parameters defined herein or in extensions to this specification. A certain parameter could also be uninteresting for a certain codec or media stream, even if it is generally supported by the end-point. This specification therefore defines capability signaling that allows a COP receiver to declare explicit support per parameter type on a per-codec level. The set of Codec Configuration Parameters that can be used for a certain media stream by a COP sender is thus restricted by the combination of applicability, capability signaling and explicit receiver parameter support signaling.

Any Codec Configuration Parameter that is applicable and feasible to use, but is not included as part of an Operation Point, has a default value. This default is defined for each Parameter Type, but should preferably whenever possible be taken from capability signaling. It is not necessary to use all defined Parameter Types in a media stream description. Some Parameter Types can, depending on media type or codec, either be un-interesting or not possible to describe or control in detail, in which case they can be left out, meaning that the effective value is "undefined" within the limits set by capability signaling (outside the scope of this specification).

6.3. Operation Point

The Codec Configuration Parameters contained in a single Message Item jointly constitutes a description of an Operation Point for a
specific media stream from a media sender.

For the purpose of COP signaling, each such Operation Point is identified with an ID number, OPID, which is scoped by the media sender’s RTP SSRC identification, and can be chosen freely by the media sender. The need for this media sub-stream identification basically only appears with scalable coding or other media encoding methods that introduces separable and configurable sub-streams within the same SSRC. An OPID thus refers to such configurable sub-stream, described by a set of related Codec Configuration Parameters.

Figure 7: Relation of OPID to Media Source, RTP session and SSRC

The above Figure 7 depicts the possible relations between media sources, RTP sessions, RTP streams (SSRCs) and their sub-streams and the OPID.

For example, a single video camera may be encoded using SVC for a combined SST and MST transmission configuration. In that case some subset of scalability layers are sent as SST in the first RTP session using SSRC2. Another set of scalability layers are transported in the second RTP session as another SST using SSRC3. The RTP packet stream from each SSRC can thus contain several sub-streams, each identified with its own OPID. As a result, a single media source is present in two RTP sessions, using two different SSRCs (2 and 3) containing a total of five sub-streams (OPID 3 to 7).

Since an Operation Point can be expected to change over time, as a result of media receiver requests (Section 6.4), resulting from local media sender considerations (Section 6.5), or both, the Operation Point (OPID) is version-handled. The version is scoped by SSRC and OPID.

It is expected that all encoders dividing a media stream into sub-
streams will include some means to identify those sub-streams in the media stream. However, it is also expected that such identification is in general codec-specific. There is thus at times a need to map the codec agnostic COP OPID identification to codec specific identification, and this specification therefore includes a method for such mapping (Section 10).

6.4. Request

The request is sent by a media receiver, which can be either an endpoint or a middle node such as an RTP Mixer. The receiver of the request may similarly be either the original media sender or a RTP Mixer. Included in the request is a description of the desired codec configuration for a specific media (sub-)stream. The parameter values communicated in a notification (Section 6.5) of that (sub-)stream is taken as a starting point when deciding what parameters and parameter values to choose for the request, and only parameters with changed values need to be in the request. The media receiver can of course also use other sources of information when choosing parameters and values, such as for example observation of the received media stream and capability signaling.

It is not an absolute requirement to have received a notification to be able to create a meaningful request. The request can include a set of changed properties for existing streams, but it can also request the addition or removal of one or more media sub-streams having certain properties, in which case there will be no notification to base the request on. A media receiver may also want to send a request prior to having received any notifications for existing streams, and can then base the request on other information such as for example observing the media stream or use information from the capability signaling. In case there is no existing stream and OPID to refer in the request, a "provisional" OPID MUST be chosen in the request, which will have to be mapped back to an existing (sub-)stream and "real" OPID through methods defined in this specification (Section 10).

The media sender receiving a specific request is not required to re-configure the encoder accordingly, even if it should try to do so, but is allowed to take other (previous or concurrent) requests and any local considerations into account, possibly modifying some of the parameter values, or even totally rejecting the request if it is not seen as feasible. It is thus not possible for a media receiver to uniquely see from the media stream or even from a notification if the media sender received the request or if the request was lost and needs to be re-sent.

A request should typically be based on a certain notification, but
there may be situations where a request is sent approximately simultaneously with a new notification for the same stream. In that case, there is a risk that the request is based on the wrong set of codec properties compared to the new notification. It is therefore necessary to have the set of codec properties, identified by an OPID, be version controlled. If a notification announces a specific version of the operation point, where the version is updated every time it is changed, the request can refer to that specific version and any mis-reference can be clearly identified and resolved. In addition, it allows for easy identification of repeated notifications and requests, simply by checking the operation point identification and the version, and without having to parse through all of the codec properties to see if any one changed.

6.5. Notification

The notification is sent by a media sender and describes a media stream or sub-stream in terms of a defined, finite set of codec properties. That same set of codec properties can also be used in a request (Section 6.4). The notification and a common set of defined properties is important to a media receiver since it is rarely possible to see from the media stream itself what controllable properties were used to generate the stream. The set of codec properties and their values used to describe a certain media stream at a certain point in time is henceforth called a codec configuration. Each Operation Point in this codec configuration is implemented using a certain RTP Payload Type, defined by capability signaling outside the scope of this specification.

It must be possible for a media sender to change codec configuration not only based on requests from media receivers, but also based on local limitations, considerations or user actions. This implies that the notification must be possible to send standalone and not only as a response to a request. To avoid that media receivers have to guess what codec configuration is used, a media sender should always send notifications whenever codec configuration for a stream changes. Loss of a notification should anyway not be critical since a media receiver could either fall back to infer approximate codec configuration from the media stream itself, or simply wait with a request until the next notification is sent.

A notification can potentially contain a large amount of codec properties. However, parameters that are not enabled by codec and COP capability signaling, or inherently not part of the used codec will not be included. The notification only describes the currently used codec configuration, and each parameter in an operation point will thus be described by a single value. To further limit the amount of properties that needs to be sent, it is possible to rely on
parameter defaults (listed by individual parameter type definitions) whenever those values are acceptable.

The media receiver could want to take some local action at the time when the codec configuration in the media stream changes. Using the same reasoning as above, this may not be possible to see from the media stream itself. This functionality is explicitly enabled by inclusion of an RTP Time Stamp in the notification, where the Time Stamp describes a time (possibly in the future) when the media stream codec configuration is (estimated to be) effective.

6.6. Status Report

The status report is sent by a media sender and is needed to confirm reception of a specific request OPID to avoid unnecessary retransmission of requests. Loss of a status report will likely trigger a request retransmission, except when the request sender can infer from the media stream or a notification that the stream is now acceptable.

The status report is not a required acknowledgement of every request, but instead reports on the last received request, identified by a request sequence number in addition to the OPID. That de-coupling of request and status report reduces the needed amount of status reports in case of frequently updated requests and/or lack of resources to send status reports.

If a request is somehow not acceptable to a media sender, the status report can also indicate failure and a reason for that failure.

In case the OPID in the request is a "provisional" OPID (Section 6.4), the status report responds with that exact OPID, but also includes a reference to a "real" media (sub-)stream identification or OPID that the media sender considers appropriate for the request.

No description of any codec configuration is included in a status report, even if the corresponding request was successful. Used codec configuration is only carried in the notification (Section 6.5) message. Multiple status reports targeted for multiple request senders can through media (sub-)stream identification and OPID point to the same notification message, reducing the need to repeat applicable codec configuration parameters with every accepted request.
6.7. Adding and Removing Operation Points

A media sender can unilaterally create a new Operation Point by simply selecting a free OPID identifier and use COPN to announce it.

To remove an Operation Point, the media sender simply stops announcing it in COPN. This procedure can be used both for entire media streams containing a single Operation Point and to add/remove sub-streams in media streams containing multiple Operation Points.

The media receiver can request a new Operation Point to be created by using a COPR with an unused identifier and a by setting a flag to indicate that this requests a new OPID. The media sender then decides if it honors the request or not, and announces the new OPID as described above.

The media receiver can indicate that it is no longer interested in receiving an Operation Point corresponding to a media sub-stream by not including any COPR Message Item for it in a single COP Message. The media receiver can indicate a wish to continue to receive an unmodified Operation Point using a COPR without any codec properties (no change).

7. Codec Control Message Extension

This specification specifies a new feedback message, COP, for codec control of real-time media, as an extension to the AVPF [RFC4585] and CCM [RFC5104] specifications. The AVPF specification outlines a mechanism for fast feedback messages over RTCP, which is applicable for IP based real-time media transport and communication services. It defines both transport layer and payload-specific feedback messages. This specification targets the payload-specific type, since a certain codec is typically described by a payload type.

AVPF defines three and CCM defines four payload-specific feedback messages (PSFB). All AVPF and CCM messages are identified by means of the feedback message type (FMT) parameter. This specification specifies one additional payload-specific feedback message.

One new PSFB FMT value is assigned in this specification:

TBA1: Codec Operation Point (COP)

This section defines the feedback message structure, message items and their semantics with the exception of the actual codec configuration parameters which are defined in the next section (Section 8).
7.1. COP Message

The COP message is a payload-specific AVPF CCM message identified by the PSFB FMT value listed above. It carries one or more COP Message Items, each with either a request for, a description of a certain "Operation Point"; a set of codec parameters, or a request status indication.

Not all Message Items makes use of the "SSRC of media source" in the common packet header. "SSRC of media source" SHALL be set to 0 if no Message Item that makes use of it is included in the FCI.

7.2. FCI Format

The COP FCI MUST contain one or more Codec Operation Point Message Items. The maximum number of COP Message Items in a COP message is limited by the [RFC4585] Common Packet Format 'length' field.

The definition of the AVPF feedback message format mandates that the FCI part is a multiple of 32-bit words. The below defined message items will not be 32-bit word aligned. Therefore it is sometimes necessary to insert one to three padding bytes at the end of the FCI. The number of padding bytes are determined by a receiver by comparing the sum of the message items and the feedback message length fields. The padding byte MUST be set to zero (0) and ignored on reception.

7.2.1. Message Item Format

All Codec Operation Point Message Items share a common header format:

```
+----------------------------------+
|       OPID          |N|   Version   |
|                    |   |             |
+----------------------------------+
```

Figure 8: COP Message Item Header Format

The message header fields are:

Type (3 bits): Message Item Type. Three item types are defined in this specification, COPR, COPN and COPS, with values as listed in Table 1 below. More item types MAY be defined in extensions to this specification. Message items with a type field that has an unknown value SHALL be ignored by the receiver.
Payload Length (13 bits): The total length in bytes of all data belonging to this message, following the Message Item Header, i.e. anything following the Version field.

OPID (8 bits): Operation Point ID. Some (typically scalable) codecs are capable of encoding into multiple simultaneous operation points using the same SSRC, and each operation point can then be referenced by OPID. MUST be unique within the scope of an SSRC when N flag is not set. MUST be set to 0 for message items not using the field. See also Section 7.2.3.

N (1 bit): A "New OPID" flag, indicating that the OPID value is chosen arbitrarily and is not meant to refer to any existing Operation Point. The message sender SHOULD NOT use an already known OPID in combination with the N flag. See also individual Message Item definitions.

Version (7 bits): Referencing a specific version of the Codec Configuration identified by the OPID.

7.2.2. Message Item Types

The Message Types defined in this specification are:

| Value | Message Item Type                      |
|-------+---------------------------------------|
| 0     | Codec Operation Point Notification (COPN) |
| 1     | Codec Operation Point Request (COPR)   |
| 2     | Codec Operation Point Status (COPS)    |
| 3-6   | Unassigned                             |
| 7     | Reserved for future extensions         |

Table 1: Message Item Type Values

Each Message Type defined in this specification is described in detail in subsequent sections.

7.2.3. Operation Point Identification

All RTP media streams belonging to the same session can per definition be identified by the SSRC. However, identification of any sub-streams contained in the same RTP media stream (SSRC) needs to use some other identification method, scoped by the SSRC. This is the case for a media stream containing more than one Operation Point, like for example SVC [RFC6190] streams being sent using Single Stream Transport (SST) RTP packetization.
The encoding of and restrictions for such sub-stream (Operation Point) identification will in general be codec specific. Therefore, the OPID used in this specification is merely an SSRC-unique identification number. It is however necessary to create a mapping between this generic number and the codec specific sub-stream identification that can be found in the media stream. This mapping is achieved by including the ID Parameter (Section 8.3) in a Message Item carrying a certain OPID.

In Section 10, codec specific ID Parameter formats are defined for a few of the most common codecs that supports scalability.

7.3. Codec Operation Point Notification

7.3.1. Message Format

The COPN-specific message fields are (see also Message Item Format (Section 7.2.1)):

Type (3 bits):  Set to 0, as listed in Table 1.

OPID (8 bits):  The OPID which is described by the Codec Configuration Parameters.

N (1 bit):  Not used by COPN and SHALL be set to 0 by senders.

Version (7 bits):  Referencing a specific version of the Codec Configuration identified by the OPID. SHALL be increased by 1 modulo 2^8 whenever the used Codec Configuration referenced by the OPID is changed. A repeated message SHALL NOT increase the Version. The initial value SHOULD be chosen randomly.
Transition Time Stamp (32 bits): The RTP Time Stamp value when the listed Codec Configuration Parameters will be effective in the media stream, using the same time line as RTP packets for the referenced SSRC (media sender SSRC). The Time Stamp value MAY express either a time in the past or in the future, and need not map exactly to an actual RTP Time Stamp present in an RTP packet for that SSRC. The same timestamp value SHOULD be used for subsequent transmissions of the identical set of Codec Configuration Parameters for the same OPID and version.

R (1 bit): Reserved. MUST be set to 0 by senders and MUST be ignored by receivers implementing this specification. MAY be defined differently by extensions to this specification.

Payload Type (7 bits): SHALL be identical to the RTP header Payload Type valid for the (sub-)stream described by this OPID.

Codec Configuration Parameters (variable length): Contains zero or more TLV carrying Codec Configuration Parameters as defined in Parameter Types (Section 8).

7.3.2. Semantics

This message is used to inform the media receiver(s) about used Codec Configuration Parameters at the media sender. The available Codec Parameter Types that can be used to describe the Codec Configuration are defined in Section 8.

Some codecs may have clear inband indications in the encoded media stream of how one or more of the Codec Configuration Parameters are configured. For those codecs and Codec Configuration Parameters, COPN is not strictly necessary. Still, for some codecs and / or for some Codec Configuration Parameters, it is not unambiguously possible to see individual Codec Configuration Parameter Values from the encoded media stream, or even possible to see some Codec Configuration Parameters at all, motivating use of COPN.

COPN SHOULD be scheduled for transmission when it becomes known that there are media receivers in the RTP session that did not yet receive any Codec Configuration Parameters for an active Operation Point, or whenever the effective Codec Configuration Parameters has changed significantly, but MAY be scheduled for transmission at any time. The media sender decides what amount of change is required to be considered significant.

The reason for a Codec Configuration Parameter change can either be local to the sending terminal, for example as a result of user interaction or some algorithmic decision, or resulting from reception
of one or more COPR messages (Section 7.4).

If a media sender can no longer fulfill the established Codec Configuration Parameter restrictions of a Operation Point that was previously described by a COPN, it MAY change any Codec Configuration Parameter or even remove the entire Operation Point, and SHOULD then signal this at the earliest opportunity by sending an updated COPN to the media receiver(s).

An OPID can implicitly be indicated as no longer being used by omitting that OPID from the set of COPN message items in the COP PSFB message. All OPIDs that the media sender intends to use at the latest time indicated by any transition timestamp value in the set of COPN present in the COP PSFB message, MUST be included in that COP message.

All Operation Points referred by a COPS (Section 7.5) SHOULD also be detailed by a COPN message contained in the same or in a subsequent COP feedback message, even if the Operation Point did not change significantly from previous COPN.

Note that the OPID Version of that COPN, subsequent to COPS, will be equal or larger than the Version indicated in the COPS. The Version difference may be larger than one (taking field wraparound into account) depending on the number of updated COPN sent since the COPR that triggered the COPS. See also description of those messages below.

Note: COPN may be seen as a more explicit and elaborate version of the TSTN message of [RFC5104] and most of the considerations detailed there for TSTN also apply to COPN.

7.3.2.1. Parameters

The media sender decides what Codec Configuration Parameters to use in the COPN to describe an Operation Point. It is RECOMMENDED that all Codec Configuration Parameters that were accepted as restrictions based on received COPR messages are included. All Codec Configuration Parameters significantly more restrictive than implicit or explicit restrictions set by capability signaling (outside the scope of this specification) SHOULD also be included. Any Codec Configuration Parameter that are either not applicable to the Payload Type or not enabled by capability signaling MUST NOT be included. All Codec Configuration Parameters not covered by the above restrictions MAY be included.

When the Operation Point has dependency to other Operation Points (such as in scalable coding), the values to use for Codec

Configuration Parameters MUST describe the result when all dependencies are utilized. For example, assume an Operation Point describing a base layer with 15 Hz framerate, and a dependent Operation Point describing an enhancement layer adding another 15 Hz to the base layer, resulting in 30 Hz framerate when both layers are combined. The correct Parameter value to use for that latter, dependent "enhancement" Operation Point is 30 Hz, not the 15 Hz difference.

The value of a Codec Configuration Parameter that was not included in a COPN message SHOULD either be inferred from other signaling, e.g. session setup or capability negotiation, outside the scope of this specification, or if such signaling is not available or not applicable, use the default value as defined per Parameter Type (Section 8).

An Operation Point describes one specific setting of Codec Parameters, and a COPN Message therefore MUST NOT include the ALT Parameter Type (Section 8.2) in the Codec Parameters describing the Operation Point.

7.3.2.2. Relation to COPR

To limit RTCP bandwidth and avoid bandwidth expansion, COPN is not mandated as response to every received COPR (Section 7.4).

A media sender implementing this specification SHOULD take requested Operation Points from COPR messages into account for future encoding, but MAY decide to use other Codec Configuration Parameter Values than those requested, e.g. as a result of multiple (possibly contradicting) COPR messages from different media receivers, or any media sender policies, rules or limitations. Thus, a COPN message Operation Point MAY use other Codec Configuration Parameters and other values than those requested in a COPR.

The media sender SHOULD try to maintain OPIDs between COPR and COPN when COPR sender suggests a new OPID value (N flag is set) in the COPR, but MAY use another OPID in COPN. Examples where other OPID values have to be chosen are for example when the suggested OPID conflicts with an already existing OPID, or when the media sender decides that a the suggested new OPID can be fulfilled by an already existing OPID.

Even if a COPR references an existing OPID (N flag cleared), the media sender may have to take other aspects than a specific COPR into account when choosing how many Operation Points to use, and the exact contents of those Operation Points. See the description on COPS (Section 7.5) on how to achieve mapping between a suggested new OPID
and what OPID will actually be used.

When OPID cannot be kept the same between COPN and COPR, the mapping
SHALL be done using identical ID Parameters (Section 8.3) in the COPS
and COPN resulting from the COPR. Further details are described in
the section on COPS (Section 7.5).

Since COPR references a certain COPN OPID, Version, and COPN is send
unreliably and may be lost, COPN senders MUST keep at least the two
last COPN Versions for each SSRC, OPID tuple and SHOULD keep at least
four.

7.3.3. Timing Rules

The timing follows the rules outlined in section 3 of AVPF [RFC4585].
This notification message may be time critical and SHOULD be sent
using early or immediate feedback RTCP timing, but MAY be sent using
regular RTCP timing.

A typical example when regular RTCP timing can be appropriate is when
the sent media stream is further restricted from what was described
by the most recent COPN, which should not cause any problems in the
media receivers. Similarly, it is likely appropriate to use early or
immediate timing when effective media stream restrictions urgently
needs to be removed, which may require media receivers to increase
their resource usage.

7.3.4. Handling in Mixers and Translators

Any media sender, including Mixers and Translators, that sends RTP
media marked with it’s own SSRC and that implements this
specification SHALL also be prepared to send COPN, even if it is not
the originating media source. As a result of that, such media sender
may have to send updated COPN whenever the included media sources
(CSRC) changes, subject to rules laid out above (Section 7.3.2).
Note that this can be achieved in different ways, for example by
forwarding (possibly cached) COPN from the included CSRC when the
Mixer is not performing transcoding.

In cases where a Mixer or Translator needs to forward a COPR from one
side (A) to the other (B) (as described in Section 7.4.4), the COPN
sent to the A side MAY need to be delayed until the Mixer or
Translator has received a corresponding COPN from the B side, as
indicated in Figure 10 below.
If a Mixer or Translator has decided to act partially (modify the media stream with respect to some Parameter Types, but not all) on a received COPR from the A side, and a COPN is received from the B side indicating that the current media modifications are no longer necessary, the mixer or translator SHOULD cease it’s own actions that are no longer needed. It SHOULD then also issue a COPN describing the new situation to the A side, as indicated in Figure 11 below.

7.4. Codec Operation Point Request

7.4.1. Message Format

The COPR-specific message fields are:

Type (3 bits): Set to 1, as listed in Table 1.
OPID (8 bits): The OPID this request refers to for an existing OPID, and an arbitrarily chosen but unique value in requests for new operations points, i.e. with the N flag set.

N (1 bit): MUST be set to 0 when OPID references an existing OPID announced in a COPN received from the targeted media sender, and MUST be set to 1 otherwise.

Version (7 bits): When N flag is not set (0), referencing a specific version of the Codec Configuration identified by the OPID in a COPN received from the targeted media sender. Not used and MUST be set to 0 when N flag is set (1).

Sequence No (8 bits): Sequence Number. SHALL be incremented by 1 modulo 2^8 for every COPR that includes an updated set of requested Codec Configuration Parameters described by the same OPID and Version as was used with the previous Sequence Number. Sequence Number SHALL be kept unchanged in repetitions of this message. Initial value SHOULD be chosen randomly.

Codec Configuration Parameters (variable length): Contains zero or more TLV carrying Codec Configuration Parameters as defined in Parameter Types (Section 8).

7.4.2. Semantics

This Message Item is sent by a media receiver wanting to control one or more Codec Configuration Parameters of the targeted media sender. The requested values MUST stay within the media capability negotiated by other means than this specification. The available Codec Configuration Parameters that can be controlled are listed in Section 8.

Note: COPR may be seen as a more explicit and elaborate version of the TSTR message of [RFC5104] and most of the considerations detailed there for TSTR also apply to COPR.

7.4.2.1. Sender Behavior

If at least one COPN (Section 7.3) is received for the targeted stream, the Codec Configuration Parameters for that stream (SSRC) with defined OPID and Version are known to the COPR sender. The COPR MUST refer to the OPID and Version of the most recently received COPN (if any) for the targeted stream. Since it references a defined set of Codec Configuration Parameters from a COPN, the COPR SHOULD only include the Codec Configuration Parameters it wishes to change in the message, but it MAY include also unchanged Codec Configuration Parameters.
If no COPN is received for the targeted stream, the COPR sender MUST choose an arbitrary OPID and set the N flag to indicate that the OPID does not refer to any existing Operation Point. In this case the Version field is not used and MUST be set to 0. The OPID value SHALL NOT be identical to any OPID from the same media source that the media receiver is aware of and has received COPN for. Since in this case no COPN reference exist, the COPR sender SHOULD include all Codec Configuration Parameters that it wishes to include a specific restriction for (other than the default). Note that for some codecs, some Codec Configuration Parameters may be possible to infer from the media stream, but if the wanted restriction includes also those and lacking a describing COPN, they SHOULD anyway be included explicitly in the COPR.

Any Codec Configuration Parameter that are not enabled by capability signaling MUST NOT be included.

A COPR sender MUST increment the SN field modulo 2^8 with every new COPR that includes any update to the Codec Configuration Parameters (referring to a specific version of an OPID compared to the previously sent SN, as long as it does not receive any COPS (Section 7.5) with the same OPID, Version, and SN as was used in the most recently sent COPR. COPR having a later SN MUST be interpreted as replacing any COPR with identical OPID and Version but with lower SN, taking field wrap into account.

A COPR sender that did not receive any corresponding COPS, but did receive a COPN with the same OPID and with a higher Version than was used in the last COPR SHALL re-consider the COPR and MAY send an updated COPR referencing the new Version.

If the capability negotiation has established that a codec supporting scalable operation is used, and if the media receiver wishes to request that scalability is used, it MAY do so by sending multiple COPR with different OPID to the same media sender. The OPID and Version used in such request MAY be based on an existing Operation Point, but it MAY also indicate a desire to introduce scalability into a previously non-scalable stream by choosing a new OPID (indicated by setting the N flag). In any case, the resulting OPIDs and sub-streams are identified through use of the ID Parameter (Section 8.3) in subsequent COPS and COPN. See also the description of COPS (Section 7.5).

An Operation Point without any Codec Configuration Parameters MAY be used and MUST be interpreted as a request to keep the Operation Point unchanged. This is especially useful when modifying some but not all in a set of sub-streams.
When a COPR sender is receiving multiple Operation Points and wants to continue to do so, it MUST include all Operation Points it still wishes to receive in the COPR, also those that can be left unchanged.

An COPR MAY also describe alternative Operation Points that the media sender can choose from, through use of one or more ALT Parameters (Section 8.2).

Since COPR references a specific COPN using SSRC, OPID and Version, a COPR sender typically needs to keep the latest Version of received COPN for each SSRC and OPID, also including the Codec Configuration Parameters.

7.4.2.2. Media Sender Behavior

A media sender receiving a COPR SHOULD take the request into account for future encoding, but MAY also take COPR from other media receivers and other information available to the media sender into account when deciding how to change encoding properties.

A media receiver sending COPR thus cannot always expect that all Parameter Values of the request are fully honored, or even honored at all. It can only know that the COPR was taken into account when receiving a COPS (Section 7.5) from the media sender with a matching OPID, Version and SN.

To what extent a COPR is honored is described by the chosen Codec Configuration Parameter values contained in a subsequent COPN message (Section 7.3) with a later (taking wraparound into account) Version than the one referred by the COPR.

7.4.3. Timing Rules

The timing follows the rules outlined in section 3 of [RFC4585]. This request message MAY be sent using Immediate, Early or Regular timing depending on the application’s needs.

A COPR sender that did not receive a corresponding COPS MAY choose to re-transmit the COPR, without increasing the SN.

When an RTP media receiver (SSRC) is timing out or leaves (BYE received) from the RTP session, it SHALL implicitly imply that all COPR restrictions put by that media receiver are removed.

7.4.4. Handling in Mixers and Translators

A Mixer or media Translator that implements this specification and encodes content sent to the media receiver issuing the COPR SHALL
consider the request to determine if it can fulfill it by changing its own encoding parameters. A Mixer encoding for multiple session participants will need to consider the joint needs of all participants when generating a COPR on its own behalf towards the media sender.

A Mixer or Translator able to fulfill the COPR partially MAY act on the parts it can fulfill (and SHALL then send COPS and COPN accordingly), but SHOULD anyway forward the unaltered COPR towards the media sender, since it is likely most efficient to make the necessary Codec Configuration Parameter changes directly at the original media source.

A media Translator that does not act on COP messages will forward them unaltered, according to normal Translator rules.

7.5. Codec Operation Point Status

7.5.1. Message Format

```
+-----------------------------------------------+----------+----------+----------+
| Type | Payload Length | OPID | N | Version |
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|
| +-----------------------------------------------+----------|----------|----------|

Figure 13: COPS Format
```

The COPS-specific message fields are:

Type (3 bits): Set to 2, as listed in Table 1.

OPID (8 bits): MUST be set identical to the same field in the COPR being reported on.

N (1 bit): MUST be set identical to the same field in the COPR being reported on.

Version (7 bits): MUST be set identical to the same field in the COPR being reported on.
SSRC of COPR sender (32 bits): MUST be set identical to the SSRC of packet sender field in the common AVPF header part of the COPR being reported on.

Sequence No (8 bits): MUST be set identical to the same field in the COPR being reported on.

RC (3 bits): Return Code. Indicates degree of success or failure of the COPR being reported on, as described in Table 2.

Reason (5 bits): Contains more detailed information on the reason for success or failure, as described in Table 3 or extensions to this specification.

Codec Configuration Parameters (variable): MAY contain an ID Codec Configuration Parameter providing codec specific media identification of the OPID, subject to conditions outlined in the text below, or MAY be empty.

7.5.2. Semantics

The COPS Message Item indicates the request status related to a certain SSRC OPID tuple by listing the latest received COPR (Section 7.4) SN. It effectively informs the COPR sender that it no longer needs to re-send that COPR SN (or any previous SN).

COPS indicates that the specified COPR was successfully received by the media sender targeted in the request. If the COPR suggested Codec Configuration Parameters could be understood (Table 2), they may be taken into account, possibly together with COPR messages from other receivers and other aspects applicable to the specific media sender. The Return Code carries an indication to which extent the COPR could be honored.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
</tr>
<tr>
<td>1</td>
<td>Partial success</td>
</tr>
<tr>
<td>2</td>
<td>Failure</td>
</tr>
<tr>
<td>3-6</td>
<td>Unassigned</td>
</tr>
<tr>
<td>7</td>
<td>Reserved for future extension</td>
</tr>
</tbody>
</table>

Table 2: Return Code Values

A Success Return Code indicates that the resulting media configuration is fully in line with the COPR.
A Partial Success Return Code indicates that the resulting media configuration is not fully in line with the COPR, but that the media sender regards the COPR to be sufficiently well represented by one or more of the existing Operation Points.

A Failure Return code indicates that the media sender failed to take the COPR into account, either due to some error condition or because no media stream could be created or changed to comply.

The Reason Values defined below are independent of Return Code, but all reasons may not be meaningful with all return codes. More reasons MAY be defined in extensions to this specification.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
</tr>
<tr>
<td>1</td>
<td>Unknown OPID</td>
</tr>
<tr>
<td>2</td>
<td>Too many Operation Points</td>
</tr>
<tr>
<td>3</td>
<td>Request violates capability limits</td>
</tr>
<tr>
<td>4</td>
<td>Too old Operation Point Version</td>
</tr>
<tr>
<td>5</td>
<td>Unknown Parameter Type</td>
</tr>
<tr>
<td>6</td>
<td>Parameter Value too long</td>
</tr>
<tr>
<td>7</td>
<td>Invalid Comparison Type</td>
</tr>
<tr>
<td>8</td>
<td>One or more parameter values in the request were changed</td>
</tr>
<tr>
<td>9-31</td>
<td>Unassigned</td>
</tr>
</tbody>
</table>

Table 3: Reason Values

COPS is typically sent without any Codec Configuration Parameters. When the N flag was set in the related COPR, a non-failing COPS MUST include an ID Parameter (Section 8.3) identifying the actual sub-stream that the media sender considers applicable to the COPR. The OPID used by that sub-stream can be found through examining ID Parameters of subsequent COPN from the same media source for ID values matching the one in COPS.

Senders implementing this specification MUST NOT use any other Codec Configuration Parameter Types than ID in a COPS message. The contained ID Parameter points to the specific media (sub-)stream that the media sender regards as applicable to the COPR.

When a COPR receiver has received multiple COPR messages from a single COPR source with the same OPID but with several different values of Version and/or SN, and for which it has not yet sent a COPS, it SHALL only send COPS for the COPR with the Highest SN, taking field wrap of those two fields into account.
7.5.3. Timing Rules

COPS SHALL be sent at the earliest opportunity after having received a COPR, with the following exception:

A media sender that receives a COPR with a previously received OPID, Version, and SN closely after sending a COPS for that same OPID, Version, and SN (within 2 times the longest observed round trip time, plus any AVPF-induced packet sending delays), SHOULD await a repeated COPR before scheduling another COPS transmission for that OPID, Version, and SN.

The exception is introduced to avoid unnecessary COPS transmission when there is a chance that already sent COPS or COPN may satisfy or invalidate the COPR.

7.5.4. Handling in Mixers and Translators

A Mixer or media Translator that implements this specification, encoding content sent to media receivers and that acts on COPR SHALL also report using COPS, just like any other media sender. An RTP Translator not knowing or acting on COPR will forward all COP messages unaltered, according to normal RTP Translator rules.

8. Parameter Types

This section defines the general Codec Configuration Parameter (CCP) TLV format. Then a number of different parameter formats are defined. It is expected that a number of additional CCPs will be defined in the future as the needs of different codecs are explored or developed.

8.1. Parameter Format

COP Message Items MAY contain one or more Codec Configuration Parameters, encoded in TLV (Type-Length-Value) format, which SHOULD then be interpreted as simultaneously applicable to the defined Operation Point. Parameter Values MUST be byte-aligned.
<table>
<thead>
<tr>
<th>ParamType</th>
<th>C</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Value</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 14: Codec Parameter Format

**ParamType (8 bits):** The Codec Configuration Parameter Type, encoded as defined in Table 4 and possible extensions to this specification. A parameter with an unknown ParamType SHALL be ignored on reception in a COPN and SHALL either be reported as unknown in COPS or be ignored when received in COPR.

**C (2 bits):** Comparison Type, encoded as defined in Table 5, unless specified otherwise by individual ParamType definitions. The Comparison Type specifies what type of restriction the Codec Configuration Parameter Value expresses and how it should be compared to other Codec Configuration Parameter Values of the same ParamType.

- **Exact:** The Parameter Value is an exact value, and no other values are acceptable. MUST NOT be used together with any other Comparison Types for the same ParamType.
- **Minimum:** The Parameter Value is an inclusive minimum restriction. MAY be used together with Maximum and/or Target Comparison Types for the same ParamType. If no minimum restriction is specified, no specific minimum restriction exist.
- **Maximum:** The Parameter Value is an inclusive maximum restriction. MAY be used together with Minimum and/or Target Comparison Types for the same ParamType. If no maximum restriction is specified, no specific maximum restriction exist.
- **Target:** The Parameter Value is a preferred target value, but other values within a specified range are acceptable. This type MUST be used together with at least one of Minimum and Maximum Comparison Types for the same ParamType. If no target is specified, no specific preference exist.
Length (6 bits): The Parameter Value Length in bytes, excluding the ParamType and the Length field itself. A Length of 0 indicates that the parameter has no value, effectively constituting a wildcarded parameter that can take on any value (expresses no specific restriction). This is also the RECOMMENDED way to explicitly remove a previously effective restriction.

Parameter Value (variable length): The actual parameter value, encoded in a format defined by the specific ParamType definition.

The meaning of Multiple Codec Configuration Parameters with the same ParamType and the same Comparison Type included as part of the same Operation Point is undefined and SHALL NOT be used.

A Codec Configuration Parameter that is encoded in a way (including incorrectly) that cannot be interpreted by the receiver SHALL be ignored.

The below parameters encoded as signed or unsigned integers uses a variable size representation in the value field. It is RECOMMENDED to only include the minimal number of bytes necessary to represent the value that is to be included in the parameter TLV. The length field in the parameter TLV will explicitly indicate how many bytes are present in the value field. All parameters using a variable size representation of their value MUST define the maximum number of bytes possible to include in the value field.

The ParamType values and the SDP tags (see Section 9) for the Codec Configuration Parameter Types defined in this specification are listed below.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ALT</td>
<td>alt</td>
</tr>
<tr>
<td>1</td>
<td>ID</td>
<td>id</td>
</tr>
<tr>
<td>2</td>
<td>Payload Type</td>
<td>pt</td>
</tr>
<tr>
<td>3</td>
<td>Bitrate</td>
<td>bitrate</td>
</tr>
<tr>
<td>4</td>
<td>Token Bucket Size</td>
<td>token-bucket</td>
</tr>
<tr>
<td>5</td>
<td>Framerate</td>
<td>framerate</td>
</tr>
<tr>
<td>6</td>
<td>Horizontal Pixels</td>
<td>hor-size</td>
</tr>
<tr>
<td>7</td>
<td>Vertical Pixels</td>
<td>ver-size</td>
</tr>
<tr>
<td>8</td>
<td>Channels</td>
<td>channels</td>
</tr>
<tr>
<td>9</td>
<td>Sampling Rate</td>
<td>sampling</td>
</tr>
<tr>
<td>10</td>
<td>Maximum RTP Packet Size</td>
<td>max-rtp-size</td>
</tr>
<tr>
<td>11</td>
<td>Maximum RTP Packet Rate</td>
<td>max-rtp-rate</td>
</tr>
<tr>
<td>12</td>
<td>Frame Aggregation</td>
<td>aggregate</td>
</tr>
<tr>
<td>13-254</td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>255</td>
<td>Reserved for future extension</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Parameter Type Values

The values of the defined Parameter Value Comparison Type are listed below.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Exact</td>
</tr>
<tr>
<td>1</td>
<td>Minimum</td>
</tr>
<tr>
<td>2</td>
<td>Maximum</td>
</tr>
<tr>
<td>3</td>
<td>Target</td>
</tr>
</tbody>
</table>

Table 5: Comparison Type Values

The following sub-sections describe the syntax and semantics of the different Codec Configuration Parameter Types defined in this specification.

Unless explicitly specified in the sub-sections below, or in extensions to this specification, all Parameter Type values are binary encoded unsigned integers, most significant byte first (for multi-byte values).
8.2. ALT

This Codec Parameter Type is a special parameter, separating the Codec Configuration Parameters preceding it from the ones that follow into two separate, alternative Operation Points.

Type Value: 0
Tag: alt
Unit: Not applicable.

Semantics: A special parameter expressing an "alternative" relation between the parameters preceding it and the parameters following it. This SHOULD be interpreted as describing two alternate Operation Points where one and only one SHALL be chosen, with the Operation Point preceding ALT in the parameter list being preferred. Multiple ALT parameters MAY be used in the same parameter list, in which case each set of parameters to evaluate can be either before the first ALT parameter, between two ALT parameters, or after the last ALT parameter. Evaluating from the top of the list and obeying the above preference rule, the first acceptable set of parameters (not containing any ALT parameter) is the one to choose.

Encoding: Not applicable.

Media Types: All.

Value Restrictions: MUST be used with the Length field set to 0. Two ALT parameters MUST be separated by at least one parameter other than ALT.

Default Value: Not applicable.

Comparison Types: MUST be set to 0.

Note:

8.3. ID

This Codec Parameter Type is a special parameter that enables codec specific identification of sub-streams, for example when there are multiple sub-streams in a single SSRC. It can also be used to reference OPID, when the used codec does not support or use sub-streams. When used, it SHALL be listed first among the Codec Parameters used to describe the sub-stream.
Type Value: 1
Tag: id
Unit: Not applicable.

Semantics: A special parameter describing the, possibly codec specific, media identification for the OPID.

Encoding: If used with non-scalable encoding, it MUST contain an OPID (Section 7.2.1). If used with scalable encoding, this codec specific encoding MUST be defined by Section 10. It MUST be defined to occupy an integer number of bytes, where all bits in the bytes are defined as part of the format.

Media Types: All.

Value Restrictions: If used with non-scalable encoding, any OPID restrictions apply. If used with scalable encoding, any restrictions MUST be defined by the definition of the codec specific sub-stream identification definition (Section 10).

Default Value: Not set.

Comparison Types: MUST be set to 0.

Note: MAY be used whenever there is a need to identify an Operation Point in codec native format, or when there is a need to map that against an OPID.

8.4. Payload Type

Type Value: 2
Tag: pt
Unit: Not applicable.

Semantics: Referencing the RTP Payload Type to use for the OPID.

Encoding: The least significant 7 bits MUST use the same encoding as the RTP Payload Type field in the RTP header. The most significant bit MUST be set to 0.

Media Types: All.
Value Restrictions: The same restrictions valid for RTP Payload Type apply, i.e. 7-bit values 0-127. MUST be represented by a single byte in the value field.

Default Value: Not set.

Comparison Types: MUST be set to 0.

Note: MAY be used whenever there is a need to specify Codec Configuration Parameters valid only for a certain RTP Payload Type. What media type, codec and possible parameters that are described by the RTP Payload Type is outside the scope of this specification, but is typically defined in capability or call setup signaling, for example SDP.

8.5. Bitrate

Type Value: 3

Tag: bitrate

Unit: Bits per second.

Semantics: Media level per second average media bitrate, excluding IP/UDP/RTP overhead, but including RTP payload headers (similar to b=TIAS from SDP signaling [RFC3890]), rounded up to the closest integer.

Encoding: Binary encoded unsigned integer, most significant byte first.

Media Types: All.

Value Restrictions: A value of 0 MAY be used. The largest value allowed is what is possible to represent in a 64-bit unsigned integer value, i.e. a value between 0 and 18,446,744,073,709,551,615.

Default Value: Maximum value computed from capability or call setup signaling, e.g. b= parameter from SDP. Note that it is often not possible to achieve more than a rough estimation from such computation.

Comparison Types: All.
Note: This parameter used with a maximum comparison type parameter is significantly similar to CCM Temporary Maximum Media Bit Rate (TMMBR). When being used with a maximum comparison type value of 0, it is also significantly similar to PAUSE [I-D.westerlund-avtext-rtp-stream-pause]. Compared to those, this parameter conveys significant extra information through the relation to other parameters applied to the same Operation Point, as well as the possibility to express other restrictions than a maximum limit. When CCM TMMBR is supported in addition to this specification, the Bitrate parameters from all Operation Points within each SSRC should be considered and CCM TMMBR messages SHOULD be sent for those SSRC that are found to be in the bounding set (see CCM [RFC5104], section 3.5.4.2). When PAUSE is supported in addition to this specification, the Bitrate parameters from all Operation Points within each SSRC should be considered and CCM PAUSE messages SHOULD be sent for those SSRC that contain only Operation Points that are limited by a Bitrate maximum value of 0.

8.6. Token Bucket Size

Type Value: 4

Tag: token-bucket

Unit: Bytes.

Semantics: Media level token bucket [RFC2212] size excluding IP/UDP/RTP overhead, but including RTP payload headers, describing the bitrate variability over time as described in [I-D.westerlund-mmusic-sdp-bw-attribute]. This parameter can be combined with the parameter bitrate (Section 8.5) (above) to provide token bucket fill rate plus bucket size for a complete token bucket model.

Encoding: Binary encoded unsigned integer, most significant byte first.

Media Types: All.

Value Restrictions: A value of 0 is generally not meaningful and SHOULD NOT be used. Values that can be represented using a 32-bit unsigned integer, i.e. 0 to 4,294,967,295.

Default Value: 4096 bytes.
Comparison Types: Maximum, Target.

Note: Changing the token bucket size does not imply changing the average bitrate, it just changes the acceptable average bitrate variation over time.

8.7. Framerate

Type Value: 5

Tag: framerate

Unit: 100th of a Hz. This definition allows e.g. distinguishing between video encoded at 30 Hz (two-byte value 3000) and 29.97 Hz (two-byte value 2997). It also allows for high speed video cameras, like 1000 Hz (three-byte value 100000), and slow-scan down to one frame every 100 seconds (one-byte value 1).

Semantics: The number of media frames to render per second.

Encoding: Binary encoded unsigned integer, most significant byte first.

Media Types: Mainly intended for video and timed image media types, but MAY be used also for other media types.

Value Restrictions: A value of 0 MAY be used, meaning single-frame, request based encoding (request procedure is out of scope for this specification). Values that can be represented using a 32-bit unsigned integer, i.e. 0 to 42,949,672.95 Hz.

Default Value: Maximum allowed by call setup and/or capability signaling, e.g. a=framerate parameter from SDP [RFC4566], or codec-specific configuration.

Comparison Types: All.

Note: A media frame is typically a set of semantically grouped samples, e.g. the relation that a video image has to its individual pixels, or the relation that an audio frame has to individual audio samples. The value applies to encoded media framerate, not the packet rate (Section 8.13) that may also be changed as a result of different Frame Aggregation (Section 8.14).
8.8. Horizontal Pixels

Type Value: 6
Tag: hor-size
Unit: Pixels.
Semantics: Horizontal image size.
Encoding: Binary encoded unsigned integer, most significant byte first.
Media Types: Video and image.
Value Restrictions: The meaning of the value 0 is not defined and SHALL NOT be used.
Default Value: Maximum allowed by call setup and/or capability signaling. Values that can be represented using a 32-bit unsigned integer, i.e. 1 to 4,294,967,295.
Comparison Types: All.
Note: The pixel and picture aspect ratios cannot be changed with this parameter. Video encoders can typically describe both pixel and picture aspect ratios as part of the encoded media stream.

8.9. Vertical Pixels

Type Value: 7
Tag: ver-size
Unit: Pixels.
Semantics: Vertical image size.
Encoding: Binary encoded unsigned integer, most significant byte first.
Media Types: Video and image.
Value Restrictions: The meaning of the value 0 is not defined and SHALL NOT be used. Values that can be represented using a 32-bit unsigned integer, i.e. 1 to 4,294,967,295.
Default Value: Maximum allowed by call setup and/or capability signaling.

Comparison Types: All.

Note: See Note in Section 8.8.

8.10. Channels

Type Value: 8
Tag: channels
Unit: Unit-less.

Semantics: The number of media channels.

Encoding: Binary encoded unsigned integer, most significant byte first.

Media Types: All.

Value Restrictions: The meaning of the value 0 is not defined and SHALL NOT be used. Values that can be represented using a 16-bit unsigned integer, i.e. 1 to 65,535.

Default Value: Taken from call setup or capability signaling, or 1 if no other value is available.

Comparison Types: All.

Note: This Codec Configuration Parameter SHOULD NOT be used if the capability negotiation did not establish that suitable multi-channel coding is supported by both ends. For audio, the interpretation and spatial mapping SHALL follow the one for the indicated payload format. For video, it SHALL be interpreted as the number of views in multi-view coding, where the number 2 SHOULD represent stereo (3D) coding, unless negotiated otherwise by means outside of this specification, e.g. SDP.

8.11. Sampling Rate

Type Value: 9
Tag: sampling
Unit: Hz.

Semantics: Frequency of the media sampling clock in Hz, as input to the codec, per channel (Section 8.10).

Encoding: Binary encoded unsigned integer, most significant byte first.

Media Types: Mainly intended for audio media, but MAY be used for other media types.

Value Restrictions: The meaning of the value 0 is not defined and SHALL NOT be used. Values that can be represented using a 32-bit unsigned integer, i.e. 1 to 4,294,967,295.

Default Value: Taken from call setup or capability signaling, e.g. RTP TS rate from SDP m-line.

Comparison Types: All.

Note: The value refers to the media sample clock, not the media framerate (Section 8.7). It does not specify any codec-internal up- or down-sampling that may take place as part of the encoding process. If multiple channels (Section 8.10) are used and different channels use different sampling rates, then this parameter MUST NOT be used unless there is a known sampling rate relationship and an ordering between the channels, in which case the specified sampling rate value SHALL be taken as applicable to the first channel of the ordered set. The relationship may e.g. be known implicitly by each party through some specification, or be negotiated using other means than this specification. Typically only a limited subset of sampling frequencies makes sense to the media encoder, and sometimes it is not possible to change at all. For video, the sampling rate is very closely connected to the image horizontal (Section 8.8), vertical (Section 8.9) resolution, and framerate (Section 8.7), which are more explicit and meaningful and SHOULD therefore be used instead. For audio, changing sampling rate may require changing codec and thus changing RTP payload type. The actual media sampling rate may not be identical to the sampling rate specified for RTP Time Stamps for that RTP Payload Type. E.g. almost all video codecs use only 90 000 Hz sampling clock for RTP Time Stamps, while the actual pixel sampling clock is typically in the range from a few to several hundred MHz. Also some recent audio codecs use an RTP Time Stamp rate that differ from the actual media sampling rate. Aspects related to mid-stream changes of RTP Time Stamp rate is described in [I-D.ietf-avtext-multiple-clock-rates].
8.12. Maximum RTP Packet Size

Type Value: 10
Tag: max-rtp-size
Unit: Bytes.

Semantics: The maximum size of an RTP packet, including the RTP header but excluding lower layers.

Encoding: Binary encoded unsigned integer, most significant byte first.

Media Types: All.

Value Restrictions: The meaning of a value less than the size of the RTP header (12 bytes for current RTP specification [RFC3550]) is not defined and SHOULD NOT be used. Values that can be represented using a 32-bit unsigned integer, i.e. 0 to 4,294,967,295.

Default Value: 1400 bytes for IPv4, 1280 bytes for IPv6 or if IP version cannot be determined.

Comparison Types: Maximum.

Note: The parameter should typically be used to adapt encoding to a known or assumed MTU limitation, and MAY be used to assist MTU path discovery in point-to-point as well as in RTP Mixer or Translator topologies.

8.13. Maximum RTP Packet Rate

Type Value: 11
Tag: max-rtp-rate
Unit: RTP packets per second.

Semantics: Maximum number of RTP packets per second, calculated or estimated as the largest value appearing during a one-second sliding window, similar to the definition of "maxprate" [RFC3890].

Encoding: Binary encoded unsigned integer, most significant byte first.
Media Types:  All.

Value Restrictions:  The meaning of the value 0 is not defined and SHALL NOT be used.  Values that can be represented using a 32-bit unsigned integer, i.e. 1 to 4,294,967,295.

Default Value:  Not set.

Comparison Types:  Maximum.

Note:  The parameter should typically be used to adapt encoding on a network that is packet rate rather than bitrate limited, if such property is known.  This Codec Configuration Parameter MUST NOT exceed any negotiated "maxprate" [RFC3890] value, if present.

8.14.  Application Data Unit Aggregation

Type Value:  12

Tag:  aggregate

Unit:  Milliseconds.

Semantics:  The amount of non-redundant application data unit (ADU) representing different RTP Time Stamps that should be included in the RTP payload, henceforth in this specification called an "ADU aggregate".  An ADU aggregation value of 1 is equivalent to no aggregation.

Encoding:  Binary encoded unsigned integer, most significant byte first.

Media Types:  Mainly intended for audio, but MAY be used also for other media, e.g. Real-Time Text [RFC4103].

Value Restrictions:  The meaning of the value 0 is not defined and SHALL NOT be used.  Values that can be represented using a 16-bit unsigned integer, i.e. 1 to 65,535.

Value Default Value:  1.

Comparison Types:  All.

Note:  To use this parameter, there MUST exist a defined way of including multiple ADUs into the same RTP payload for the used RTP Payload Type.  There MUST also exist a known internal timing relationship between individual ADUs within the RTP payload for the used RTP Payload Type.  Some payload formats (typically video)
do not allow multiple ADUs (representing different sampling times) in the RTP payload. This Codec Configuration Parameter SHOULD NOT be used unless the "maxrate" [RFC3890] and/or "ptime" parameters are included in the SDP. The requested ADU aggregation level MUST NOT cause exceeding the negotiated "maxrate" value, if present, and SHOULD NOT exceed the negotiated "ptime" value, if present. The requested frame aggregation level MUST NOT be in conflict with any Maximum RTP Packet Size (Section 8.12) or Maximum RTP Packet Rate (Section 8.13) parameters. The packet rate that may result from different frame aggregation values is related to, but semantically not the same as, media Framerate (Section 8.7).

9. SDP Extensions

As described in [RFC4585] and [RFC5104], the rtcp-fb attribute may be used to negotiate capability to handle specific AVPF commands and indications, and specifically the "ccm" feedback value is used for codec control. All rules defined there related to use of "rtcp-fb" and "ccm" also apply to the new feedback message defined in this specification.

9.1. Extension of the rtcp-fb Attribute

In this document, a new "ccm" rtcp-fb-ccm-param is defined, according to the method of extension described in [RFC5104]:

- "cop" indicates support for all COP Message Items defined in this specification, and one or more of the Codec Configuration Parameters defined in this specification

The ABNF [RFC5234] for the new rtcp-fb-ccm-param is:
rtcp-fb-ccm-param = SP "cop" 1*rtcp-fb-ccm-cop-param
; rtcp-fb-ccm-param defined in [RFC5104]

rtcp-fb-ccm-cop-param = SP "alt"
/ SP "id"
/ SP "pt"
/ SP "bitrate"
/ SP "token-bucket"
/ SP "framerate"
/ SP "hor-size"
/ SP "ver-size"
/ SP "channels"
/ SP "sampling"
/ SP "max-rtp-size"
/ SP "max-rtp-rate"
/ SP "aggregate"
/ SP token ; for future extensions
; token defined in [RFC4566]

Figure 15: ABNF for cop

Token values for rtcp-fb-ccm-cop-param are defined in Table 4. Their semantics are described in Section 8.

Supported Parameter Types are indicated by including one or more rtcp-fb-ccm-cop-param.

9.2. Offer/Answer Usage

The usage of Offer/Answer [RFC3264] in this specification inherits all applicable usage defined in [RFC5104].

In order to announce support, and willingness to use, the CCM "cop" feedback message, an offerer or answerer SHALL indicate that capability through the extended SDP rtcp-fb attribute, defined in Section 9.1. The offerer or answerer MUST include a list of the Parameter Types that it is willing to receive.

If an SDP offer does not indicate support of the CCM "cop" feedback message, the answerer MUST NOT indicate support in the associated SDP answer.

The answerer MAY add and/or remove Parameter Types that were not present in the associated SDP offer. If the answerer adds Parameter Types to the SDP answer, it MUST be able to receive such messages, but the answerer MUST NOT send such messages towards the offerer.

If an SDP answer does not indicate support of the CCM "cop" feedback
message, the offerer MUST NOT send such messages towards the
answerer.

The offerer and the answerer SHOULD NOT send any Parameter Types that
the remote party did not indicate receive support for. As described
in Section 8, a parameter with an unknown ParamType SHALL be ignored
on reception in a COPN and SHALL either be reported as unknown in
COPS or be ignored when received in COPR.

Entities MUST list all supported Parameter Types in every subsequent
SDP offer or answer associated with the session. If a Parameter Type
is not listed, it is an indication that the offerer or answerer is no
longer willing to receive such messages within the session.

9.3. Declarative Usage

Declarative use of the CCM "cop" does not differ from the Offer/
Answer usage.

10. Codec Sub-Stream Identification

The defined mechanism is not bound to a specific codec. It uses the
main characteristics of a chosen set of media types, including audio
and video. To what extent this mechanism can be applied depends on
which specific codec is used.

When using a codec that can produce separate sub-streams within a
single SSRC, those sub-streams can only be referred with a COP O PID
if there is a defined relation to the codec-specific sub-stream
identification. This is accomplished in this specification by
defining an ID Parameter format using codec-specific sub-stream
identification for each such codec.

If such sub-streams have dependencies, the O PID describes the
characteristics of the sub-stream including all it’s dependencies,
but excluding any sub-streams that are dependent on this sub-stream.
The sub-stream identification describes a single, payload specific
node in a dependency tree, and does in general not include any
identification of the sub-streams it depends on, or the dependency
structure between sub-streams. Any dependency structure must thus be
described by the media stream payload format and is out of scope for
this specification.

This section contains ID Parameter format definitions for a few
selected codecs. The format definitions MUST use an integer number
of bytes and MUST define all bits in those bytes. Note, the ID
parameter is interpreted in the context of a given SSRC and a
specific RTP payload type.

Extensions to this specification MAY add more codec-specific definitions than the ones described in the sub-sections below. Such definitions made in extensions to this specification SHOULD be considered as an integrated part of this section, with respect to usage with other mechanisms defined in this specification.

10.1. H.264 AVC

Some non-scalable video codecs such as H.264 AVC [H264] and corresponding RTP payload format [RFC6184] can accomplish simultaneous encoding of multiple operation points. H.264 AVC can encode a video stream using limited-reference and non-reference frames such that it enables limited temporal scalability, by use of the nal_ref_id syntax element.

The ID Parameter Type is defined below:

```
0
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+
|  Reserved | N |
+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 16: ID Definition for AVC

Reserved (6 bits): Reserved. SHALL be set to 0 by senders and SHALL be ignored by receivers implementing this specification. MAY be defined differently by extensions to this specification.

N (2 bits): SHALL be identical to the highest value of the nal_ref_idc H.264 NAL header syntax element valid for the sub-bitstream described by this OPID, with the exception of nal_ref_idc value 3 that is valid for and is part of all sub-bitstreams.

10.2. H.264 SVC

This document specifies the usage of multiple, simultaneous codec operation points and therefore maps well to scalable video coding. Scalable video coding such as H.264 SVC (Annex G) [H264] uses three scalability dimensions: temporal, spatial, and quality. It also includes the possibility to use redundant encodings and priority among sub-streams.

The ID SHALL be considered describing an SVC sub-bitstream, which is defined in G.3.59 of H.264 [H264] and corresponding RTP payload format [RFC6190]. For use with H.264 SVC, ID SHALL be constructed as
defined below:

<table>
<thead>
<tr>
<th>R</th>
<th>PID</th>
<th>RPC</th>
<th>DID</th>
<th>QID</th>
<th>TID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 17: ID Definition for SVC

R (1 bit): Reserved. SHALL be set to 0 by senders and SHALL be ignored by receivers implementing this specification. MAY be defined differently by extensions to this specification.

PID (6 bits): SHALL be identical to an unsigned binary integer representation of the priority_id H.264 syntax element valid for the sub-bitstream described by this OPID. SHALL be set to 0 if no priority_id is available.

RPC (7 bits): SHALL be identical to an unsigned binary integer representation of the redundant_pic_cnt H.264 syntax element valid for the sub-bitstream described by this OPID. SHALL be set to 0 if no redundant_pic_cnt is available.

DID (3 bits): SHALL be identical to the dependency_id H.264 syntax element valid for the sub-bitstream described by this OPID.

QID (4 bits): SHALL be identical to the quality_id H.264 syntax element valid for the sub-bitstream described by this OPID.

TID (3 bits): SHALL be identical to the temporal_id H.264 syntax element valid for the sub-bitstream described by this OPID.

11. Examples

COP messages are binary encoded. However, in the following examples, all COP messages are for clarity listed in symbolic, pseudo-code form, where only COP message fields of interest to the example are included, along with the COP Parameters.

11.1. SDP Offer/Answer

The SDP capabilities for COP are defined as receiver capabilities, meaning that there is no explicit indication what COP messages an end-point will use in the send direction. It is however reasonable to expect that an end-point can also send the same messages that it can understand and act on when received. This is assumed in all the SDP examples below, but note that symmetric COP capabilities is not a
The example below shows an SDP Offer, where support of CCM "cop" message is announced for the video codecs.

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.atlanta.example
s=-
c=IN IP4 host.atlanta.example
t=0 0
m=audio 50000 RTP/AVP 0 8 97
b=AS:80
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 iLBC/8000
m=video 50010 RTP/AVPF 31 32
b=AS:600
a=rtpmap:31 H261/90000
a=rtpmap:32 MPV/90000
a=rtcp-fb:31 ccm cop framerate bitrate token-rate
a=rtcp-fb:32 ccm cop hor-size ver-size framerate bitrate \
   token-rate
```

Figure 18: SDP Offer (COP support indicated)

Note that the offer contains two different video payload types, and that the COP Parameters differ between them, meaning that the possibility for codec configuration also differ. In this case, the MPEG-1 codec can control both framerate and image size, but for H.261 only the framerate can be controlled.

In the SDP Answer below, responding to the above offer, the answerer supports CCM "cop" messages.

```
v=0
o=bob 2808844564 2808844564 IN IP4 host.biloxi.example
s=-
c=IN IP4 host.biloxi.example
t=0 0
m=audio 52000 RTP/AVP 0
b=AS:80
a=rtpmap:0 PCMU/8000
m=video 52100 RTP/AVPF 32
b=AS:600
a=rtpmap:32 MPV/90000
a=rtcp-fb:32 ccm cop hor-size ver-size framerate bitrate \
   token-rate packet-size
```
Figure 19: SDP Answer (COP support indicated)

Note that the answerer indicates support for more parameter types than the offerer.

Below is another SDP Answer, also responding to the same offer above, where the answerer does not support "cop".

v=0
o=bob 2808844564 2808844564 IN IP4 host.biloxi.example
s=-
c=IN IP4 host.biloxi.example
t=0 0
m=audio 52000 RTP/AVP 0
b=AS:80
a=rtpmap:0 PCMU/8000
m=video 52100 RTP/AVPF 32
b=AS:600
a=rtpmap:32 MPV/90000

Figure 20: SDP Answer (COP support not indicated)

11.2. Dynamic Video Re-sizing

In this example, two COP-enabled end-points communicate in an audio/video session. The receiving end-point has a graphical user interface that can be dynamically changed by the user. This user interaction includes the ability to change the size of the receiving video window, which is also indicated in the previous SDP example (Section 11.1).

At some point during the established communication, a notification about current video stream Codec Operation Point is sent to the re-sizable window end-point that receives the video stream.

```
COPN {SSRC:123456, OPID:123, Version:5,
    bitrate(max):325000,
    token-bucket(exact):1000,
    framerate(exact):15,
    hor-size(exact):320,
    ver-size(exact):240}
```

Figure 21: COPN for QVGA 15 Hz

Some time later the user of the re-sizable window end-point reduces the size of the video window. As a result of the re-size operation, the video window can no longer make full use of the received video resolution, wasting bandwidth and decoder processing resources. The
re-sizable window end-point thus decides to notify the video stream sender about the changed conditions by sending a request for a video stream of smaller size:

COPR {SSRC:123456, OPID:123, Version:5,  
  hor-size(target):243,  
  ver-size(target):185}

Figure 22: COPR for 243x185

The COPR refers to the previously received COPN with the same OPID and Version, and thus need only list parameters that need be changed. The request could arguably contain also other parameters that are potentially affected by the spatial resolution, such as the bitrate, but that can be omitted since the media sender is not slaved to the request but is allowed to make it’s own decisions based on the request.

The request sender has chosen to use target type values instead of an exact value for the horizontal and vertical sizes, which can be interpreted as "anything sufficiently similar is acceptable". The target values is in this example chosen to correspond exactly to the re-sized video display area. Many video coding algorithms operate most efficiently when the image size is some even multiple, and this way of expressing the request explicitly leaves room for the media sender to take such aspect into account.

The media sender (COPR receiver) responds with the following:

COPS {SSRC:123456, OPID:123, Version:5,  
  Partial Success,  
  One or more parameter values in the request were changed}

COPN {SSRC:123456, OPID:123, Version:6,  
  bitrate(max):240000,  
  token-bucket(exact):1000,  
  framerate(exact):15,  
  hor-size(exact):240,  
  ver-size(exact):176}

Figure 23: COPS and COPN for Partial Success

It can be noted that the updated COPN (version 6) indicates that the media sender has, in addition to reducing the video horizontal and vertical size, chosen to also reduce the bitrate. This bitrate reduction was not in the request, but is a reasonable decision taken by the media sender. It can also be seen that the horizontal and vertical sizes are not chosen identical to the request, but is in
fact adjusted to be even multiples of 16, which is a local restriction of the fictitious video encoder in this example. To handle the mismatch of the request and the resulting video stream, the video receiver can perform some local action such as for example automatic re-adjustment of the re-sized window, image scaling (possibly combined with cropping), or padding.

11.3. Illegal Request

In this example, the sent request is asking the media sender to go beyond what is negotiated in the SDP. The SDP Offer below indicates to use video with H.264 Constrained Baseline Profile at level 1.1.

```
v=0
o=alice 2893746526 2893746526 IN IP4 host.atlanta.example
s=-
c=IN IP4 host.atlanta.example
t=0 0
m=audio 49160 RTP/AVP 96
b=AS:80
a=rtpmap:96 G722/16000
m=video 51920 RTP/AVPF 97
b=AS:200
a=rtpmap:97 H264/90000
a=fmtp:97 profile-level-id=42e00b
a=rtcp-fb:97 ccm cop framerate bitrate token-rate
```

Figure 24: SDP Offer With H.264 Level 1.1

Assuming this offer is accepted and that the answerer also supports COP, further assume that this COP message exchange occurs at some time during the established communication:
Media Sender                      Media Receiver
------------                      --------------
COPN {SSRC:9876, OPID:67,      ->
VERSION:2,
bitrate(exact):190000,
token-bucket(exact):500,
framerate(exact):10,
hor-size(exact):320,
ver-size(exact):240}

<-  COPR {SSRC:9876, OPID:67,
VERSION:2,
framerate(exact):10,
hor-size(exact):352,
ver-size(exact):288}

COPS {SSRC:9876, OPID:67,      ->
VERSION:2,
Failure,
Request violates capability limits}

Figure 25: COP Message Exchange Indicating Failure

The failure above is due to a combination of frame size and frame rate that exceeds H.264 level 1.1, which would thus exceed the limits established by SDP Offer/Answer. The maximum permitted framerate for 352x288 pixels (CIF) is 7.6 Hz for H.264 level 1.1, as defined in Annex A of [H264].

11.4. Reference Response to Modification of Scalable Layer

When scalable coding is used, each layer correspond to a Codec Operation Point. A media receiver can thus target a request towards a single layer. Assume a video encoding with three framerate layers, announced in a (multiple operation point) notification as:
COPN \{SSRC:9876, OPID:67, Version:2, ID:2
  bitrate(exact):190000,
  token-bucket(exact):500,
  framerate(exact):10,
  hor-size(exact):320,
  ver-size(exact):240\}

COPN \{SSRC:9876, OPID:73, Version:1,
  bitrate(exact):350000, ID:1
  token-bucket(exact):600,
  framerate(exact):30,
  hor-size(exact):320,
  ver-size(exact):240\}

COPN \{SSRC:9876, OPID:95, Version:5, ID:0
  bitrate(exact):400000,
  token-bucket(exact):800,
  framerate(exact):60,
  hor-size(exact):320,
  ver-size(exact):240\}

Figure 26: COPN Indicating Three Framerate Layers

Assume further that the media receiver is not pleased with the low framerate of OPID 67, wanting to increase it from 10 Hz to 25-30 Hz. Note that the media receiver still wants to receive the other layers unchanged, not remove them, and thus has to explicitly indicate this by including them without parameters.

COPR \{SSRC:9876, OPID:67, Version:2,
  framerate(greater):25,
  framerate(less):30\}

COPR \{SSRC:9876, OPID:73, Version:1\}

COPR \{SSRC:9876, OPID:95, Version:5\}

Figure 27: COPR Requesting to Change One Layer

The media sender decides it cannot meet the request for OPID 67, but instead considers (an unmodified) OPID 73 (with ID 1) to be a sufficiently good match:
COPS {SSRC:9876, OPID:67, Version:2,  
Partial Success,  
One or more parameter values in the request were changed,  
ID:1}

(COPN for the other two OPIDs omitted here for brevity)

COPN {OSSRC:9876, OPID:73, Version:1, ID:1  
bitrate(exact):350000,  
token-bucket(exact):600,  
framerate(exact):30,  
hor-size(exact):320,  
ver-size(exact):240}

Figure 28: COPS and COPN With Layer Modification Partial Success

The COPS indicates partial success and uses the ID number to refer  
another OPID, describing the best compromise that can currently be  
used to meet the request.  COPS does not contain the referred OPID,  
but ID should be defined in a codec-specific way that makes it  
possible to identify the layer directly in the media stream.  If the  
corresponding OPID is needed, for example to attempt another request  
targeting that, it can be found by searching the active set of COPN  
for matching ID values.

11.5. Successful Request to Add Codec Operation Point

In this example, the media receiver is receiving a non-scalable  
stream from a codec that can support scalability, and wishes to add a  
scalloability layer.  Assume the existing OPID from the media sender is  
announced as:

COPN {SSRC:3492, OPID:4, Version:2,  
bitrate(exact):350000,  
token-bucket(exact):600,  
framerate(exact):30,  
hor-size(exact):320,  
ver-size(exact):240}

Figure 29: COPN With Single Operation Point

The media receiver constructs a request for multiple streams by  
including multiple requests for different OPID.  Since the new stream  
does not exist, it has no OPID from the media sender and the receiver  
chooses a random value as reference and indicates that it is a new,  
temporary OPID.  The request for the new stream includes all  
parameters that the media receiver has an opinion on, and leaves the  
other parameters to be chosen by the media sender.  In this case it
is a request for identical frame size and doubled framerate.

COPR {SSRC:3492, OPID:4, Version:2}


Figure 30: COPR Requesting to Add Operation Point

The media sender decides it can start layered encoding with the requested parameters. The status response to the new OPID contains a reference to an ID that is included as part of the matching, subsequent COPN. Note that since both the original and the new streams are now part of a scalable set, they must both be identified with ID parameters to be able to distinguish between them. The media sender has chosen an OPID for the new stream in the COPN, which need not be identical to the temporary one in the request, but the new stream can anyway be uniquely identified through the ID that is announced in both the COPS and COPN.

Note that since the ID has a defined relation to the media sub-stream identification, decoding of that new sub-stream can start immediately after receiving the COPS. It may however not be possible to describe the new stream in COP parameter terms until the COPN is received (depending on COP parameter visibility directly in the media stream).
COPS {SSRC:3492, OPID:4, Version:2, Success, Success, ID:1}

COPS {SSRC:3492, OPID:237, New, Version:0, Success, Success, ID:0}


Figure 31: COPS and COPN Indicating Operation Point Added

12. IANA Considerations

Following the guidelines in [RFC4566], in [RFC4585], and in [RFC3550], the IANA is requested to register:

1. The 'cop' tag to be used with ccm under rtcp-fb AVPF attribute in SDP.

2. The FMT number TBA1 to be allocated to the COP feedback message from this specification.

3. A registry listing registered values for 'cop' Message Item Type, with initial values from Table 1.

4. A registry listing registered values and tag names for 'cop' Parameter Type, with initial values from Table 4.

13. Security Considerations

Editor’s Note: Security considerations must be added.
14. Open Issues

There is currently no defined way for a media receiver to indicate that it wants to release the restrictions it previously had on an Operation Point, if the media stream contains only a single Operation Point.

15. Acknowledgements

The authors would like to thank Prof. Dr.-Ing. Markus Kampmann at Fachhochschule Koblenz University of Applied Sciences and Prof. Dr.-Ing. Frank Hartung at Multimediatechnik, Audio- und Videotechnik at Fachhochschule Aachen for fruitful contributions and discussions during the initial stages of writing this specification. The authors would also like to thank Christer Holmberg for feedback on the specification.

16. References

16.1. Normative References


16.2. Informative References

[I-D.ietf-avtext-multiple-clock-rates]

[I-D.westerlund-avtext-rtp-stream-pause]

[I-D.westerlund-mmusic-sdp-bw-attribute]


Internet-Draft                     COP                        March 2012


Authors’ Addresses

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

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

Bo Burman
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

Phone: +46 10 714 13 11
Email: bo.burman@ericsson.com

Laurits Hamm
Ericsson
Ericsson Allee 1
DE-52134 Herzogenrath
Germany

Phone: +49 2407 575 6779
Email: laurits.hamm@ericsson.com
IEEE 1588/802.1AS Synchronisation for RTP Streams
draft-williams-avtext-avbsync-02

Abstract

This memo specifies an RTP header extension for carrying in-band
synchronization metadata provided by the IEEE1588/802.1AS Precision
Time Protocols.

Requirements Language

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 [1].

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 3, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the
document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
Table of Contents

1. Outline ......................................................... 3
2. Introduction .................................................. 3
3. Reference Clocks and Clock Domains .......................... 4
4. Timestamp formats ............................................. 5
5. IEEE 1733 / AVB RTCP Packet Type .......................... 5
   5.1. Observations ............................................. 7
   5.2. RTCP Packet Subtypes ...................................... 7
6. Timing Header Extension ........................................ 8
7. SDP signalling .................................................. 10
   7.1. Clock domain ............................................. 10
   7.2. Quality of Service ....................................... 11
8. An Alternative Approach ........................................ 11
9. IANA Considerations ........................................... 12
10. Acknowledgements ............................................. 12
11. References .................................................... 12
   11.1. Normative References .................................... 12
   11.2. Informative References .................................. 12
Appendix A. An Appendix .......................................... 13
Author’s Address .................................................. 13
1. Outline

Alternative approach

Use existing header extension for timestamps. (RFC 6051)

Create additional small header extension for metadata (traceability, uncertainty, media clock discontinuity).

IDMS wants accurate timing too and has a method for describing clocks. Create a single specification for signalling clock domains which covers both use cases. IDMS can use that too.

2. Introduction

Synchronisation between RTP flows and between devices rendering RTP flows is currently facilitated by means of NTP format timestamps taken with respect to a shared reference clock. In many applications (e.g. professional, commercial and automotive AV), the NTP clock synchronisation protocol does not meet the necessary time alignment and synchronisation speed requirements.

Like NTP, the IEEE1588 Precision Time Protocol (PTP) family of clock synchronisation protocols provide a shared reference clock in a network - typically a LAN. IEEE1588 provides sub-microsecond synchronisation between devices on a LAN and typically locks within seconds at startup rather than minutes. With support from Ethernet switches, IEEE1588 protocols can achieve nanosecond timing accuracy in LANs. Network interface chips and cards supporting hardware time-stamping of timing critical protocol messages are also available.

When using IEEE1588 clock synchronisation, networked AV systems can achieve sub 1 microsecond time alignment accuracy when rendering AV signals and can support latencies less than 1ms through a gigabit LAN.

Three flavours of IEEE1588 are in use today:

- IEEE 1588-2002 [4]: the original "Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". This is often called IEEE1588v1 or PTPv1.

- IEEE 1588-2008 [5]: the second version of the "Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". This is a revised version of the original IEEE1588-2002 standard and is often called IEEE1588v2 or PTPv2.
IEEE 802.1AS [6]: "Timing and Synchronization for Time Sensitive Applications in Bridged Local Area Networks". This is a Layer-2 only profile of IEEE 1588-2008 for use in Audio/Video Bridged LANs.

By using an IEEE 1588 derived reference clock, synchronisation of RTP streams and devices in LANs can be considerably improved.

3. Reference Clocks and Clock Domains

RTP uses NTP format timestamps to exchange information about media clock timing. NTP timestamps may come from a clock which is synchronised to a global time reference, but this is not assumed nor is there a standardised mechanism available to indicate that timestamps are derived from a common reference clock. Therefore, RTP implementations typically assume that NTP timestamps are taken using unsynchronised clocks and must compensate for absolute time differences and rate differences. Without a shared reference clock, RTP can time align flows from the same source at a given receiver using relative timing, however tight synchronisation between two or more different receivers (possibly with different network paths) or between two or more senders is not possible.

Each IEEE1588 clock is identified by a globally unique EUI-64 called a "ClockIdentity". A slave clock using one of the IEEE1588 family of network time protocols acquires the ClockIdentity/EUI-64 of the grandmaster clock that is the ultimate source of timing information. A master clock which is itself slaved to another master clock passes the grand master clock identity through to its slaves. The grandmaster clock identity can be used to identify a set of clocks in a single clock domain.

The IEEE1588 protocol family also carries an indication that the grand master clock provides traceable time. Traceability of a time source implies that the clock is ultimately slaved to a global time source (e.g. GPS). A slave may consider one or more grandmaster clocks providing traceable time as belonging to the same (global) clock domain. Informally, traceability trumps ClockIdentity when comparing clock domains. Traceable time is particularly applicable for wide area applications such as broadcast, transport and reflection seismology.

Several instances of the IEEE1588v1/v2 protocol may operate independently on a single network, forming distinct PTP network protocol domains each of which may have a different master clock. As the IEEE1588 standards have developed, the definition of PTP domains has changed. IEEE1588v1 identifies protocol subdomains by a textual
name and IEEE1588v2 identifies protocol domains using a numeric domain number. 802.1AS is a Layer2 profile of IEEE1588v2 supporting a single numeric clock domain (0). This specification assumes that an IEEE1588 clock master for multiple domains will provide the same timing information to all domains or that each clock domain has a different master. In other words, this specification assumes that a timing domain can be uniquely identified using the ClockIdentity of the grandmaster clock alone.

Accurate signal playout time alignment and accurate capture of input signal phase relationships are important requirements for A/V systems. In distributed AV systems containing many senders/receivers and differing network path lengths, a shared reference clock providing absolute time is needed. Relative timing using unsynchronized clocks cannot provide the required level of time alignment (+/- 1us).

4. Timestamp formats

IEEE 1588/802.1AS timestamps use International Atomic Time (TAI) rather than UTC. Timestamps are 80 bits in total, divided into two parts:

PTP_sec: 48 bits seconds since epoch

PTP_nsec: 32 bits nanoseconds

A shorter 32 bit timestamp has also been defined for use in streaming media protocols in the following way:

as_timestamp = (PTP_sec * 10^9 + PTP_nsec) modulo 2^32

The shorter as_timestamp field covers a time interval slightly larger than 4 seconds.

5. IEEE 1733 / AVB RTCP Packet Type

IEEE 1733 [7] defines the "AVB RTCP packet" type reproduced in Figure 1. RTCP AVB packets contain a mapping between RTP timestamp and an 802.1AS timestamp as well as additional clock and QoS information.

The RTCP packet type shown below provides the corresponding IEEE1588 timestamp for an RTP timestamp in a similar fashion to an RTCP Sender Report (SR). In addition, the source of the timing information (i.e. the grandmaster ClockIdentity) is included. Clock domain information
allows an RTP implementation to determine whether sender and receiver
timestamps have been taken using a shared reference clock. If the
sender and receiver share a common clock domain, timestamps and be
used and compared in an absolute sense.

Figure 1: IEEE 1733 / AVB RTCP packet format

A brief description of the major fields follows:

name Reserved, must be set to zero and ignored on reception.

gmTimeBaseIndicator This field identifies the clock master time
base. The gmTimeBaseIndicator increments each time a step change
in time or frequency occurs. If the value of the
gmTimeBaseIndicator and gmidentity in the RTCP packet (i.e., of
the sender) match the gmTimeBaseIndicator and gmidentity at the
receiver, the receiver is assured that timestamps have been taken
using a shared reference clock. If they do not match, actions
performed by the receiver are application dependent but may
include entering a holdover mode until a positive match is again
achieved.
gmPortNumber  An integer identifying the port used on a grand master.

gmClockIdentity  An EUI-64 identifying the grand master clock used by
the source to generate as_timestamps for this flow.

stream_id  A 64 bit number identifying the 802.1Qat [8] QoS
reservation associated with this RTP flow.

as_timestamp  The 32 bit 802.1AS timestamp (Section 4) associated
with the RTP timestamp carried in this packet.

rtp_timestamp  The RTP timestamp of a media packet.


5.1. Observations

The RTCP packet type above provides the basic information for linking
IEEE1588 time to RTP timestamps, however there are some additions
which would improve the functionality provided.

The IEEE1733 specification does not define any SDP signalling. This
means that the QoS parameters (ie the stream_id) cannot be signalled
at call/flow setup time using SIP/RTSP. Whilst IEEE1733 provides
clock domain information, it doesn’t really define how clock domains
work. Further, IEEE1733 does not include metadata to indicate clock
changes. This specification addresses each of these issues.

5.2. RTCP Packet Subtypes

Systems not using the full AVB protocol suite can still benefit from
timing improvements offered by IEEE 1588v1 and IEEE 1588v2. In
general, the IEEE 1733 / AVB RTCP packet format (Figure 1) can be
used to relate RTP timestamp instants in media packets to a reference
clock provided by any of the IEEE 1588/PTP family of clock
synchronisation protocols.

The subtype field indicates which IEEE 1588 protocol was used by the
timestamping clock:

0x00  IEEE 802.1AS (defined by IEEE 1733)

0x01  IEEE 1588v1

0x02  IEEE 1588v2
6. Timing Header Extension

The header extension shown below provides a combination of media timestamps and timing metadata.

The provision of IEEE1588 timestamps in the RTP header is analogous to the existing NTP timestamp header extension supporting rapid synchronisation of RTP flows. High performance systems often handle RTP media packets in hardware and carriage of timing metadata in the media packets provides increased performance (e.g. faster lock times) and simplifies the system. In contrast to using RTCP, a header extension guarantees that timing metadata arrives at the receiver with the first media packet allowing them to be processed immediately. This approach facilitates fast switching between sources as is commonly used in zoned paging applications.

If the sender and receiver share a clock domain, source timestamps in media packets facilitate the collection of high quality information about the actual delays experienced by media packets through the network. Accurate delay information is important for diagnosing operational problems and for system optimisation.

The metadata included in the header allows senders to signal changes in clock disposition to receivers. Since IEEE1588 uses an election mechanism to determine the clock master it is possible for the master and/or grand master to change during the transmission of an RTP flow. Changing from one master clock to another may involve changes in clock frequency and/or phase. Additionally, the election protocol is not guaranteed to complete at the same time at all nodes in the clock domain, so there will be a transition period during which timestamps at the sender and receiver are not directly comparable. A sender indicates changes in IEEE1588 clocking by setting the timestamp uncertain (U) bit in the header. For example, the bit may be set when the best master clock election process is triggered due to loss of the current master clock. Once set, this bit should remain set during the entire period where PTP timing is in flux and for at least 5 seconds after PTP timing has stabilised (5 seconds allows as_timestamps which cover about 4 seconds to be disambiguated). When PTP timestamps are uncertain the receiver may refrain from updating the rtp_timestamp to wallclock mapping, effectively processing packets on the basis of rtp_timestamp alone.

In AV systems, media clocks are often provided via an external connection. When a media clock is disconnected or switched from one source to another (e.g. between two different SP/DIF ports), synchronisation may be lost and noise may be generated. A sender can indicate the loss of a media clock for a transmitted signal by toggling the media clock restart (M) bit. A receiver may use a
change in the media clock restart bit to mute audio or to hold a
frame or to trigger resynchronisation to the new media clock.

In some systems, senders and receivers are on different networks with
different grand master clocks however they may be part of a single
global clock domain that uses traceable time. A sender can indicate
that media packet timestamps have been taken with a traceable clock
by setting T=1. Since it is possible for a PTP clock master to
change during an RTP flow, the T bit may change. For example, if the
traceable grand master clock used by the sender fails, PTP will elect
a new grand master which may not be slaved to a global time source.
In systems where traceable time is important, all candidate PTP
masters should support traceable time and media signals timestamps
should be clearly marked as traceable or not traceable. If
timestamps are not traceable, T MUST be set to zero.

Figure 2 shows the fields of the AVB sync header extension. It uses
the standard RTP header extension mechanism defined in RFC 5285 [2].

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>V=2</td>
<td>P</td>
<td>1</td>
<td>CC</td>
</tr>
<tr>
<td>--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-------------------------------------------R</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>timestamp</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>+-------------------------------------------P</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>synchronisation source (SSRC) identifier</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0xBE</td>
<td>0xDE</td>
<td>length=2</td>
</tr>
<tr>
<td>+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID=7</td>
<td>L=6</td>
<td>subtype</td>
</tr>
<tr>
<td>+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>as_timestamp</td>
<td>t</td>
<td></td>
</tr>
<tr>
<td>+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>payload data</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-------------------------------+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Figure 2: Timestamp Header Extension</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The fields are defined as follows:

T  Bit field indicating traceable timestamps. If T=1, the timestamp
in this packet was taken by a clock slaved to a global time source
otherwise T MUST be set to zero.
M  Toggling the value of the media clock restart field (M) indicates
a change in the source of the media clock. This kind of change
need not be seamless but allows the receiver to take any
appropriate action to minimize the disruption to the stream. The
value of the M bit is toggled once by the sender each time the
media clock source is changed. Toggling the bit rather than
setting/resetting ensures media clock restarts will not be missed
if it were to appear in a single RTP packet which happens to be
dropped.

U  The sender sets U=1 to indicate that timestamps may not be
globally synchronized with network time. An example is the
invocation of the best master clock algorithm because of a PTP
SYNC timeout. Receivers may use the U bit, possibly in
conjunction with their knowledge of the status of the PTP clock,
to prevent unacceptable disturbances in the recovered media
streams.

subtype  See Section 5.2.

as_timestamp  A 32 bit IEEE 1588/802.1AS timestamp as defined in
Section 4.

reserved  As this specification evolves, additional fields may be
included in this header.

The as_timestamp MUST correspond to the same instant as the RTP
timestamp in the packet’s header, and MUST be derived from the same
clock used to generate the as_timestamps in the RTCP AVB packets.
Provided that it has knowledge of the SSRC to CNAME mapping, either
from prior receipt of an RTCP CNAME packet or via out of band
signalling such as RFC 5576 [3], the receiver can use the information
provided as input to the synchronization algorithm, in exactly the
same way as if an additional RTCP AVB packet had been received for
the flow.

7.  SDP signalling

The Session Description Protocol (SDP) allows clock domain and QoS
information to be signalled via SIP or RTSP at call setup time. In
the case of SIP, changes in information can also be signalled during
the RTP session.

7.1.  Clock domain

General format for signalling clock domains (todo: convert to EBNF):

\[
\text{clock_domain} = \text{as_timestamp} | \text{reserved} | \text{subtype} | \text{reserved}
\]
Examples:

a=clockdomain:ptp-version=IEEE1588v1 gmid=39-A7-94-FF-FE-07-CB-D0 traceable=yes

a=clockdomain:ptp-version=IEEE1588v2 gmid=39-A7-94-FF-FE-07-CB-D0 traceable=yes

a=clockdomain:ptp-version=802.1AS gmid=39-A7-94-FF-FE-07-CB-D0 traceable=no

7.2. Quality of Service

General format for signalling AVB QoS to a receiver:

a=8021qat-qos:stream-id=EUI-64

Example:

a=8021qat-qos:stream-id=00-1D-C1-97-BB-3A-01-01

8. An Alternative Approach

Some RTP specifications already exist which cover aspects of the functionality described in this specification. Most notably, there is a specification for a header extension allowing NTP format timestamps to be included in RTP media packet headers (RFC 6051).

In addition, some new work (http://tools.ietf.org/html/draft-ietf-avtcore-idms-02) specifies a mechanism for describing clock sources including those based on IEEE 1588.

Rather developing new specifications based on the 32 bit as_timestamp format, an alternative approach which could fulfil the goals of this specification could be to:

1. Use the existing RFC 6051 to carry timestamps in RTP media packets (nothing to be done).

2. Create a new specification describing clock sources/domains and their signalling that is independent of both this specification and the IDMS specification. This would allow clock domains to be signalled generally in RTP. This might be done in avtcore.

3. Create a new specification augmenting RFC 6051 which carries the timing metadata (ptp uncertainty, media clock change,
traceability).

4. This specification: the header extension and clock domain signally go into other documents. There may be some remaining text describing how AVB systems in particular would use the above mechanisms.

9. IANA Considerations

TBD: A URN will be required to signal the presence of this header extension, such as:

urn:ietf:params:rtp-hdrext:avb-sync

10. Acknowledgements

The timing metadata (U and M) bits were inspired by the IEEE 1722 specification.

11. References

11.1. Normative References


11.2. Informative References


Appendix A. An Appendix

Author’s Address

Aidan Williams
Audinate
Level 1, 458 Wattle St
Ultimo, NSW 2007
Australia

Phone: +61 2 8090 1000
Fax: +61 2 8090 1001
Email: aidan.williams@audinate.com