

avtcore
Internet-Draft
Intended status: Standards Track
Expires: 30 August 2026

Y. Lim
M. Park
M. Budagavi
R. Joshi
K. Choi
Samsung Electronics
26 February 2026

RTP payload format for APV
draft-ietf-avtcore-rtp-apv-01

Abstract

This document describes RTP payload format for bitstream encoded with Advanced Professional Video (APV) codec.

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 <https://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 30 August 2026.

Copyright Notice

Copyright (c) 2026 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 (<https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Terms	3
2.1. Terms and definitions	3
3. Conventions used in this document	6
3.1. General	6
4. Overview of APV	6
4.1. Overview of APV coding tools	6
4.2. Structure of APV bitstream	6
4.2.1. Structure of APV access unit	7
4.2.2. Structure of APV primitive bitstream unit	7
4.2.3. Structure of APV PBU for coded frame	8
4.2.4. Structure of APV PBU for access unit information	9
4.2.5. Structure of APV PBU for metadata	9
4.2.6. Structure of APV PBU for filler	9
5. Packetization rules	9
5.1. General rules	10
5.2. Simple mode	10
5.3. Low delay mode	10
5.4. RTP Header Usage	11
5.5. Payload Header Usage	12
6. Payload Format Parameters	14
6.1. Media Type Registration	14
6.1.1. Definition of optional parameters	15
6.2. SDP Parameters	16
6.2.1. Mapping of Payload Type Parameters to SDP	16
7. Congestion Control	17
8. Security Considerations	17
9. IANA Considerations	18
10. References	18
10.1. Normative References	18
10.2. Informative References	19
Authors' Addresses	19

1. Introduction

This document defines the RTP payload format for bitstream encoded with Advanced Professional Video (APV) codec [RFC9924]. This document defines how to packetize bitstream encoded with APV codec and set the payload header fields. This document also defines how to set the fields of RTP payload header when it carries bitstream encoded with APV codec. The APV codec is a professional video codec that was developed in response to the need for high quality video recording and post production.

The primary purpose of the APV codec is for use in professional video recording and editing workflows for various types of content. The APV codec supports the following features:

- * Perceptually lossless video quality that is close to the original, uncompressed quality;
- * Low complexity and high throughput intra frame only coding without inter frame coding;
- * Intra frame coding without prediction between pixel values but with prediction between transformed values for low delay encoding;
- * High bit rates of up to a few Gbps for 2K, 4K, and 8K resolution content, enabled by a lightweight entropy coding scheme;
- * Frame tiling for immersive content and for enabling parallel encoding and decoding;
- * Various chroma sampling formats from 4:0:0 to 4:4:4:4, and bit depths from 10 to 16 (Note: Only the profiles supporting 10 bits and 12 bits are currently defined);
- * The ability to decode and re-encode multiple times without severe visual quality degradation; and
- * Various metadata including HDR10/10+ and user-defined formats.

2. Terms

2.1. Terms and definitions

access unit (AU) a collection of primitive bitstream units (PBU) including various types of frames, metadata, filler, and access unit information, associated with a specific time

band a defined set of constraints on the value of the maximum coded data rate of each level

block MxN (M-column by N-row) array of samples, or an MxN array of transform coefficients

byte-aligned a position in a bitstream that is an integer multiple of 8 bits from the position of the first bit in the bitstream

chroma a sample array or single sample representing one of the two

color difference signals related to the primary colors, represented by the symbols Cb and Cr in 4:2:2 or 4:4:4 color format

coded frame a coded representation of a frame containing all macroblocks of the frame

coded representation a data element as represented in its coded form

component an array or a single sample from one of the three arrays (luma and two chroma) that compose a frame in 4:2:2, or 4:4:4 color format, or an array or a single sample from an array that compose a frame in 4:0:0 color format, or an array or a single sample from one of the four arrays that compose a frame in 4:4:4:4 color format.

decoded frame a frame derived by decoding a coded frame

decoder an embodiment of a decoding process

decoding process a process specified that reads a bitstream and derives decoded frames from it

encoder an embodiment of an encoding process

encoding process a process that produces a bitstream conforming to this document

flag a variable or single-bit syntax element that can take one of the two possible values: 0 and 1

frame: an array of luma samples and two corresponding arrays of chroma samples in 4:2:2 and 4:4:4 color format, or an array of samples in 4:0:0 color format, or four arrays of samples in 4:4:4:4 color format

level a defined set of constraints on the values that are taken by the syntax elements and variables of this document, or the value of a transform coefficient prior to scaling

luma a sample array or single sample representing the monochrome signal related to the primary colors, represented by the symbol or subscript Y or L

macroblock (MB) a square block of luma samples and two corresponding

blocks of chroma samples of a frame in 4:2:2 or 4:4:4 color format, or a square block of samples of a frame in 4:0:0 color format, or four square blocks of samples of a frame in 4:4:4:4 color format

metadata data describing various characteristics related to a bitstream without directly affecting the decoding process of it.

partitioning a division of a set into subsets such that each element of the set is in exactly one of the subsets

prediction an embodiment of the prediction process

prediction process use of a predictor to provide an estimate of the data element currently being decoded

predictor a combination of specified values or previously decoded data elements used in the decoding process of subsequent data elements

primitive bitstream unit (PBU) a data structure to construct an access unit with frame and metadata

profile a specified subset of the syntax of this document

quantization parameter (QP) a variable used by the decoding process for the scaling value of transform coefficients

raster scan a mapping of a rectangular two-dimensional pattern to a one-dimensional pattern such that the first entries in the one-dimensional pattern are from the top row of the two-dimensional pattern scanned from left to right, followed by the second, third, etc., rows of the pattern each scanned from left to right

raw bitstream an encapsulation of a sequence of access units where a field indicating the size of an access unit precedes each access unit as defined in Appendix A. of [RFC9924]

source a term used to describe the video material or some of its attributes before the encoding process

syntax element an element of data represented in the bitstream

syntax structure zero or more syntax elements present together in a bitstream in a specified order

tile a rectangular region of MBs within a particular tile column and a particular tile row in a frame

tile column a rectangular region of MBs having a height equal to the height of the frame and width specified by syntax elements in the frame header

tile row a rectangular region of MBs having a height specified by syntax elements in the frame header and a width equal to the width of the frame

tile scan a specific sequential ordering of MBs partitioning a frame in which the MBs are ordered consecutively in MB raster scan in a tile and the tiles in a frame are ordered consecutively in a raster scan of the tiles of the frame

transform coefficient a scalar quantity, considered to be in a frequency domain, that is associated with a particular one-dimensional or two-dimensional index

3. Conventions used in this document

3.1. General

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

4. Overview of APV

4.1. Overview of APV coding tools

The APV codec encodes each frame individually from other frames so that there are no coding dependencies among the frames. A frame is divided into one or more rectangular tiles. Each tile is also encoded independently from other tiles so that parallel processing of tiles is possible. A tile is further divided into 16 x 16 pixel size macroblocks which include 4 transform blocks of 8 pixel x 8 pixel. Each transform block is transformed using a fixed-point DCT and then transformed coefficients are quantized using uniform scalar quantizer. A prediction is applied to the quantized coefficients in the frequency domain. Finally, entropy coding specially designed to support very high throughput is applied.

4.2. Structure of APV bitstream

4.2.1. Structure of APV access unit

As the APV codec encodes each video frame independently from other video frames, the coded data of each individual video frame, access unit, is self-contained. As there are cases that there are more than one video frames corresponding to a specific time, the access unit is designed to carry multiple video frames and metadata associated to such video frames corresponding to a single time in a single self-contained unit. The access unit consists of one or more primitive bitstream units as shown in Figure 1. Each PBU carries a single video frame, metadata or filler data. It is assumed that the size of AU is known by external means. The detailed syntax and semantics of the access unit is defined in section 5.3.1 of [RFC9924].

```

+-----+
| primitive bitstream unit | ... | primitive bitstream unit |
+-----+

```

Figure 1: A conceptual structure of APV access unit

4.2.2. Structure of APV primitive bitstream unit

The primitive bitstream unit is designed to carry various type of data consisting a single access unit in a consistent structure. The PBU is composed of PBU size, PBU header and PBU data as shown in Figure 2.

```

+-----+
| PBU size | PBU header | PBU data |
+-----+

```

Figure 2: A conceptual structure of primitive bitstream unit

For easy identification of specific type of data and fast parsing of large size AU, the PBU start with the size information and the first byte of the PBU header provides a type of data carried in a PBU. The list of types of data to be carried in a PBU is listed in Table 1. The detailed syntax and semantics of PBU is specified in section 5.3.2 and 5.3.3 of [RFC9924].

pbu_type	meaning	notes
0	reserved	
1	primary frame	
2	non-primary frame	
3...24	reserved	
25	preview frame	
26	depth frame	
27	alpha frame	
28...64	reserved	
65	access unit information	
66	metadata	
67	filler	
68...255	reserved	

Table 1: List of PBU types

4.2.3. Structure of APV PBU for coded frame

The primitive bitstream unit which carries encoded video frames such as primary frame, non-primary frame, preview frame, depth frame or alpha frame, PBU carries a coded frame as shown in Figure 3. To support independent decoding of each frame, each coded frame starts with frame header. Frame header provides basic information for decoder configuration and bitstream processing. It includes profile, level, and band of the required decoder, width and height of frame, chroma format and bit depths of pixel data and so on. It also provide distance of capture time between the previously encoded frame and the current frame to indirectly indicate frame rates of encoded video. It can optionally contain color description or quantization matrix.

As a video frame can be divided into multiple tiles there can be one or more pair of tile size and tile data in a frame. The information about the structure of tiles in a frame is carried in the frame header for random access of a certain tile and parallel decoding of tiles.

A coded frame can optionally have filler at the end. The detailed syntax and semantics of frame and frame header are specified in section 5.3.4 and 5.3.5 of [RFC9924], respectively.

```
+-----+
|PBU size|PBU header|frame header|tile size|tile data|...| filler|
+-----+
```

Figure 3: A conceptual structure of APV PBU containing an encoded video frame

4.2.4. Structure of APV PBU for access unit information

When PBU carries access unit information, PBU data carries the data specified in 5.3.9 of [RFC9924]. Access unit information provides number of frames contained in access unit and the types of such frames and information to understand capability of the required decoder such as profile, level, band, the width and height of frame and so on without parsing frame header of all frames. Access unit information can optionally include filler at the end.

4.2.5. Structure of APV PBU for metadata

When PBU carries metadata, PBU data carries metadata specified in 5.3.10 of [RFC9924]. Metadata starts with its size and can include more than one type of metadata where each consists of type, length and value of data. The list of type of metadata is defined in the section 8.2 of [RFC9924].

4.2.6. Structure of APV PBU for filler

When PBU carries filler, PBU data carries filler specified in 5.3.11 of [RFC9924]. Filler can be used to make empty space with a certain size within an access unit to make the start position each frames aligned to be a multiple of certain size for fast parsing or to avoid rewriting of entire access unit during editing.

5. Packetization rules

5.1. General rules

When APV bitstream is carried by RTP packets, the size of AU is added before the first PBU of each AUs as specified in section Appendix A. of [RFC9924]. The length of the AU size field, `au_size`, is 32 bits. Then the first byte of the field indicating the size of AU must be the first byte of a payload of an RTP packets when it is carried so that the start of an APV access unit can be always aligned with the start of the payload of an RTP packet. There MUST be no RTP packet carrying data from two different APV AUs.

5.2. Simple mode

In this mode an APV AU can be fragmented anywhere. The payload of RTP packets does not have to be aligned with beginning or end of any particular internal data structure of an APV bitstream. An example of this mode is shown in Figure 4

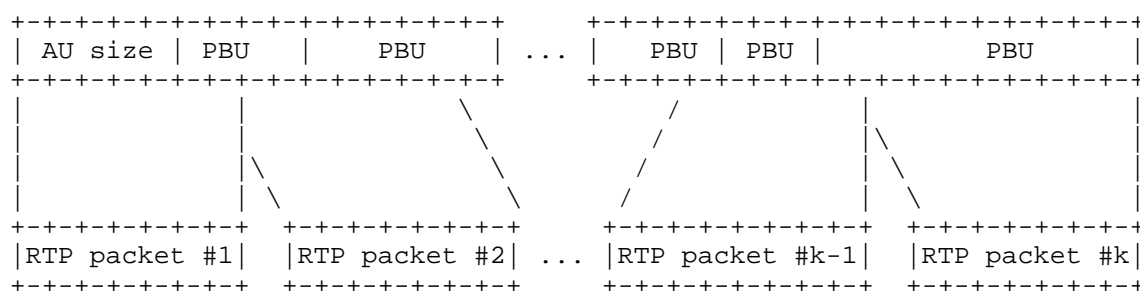


Figure 4: Example of simple mode

5.3. Low delay mode

In this mode both the first byte of each PBU except the one immediately following the field indicating the size of AU and the field indicating the size of each tile, `tile_size` specified in section 5.3.4 of [RFC9924], MUST be aligned with the beginning of RTP packet payloads. The first byte of the `tile_size_minus1` field MUST be the first byte of a RTP packet payload after payload header except the first one following `frame_header`. Metadata and filler data can be added to the payload after the last tile data of a coded frame. An example of this mode is shown in Figure 5

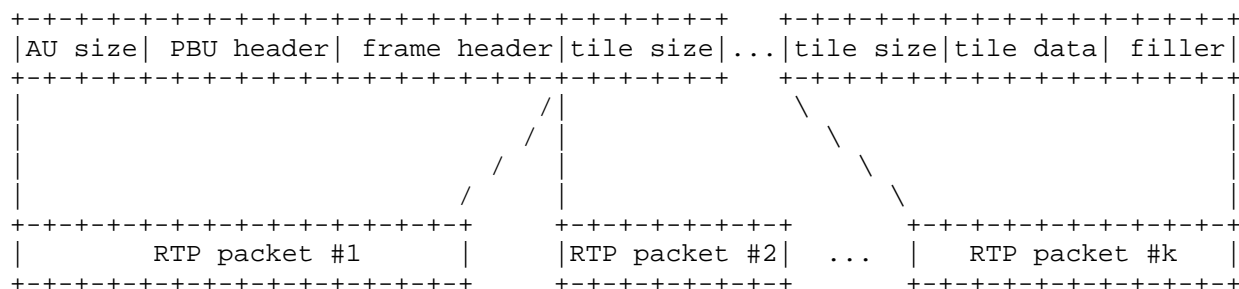


Figure 5: Example of low delay mode

In this mode, frame header can be repeated in any packet containing the last part of a frame or a tile. When frame header is repeated it MUST be placed immediately after the end of tile data or filler data, if exist.

5.4. RTP Header Usage

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

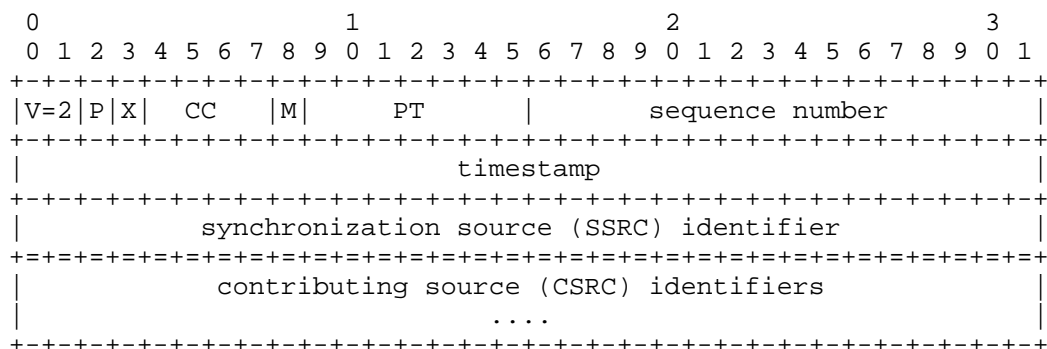


Figure 6: RTP Header According to [RFC3550]

The RTP header information to be set according to this RTP payload format as follows and the usage of the fields not specified in this section follows the rules defined in [RFC3550] :

Marker bit (M): 1 bit

set to 1 for the first packet of each APV AU, i.e. the packet containing the first byte of the field indicating the size of an APV AU.

Timestamp: 32 bits

The RTP timestamp is set to the sampling timestamp of an APV AU. A 90 kHz clock rate **MUST** be used. The RTP packets containing the data belong to a single APV AU **MUST** have same value for this field.

5.5. Payload Header Usage

Each packet carries APV encoded bitstream **MUST** have a payload header as shown in Figure 7.

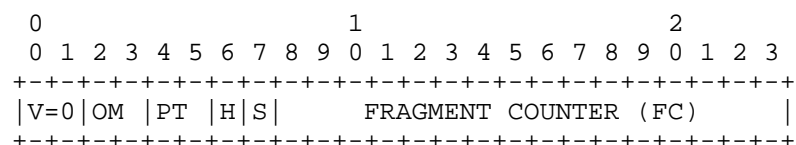


Figure 7: Payload Header

***Version (V)* : 2 bits**

This field indicates the version of the payload header. The version of the header shown in Figure 7 **MUST** have '0' as the value of this field.

***Operation Mode (OM)* : 2 bits**

This field indicates which operation mode is used for packetization of the bitstream.

- 00b : reserved
- 01b : simple mode as defined in Section 5.2
- 10b : low-delay mode as defined in Section 5.3
- 11b : reserved

***Payload Type (PT)* : 2 bits**

This field indicates the type of payload. Depending on the packetization mode the semantics of this field is slightly different. When a single packet carries entire frame in simple mode or tile in low delay mode then this field **MUST** be set to 01b.

For simple mode (OM == '01b')

- + 00b: neither the first payload nor the last payload

- + 01b: the last payload of an APV AU

- + 10b: the first payload of an APV AU

- + 11b: reserved

For low delay mode (OM == '10b')

- + 00b: a payload containing the first byte of neither an APV PBU nor the first byte of a field indicating the size of tile

- + 01b: a payload containing the first byte of an APV PBU

- + 10b: a payload containing the first byte of a field indicating the size of tile

- + 11b: reserved

Frame Header repeated (H) : 1 bit

This field indicates that the frame header specified in section 5.3.5 of [RFC9924] is repeated in this payload. When the value of this field is equal to '1', the payload carries frame header. The value of this field MUST NOT be set to '1' when a payload carries a frame header at the beginning of a coded frame. The value of this field MUST be set to 1 when the value of OM field is equal to 10b and the value of PT field is either 01b or 10b and the payload carries a copy of the frame header already sent. When the value of the OM field is equal to 10b and the value of this field is equal to '1', the payload includes a copy of frame header data after the end of a tile data. If the payload carries the data from the last tile of a frame and there are filler then the copy of a frame header is carried after it. When the value of the OM field is equal to 01b then the value of this field is ignored.

Static Frame Header (S) : 1 bit

This field indicates the values of frame header are identical except the value of `capture_time_distance` field with the immediately preceding coded frame sent. When the value of this field is equal to '1' for the RTP packet carrying the first byte of AU, it means that the values of frame header are identical except the value of `capture_time_distance` field with the last frame header of the AU immediately preceding this AU.

Fragment Counter (FC) : 16 bit

This field indicates the number of remaining payloads, excluding the current one carrying the current APV AU or tile data, depending on the operation mode. When the value of the Operation Mode field is '01b', then the value of this field indicates the number of payloads carrying the bitstream from a same APV AU. When the value of the Operation Mode field is '10b', then the value of this field indicates the number of payloads carrying the bitstream from a same tile data. When there is a filler immediately after a tile data, such filler is considered as an integral part of the tile data. When there are APV PBUs carrying AU info, metadata or filler, they are considered as an integral part of an APV AU but tile data. When the value of the H field is equal to '1', the frame header repeated after the end of a tile data is considered as an integral part of the tile data. The value of this field carrying the last byte of an APV AU or tile data depending on the value of the OM field will be '0'.

6. Payload Format Parameters

6.1. Media Type Registration

The receiver MUST ignore any parameter unspecified in this document.

Type name: video

Subtype name: apv

Required parameters: N/A

Optional parameters: profile-id, level-id, band-id

Encoding considerations:

This type is only defined for transfer via RTP (RFC 3550).

Security considerations:

See Section 8 of RFC 9924.

Interoperability considerations: N/A

Published specification:

Please refer RFC on APV [RFC9924].

Applications that use this media type:

Any application that relies on APV encoded video delivery over RTP

Fragment identifier considerations: N/A

Additional information: N/A

Person & email address to contact for further information:

Youngkwon Lim (yklwhite@gmail.com)

Intended usage: COMMON

Restrictions on usage: N/A

Author: See Authors' Addresses section of RFC XXXX.

Change controller:

IETF <avtcore@ietf.org>

6.1.1.1. Definition of optional parameters

profile-id:

When profile-id is not present, a value of 33 (i.e., the Baseline profile) MUST be inferred.

When used to indicate properties of a bitstream, profile-id MUST be derived from the profile_idc field in the frame header. When there is more than one value of profile_idc field is found from frame headers then the largest value among them MUST be used.

APV encoded data transported over RTP using the technologies of this document SHOULD refer only to frame header that have the same or smaller value in profile_idc.

level-id:

When level-id is not present, a value of 153 (corresponding to level 5.1, the highest level) MUST be inferred.

When used to indicate properties of a bitstream, level-id MUST be derived from the level_idc field in the frame header. When there are more than one value of profile_idc field is found from frame headers then the largest value among them MUST be used.

For either receiving or sending, all levels that are lower than the indicated level MUST also be supported.

band-id:

When band-id is not present, a value of 0 MUST be inferred.

When used to indicate properties of a bitstream, band-id MUST be derived from the band_idc in the frame header. When there are more than one value of band_idc field is found from frame headers then the largest value among them MUST be used.

For either receiving or sending, all band that are lower than the indicated band MUST also be supported.

6.2. SDP Parameters

The receiver MUST ignore any parameter unspecified in this document.

6.2.1. Mapping of Payload Type Parameters to SDP

When Session Description Protocol (SDP) [RFC8866] is used to describe the sessions using this payload format the mapping is done 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 apv (the media subtype).
- * The clock rate in the "a=rtpmap" line MUST be 90000.
- * The optional parameters profile-id and level-id, when present, MUST be included in the "a=fmtp" line of SDP. The fmtp line is expressed as a media type string, in the form of a semicolon-separated list of parameter=value pairs.

As main application area of APV is high quality video capturing and editing, it is expected that generally one way APV session is offered over RTP using SDP in a declarative style. All parameters are used to indicate only bitstream properties. For example, in this case, the parameters profile-id and level-id declare the values used by the bitstream, not the capabilities for receiving bitstreams. An example of media representation in SDP for such case is as follows:

```
m=video 49170 RTP/AVP 98
a=rtpmap:98 apv/90000
a=fmtp:98 profile-id=30; level-id=153; band-id=0;
```

The above represents a stream of data using [RFC9924] and its payload specification at the baseline profile and level 5.1.

It is not expected that [RFC9924] is offered over RTP using SDP in and Offer/Answer model with negotiation.

7. Congestion Control

Congestion control for RTP SHALL be used in accordance with RTP [RFC3550] and with any applicable RTP profile, e.g., AVP [RFC3551]. If best-effort service is being used, an additional requirement is that users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within an acceptable range. Packet loss is considered acceptable if a TCP flow across the same network path and experiencing the same network conditions would achieve an average throughput, measured on a reasonable timescale, that is not less than all RTP streams combined are achieved. This condition can be satisfied by implementing congestion-control mechanisms to adapt the transmission rate, by implementing the number of layers subscribed for a layered multicast session, or by arranging for a receiver to leave the session if the loss rate is unacceptably high.

The bitrate adaptation necessary for obeying the congestion control principle is easily achievable when real-time encoding is used, for example, by adequately tuning the quantization parameter. However, when pre-encoded content is being transmitted, bandwidth adaptation requires the pre-coded bitstream to be tailored for such adaptivity. Regardless of the method used for bandwidth adaptation, the resulting bitstream MUST be compliant with [RFC9924].

8. Security Considerations

The scope of this section is limited to the payload format itself and to one feature of [RFC9924] that may pose a particularly serious security risk if implemented naively. Implementers are advised to read and understand relevant security-related documents, especially those pertaining to RTP (see the Security Considerations section in [RFC3550]). Implementers should also consider known security vulnerabilities of video coding and decoding implementations in general and avoid those.

Within this RTP payload format no security threats other than those common to RTP payload formats are known. In other words, neither the various media-plane-based mechanisms nor the signaling part of this document seem to pose a security risk beyond those common to all RTP-based systems.

Because the data compression used with this payload format is applied end-to-end, any encryption needs to be performed after compression. A potential denial-of-service threat exists for data encodings using compression techniques that have non-uniform receiver-end

computational load. The attacker can inject pathological datagrams into the bitstream that are complex to decode and that cause the receiver to be overloaded.

APV data can include user-data as a part of metadata. [RFC9924] does not specify how to process such data. Depending on the user-data, it might be possible for a sender to generate user-data in a manner to crash the receiver. Receivers must ensure that it knows the format of user-data and trust the sender before it processes user-data. In any case, processing of user-data is not required for decoding of APV data. So, receivers do not have to try to process unknown user-data.

9. IANA Considerations

A new media type, as specified in Section 6.1 of this document, is to be registered with IANA.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/rfc/rfc2119>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<https://www.rfc-editor.org/rfc/rfc3550>>.
- [RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <<https://www.rfc-editor.org/rfc/rfc3551>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/rfc/rfc8174>>.
- [RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <<https://www.rfc-editor.org/rfc/rfc8866>>.

[RFC9924] Lim, Y., Park, M., Budagavi, M., Joshi, R., and K. Choi,
"Advanced Professional Video", RFC 9924,
DOI 10.17487/RFC9924, February 2026,
<<https://www.rfc-editor.org/rfc/rfc9924>>.

10.2. Informative References

[AOSP16APV] "Android open source project version 16",
<<https://developer.android.com/about/versions/16/features#apv>>.

[FFmpegAPVdec] "FFmpeg implementation of APV decoder", 20 November 2025,
<https://ffmpeg.org/download.html#release_8.0>.

[FFmpegAPVenc] "FFmpeg implementation of APV encoder", 4 May 2025,
<<https://git.ffmpeg.org/gitweb/ffmpeg.git/commit/fab69ledaf53bbf10429ef3448f1f274e5078395>>.

[OpenAPV] "OpenAPV",
<<https://github.com/AcademySoftwareFoundation/openapv>>.

Authors' Addresses

Youngkwon Lim
Samsung Electronics
6105 Tennyson Pkwy, Ste 300
Plano, TX, 75024
United States of America
Email: yklwhite@gmail.com

Minwoo Park
Samsung Electronics
34, Seongchon-gil, Seocho-gu
Seoul
3573
Republic of Korea
Email: m.w.park@samsung.com

Madhukar Budagavi
Samsung Electronics
6105 Tennyson Pkwy, Ste 300
Plano, TX, 75024
United States of America

Email: m.budagavi@samsung.com

Rajan Joshi
Samsung Electronics
11488 Tree Hollow Ln
San Diego, CA, 92128
United States of America
Email: rajan_joshi@ieee.org

Kwang Pyo Choi
Samsung Electronics
34, Seongchon-gil, Seocho-gu
Seoul
3573
Republic of Korea
Email: kwangpyo.choi@gmail.com