avtcore
Internet-Draft
Intended status: Standards Track
Expires: 1 September 2025

Y. Lim
M. Park
M. Budagavi
R. Joshi
K. Choi
Samsung Electronics
28 February 2025

RTP payload format for APV
draft-lim-rtp-apv-01

Abstract

This document describes RTP payload format for bitstream encoded with Advanced Professional Video (APC) 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 1 September 2025.

Copyright Notice

Table of Contents

1. Introduction

This document defines the RTP payload format for bitstream encoded with Advanced Professional Video (APV) codec [I-D.lim-apv]. 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 raw video quality

* Low complexity and high throughput intra frame only coding without pixel domain prediction

* Support for high bit-rates 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

* Support for various chroma sampling formats from 4:2:2 to 4:4:4, and bit-depths from 10 to 16

* Support for multiple decoding and re-encoding without severe visual quality degradation

2. Terms

2.1. Terms and definitions

* access unit (AU): a collection of PBUs 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 and frame header corresponding to the syntax structure specified in section 5.3.4 of [I-D.lim-apv]

* 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 may be 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 sqaure block of samples of a frame in 4:0:0 color format, or a square block of four samples of a frame in 4:4:4:4 color format

* 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 scaling of transform coefficient levels

* 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 units

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

* syntax element: an element of data represented in the bitstream

* syntax structure: zero or more syntax elements present together in the 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 [RFC2119].

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 tiles are also encoded independently from other tiles so that parallel processing of tiles is possible. A tile is further divided into a 16 pixel x 16 pixel size macroblock whch 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 selfcontained. As there are cases that there are more than one video frames corresponding to a specific time, the access unit is designed to carrry multiple video frames and metadata associated to such video frames corresponding to a single time in a signle selfcontained unit. The access unit is consisted 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 access unit is defined in section 5.3.1 of [I-D.lim-apv].

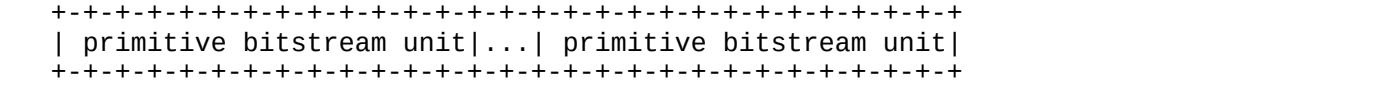

Figure 1: A conceptual structure of APV access unit

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.

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 type of data to be carried in a PBU is listed in Table 1. The detailed syntax and semantics of PBU is specificed in section 5.3.2 and 5.3.3 of [I-D.lim-apv].

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 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 [I-D.lim-apv], respectively.

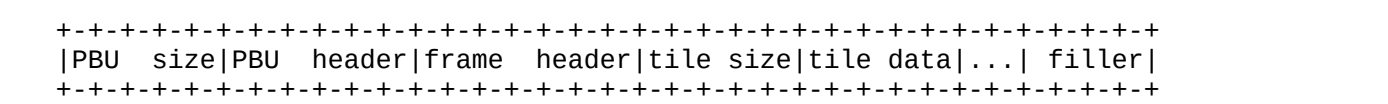

Figure 3: A conceptual structure of APV PBU containing an encoded video frame

4.2.4. Structure of APV PBU for access unit information

When PBU carries access unit information, PBU data carries access unit information specified in 5.3.9 of [I-D.lim-apv]. 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 [I-D.lim-apv]. Metadata starts with its size and it can include more than one type of data consist of type, length and value of data. The list of type of metadata is defined in the section 10.3 of [I-D.lim-apv].

4.2.6. Structure of APV PBU for filler

When PBU carries filler, PBU data carries filler specified in 5.3.11 of [I-D.lim-apv]. Filler can be used to make empty space with a certain size within an access unit to make start position each frames aligned to be a mulitiple 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 10.2 of [I-D.lim-apv]. The length of the AU size field, au_size specified in section 10.2 of [I-D.lim-apv], 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 alinged 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 begining or end of any particular internal data structure of an APV bitstream. An example of this mode is shown in Figure 4

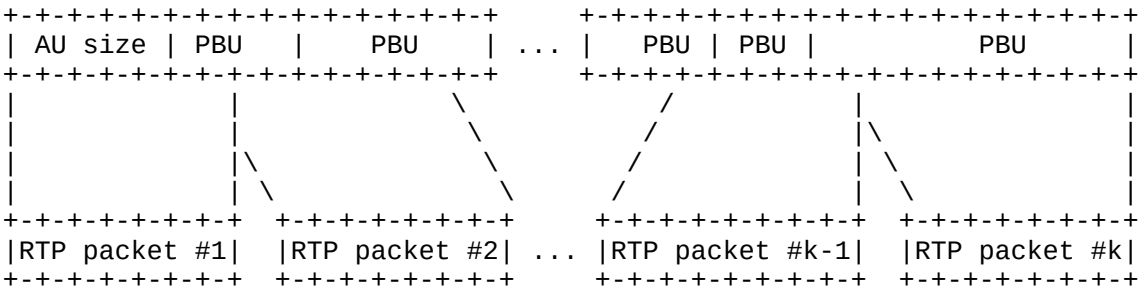

Figure 4: Example of simple mode

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 [I-D.lim-apv], MUST be aligned with the begining 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

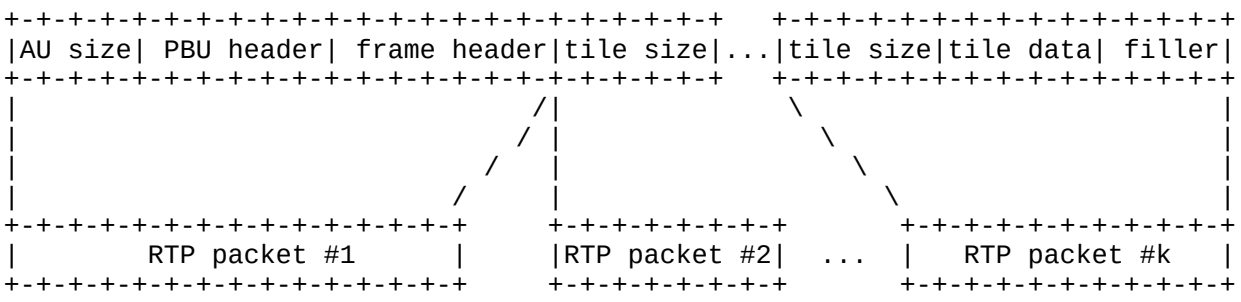

Figure 5: Example of low delay mode

In this mode, frame header can be repeated in any packet containig 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.

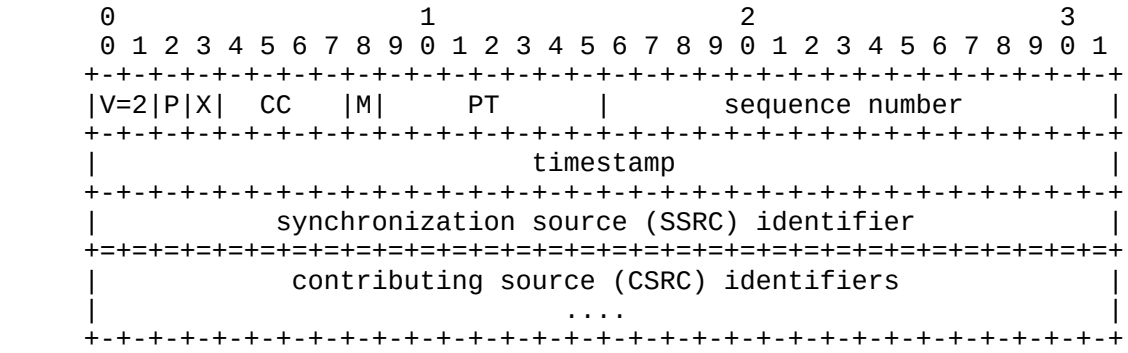

Figure 6: RTP Header According to [RFC3550]

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.

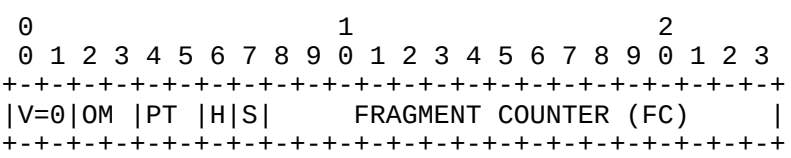

Figure 7: Payload Header

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 [I-D.lim-apv] 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 to be set to ’1’ when a payload carries a frame header at the begnining 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 valuess 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 payload 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 payload 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 payload 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 XXXX.

Interoperability considerations: N/A

Published specification:

Please refer to RFC XXXX and APV standard [I-D.lim-apv].

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. 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 are more than one value of profile_idc field are 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 are 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 the frame header. When there are more than one value of band_idc field are 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 stye. 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 [I-D.lim-apv] and its payload specification at the baseline profile and level 5.1.

It is not expected that [I-D.lim-apv] 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 adpatation, the resulting bitstream MUST be compliant with [I-D.lim-apv].

8. Security Considerations

The scope of this section is limited to the payload format itself and to one feature of [I-D.lim-apv] 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. [I-D.lim-apv] 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 process user-data. In any case, processing of user-data is not required for decoding of APV data. So, receivers does 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>.

[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>.

10.2. Informative References

[I-D.lim-apv]
Lim, Y., Park, M., Budagavi, M., Joshi, R., and K. P. Choi, "Advanced Professional Video", Work in Progress, Internet-Draft, draft-lim-apv-03, 12 December 2024, <https://datatracker.ietf.org/doc/html/draft-lim-apv-03>.

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