AVTCORE WG
IETF 122

Hybrid Meeting
March 20, 2025
Bangkok, Thailand
15:00 - 16:30
Thursday Session III

Mailing list: avtcore@ietf.org
Notes: https://notes.ietf.org/notes-ietf-122-avtcore
MeetEcho link: Meetecho (ietf.org)

Note well

This is a reminder of IETF policies in effect on various topics such as patents or code of conduct. It is only meant to point you in the right direction. Exceptions may apply. The IETF's patent policy and the definition of an IETF "contribution" and "participation" are set forth in BCP 79; please read it carefully.

As a reminder:

- By participating in the IETF, you agree to follow IETF processes and policies.
- If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.
- As a participant in or attendee to any IETF activity you acknowledge that written, audio, video, and photographic records of meetings may be made public.
- Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.
- As a participant or attendee, you agree to work respectfully with other participants; please contact the ombudsteam (https://www.ietf.org/contact/ombudsteam/) if you have questions or concerns about this.

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:

- BCP 9 (Internet Standards Process)
- BCP 25 (Working Group processes)
- BCP 25 (Anti-Harassment Procedures)
- BCP 54 (Code of Conduct)
- BCP 78 (Copyright)
- BCP 79 (Patents, Participation)
- https://www.ietf.org/privacy-policy/(Privacy Policy)

Participant Obligations

- When starting a presentation you MUST say if:
  - There is IPR associated with your draft
- When asking questions or commenting on a draft:
  - You MUST disclose any IPR your employer controls relating to the technology under discussion
- RFC 6701 “Sanctions Available for application to Violators of IETF PR Policy”
  - Describes potential consequences of violating these policies.

Note really well

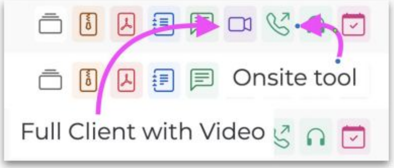
- IETF meetings, virtual meetings, and mailing lists are intended for professional collaboration and networking, as defined in the IETF Guidelines for Conduct (RFC 7154), the IETF Anti-Harassment Policy, and the IETF Anti-Harassment Procedures (RFC 7776). If you have any concerns about observed behavior, please talk to the Ombudsteam, who are available if you need to confidentially raise concerns about harassment or other conduct in the IETF.
- The IETF strives to create and maintain an environment in which people of many different backgrounds are treated with dignity, decency, and respect. Those who participate in the IETF are expected to behave according to professional standards and demonstrate appropriate workplace behavior.
- IETF participants must not engage in harassment while at IETF meetings, virtual meetings, social events, or on mailing lists. Harassment is unwelcome hostile or intimidating behavior -- in particular, speech or behavior that is aggressive or intimidates.
- If you believe you have been harassed, notice that someone else is being harassed, or have any other concerns, you are encouraged to raise your concern in confidence with one of the Ombudspersons.

This session is being recorded

IETF 122 Meeting Tips

In-person participants

- Make sure to sign into the session using the Meetecho (usually the "Meetecho lite" client) from the Datatracker agenda
- Use Meetecho to join the mic queue
- *Keep audio and video off if not using the onsite version*



Remote participants

- Make sure your audio and video are off unless you are chairing or presenting during a session
- Use of a headset is strongly recommended

This session is being recorded

IETF 122 Remote Meeting Tips

- Enter the queue with  , leave with 
- When you are called on, you need to enable your audio to be heard.
- Audio is enabled by unmuting  and disabled by muting 
- Video can also be enabled, but it is separate from audio.
- Video is encouraged to help comprehension but not required.

Resources for IETF 122

- Information about IETF 122
  https://www.ietf.org/how/meetings/122
- Agenda
  https://datatracker.ietf.org/meeting/agenda
- If you need technical assistance, see the Reporting Issues page:
  http://www.ietf.org/how/meetings/issues/

About this meeting

- Agenda: https://datatracker.ietf.org/doc/agenda-122-avtcore/
- Notes: https://notes.ietf.org/notes-ietf-122-avtcore
- WG Chairs
  - Jonathan Lennox
  - Marius Kleidl
- Zulip Scribe: Jonathan Lennox
- Note takers:
  - Youngkwon Lim
  - Mo Zanaty

In Memoriam

Our former co-chair, Bernard Aboba, sadly passed away on February 1. We will remember him fondly.

Draft Status

- Published RFCs
  - RFC 9751 (was draft-ietf-avtcore-rtp-payload-registry)
- RFC Editor Queue
  - draft-ietf-payload-vp9 (AUTH48-DONE)
  - draft-ietf-avtext-lrr (AUTH48-DONE)
  - draft-ietf-avtext-framemarking (AUTH48)
    - Question to the WG about URI change
- Publication Requested
  - draft-ietf-avtcore-rtp-j2k-scl (WGLC Summary)
    - Stephan Wenger has volunteered to shepherd
- Completed WGLC (ended October 22, 2024)
  - draft-ietf-avtcore-rtcp-green-metadata
    - A comment from Eric Språng on March 5

Draft Status (cont'd)

- Second WGLC in progress (ends March 27 2025)
  - draft-ietf-avtcore-rtp-v3c
- Adopted
  - draft-ietf-avtcore-abs-capture-time
  - draft-ietf-avtcore-rtp-haptics
  - draft-ietf-avtcore-hevc-webrtc
    - Philipp Hancke taking over as editor
  - draft-ietf-avtcore-rtp-over-quic
  - draft-ietf-avtcore-rtp-volumetric-media-roi (Summary of CFA)
    - Needs submission as a WG draft
- Expired, may be parked
  - draft-ietf-avtcore-rtp-sframe
  - Call for interest on 12 February 2025; will be parked if none

AVTCORE GitHub Setup

- Organization created: https://github.com/ietf-wg-avtcore
- Recently adopted drafts can create or transfer repositories within the new hierarchy
  - Done:
    - draft-ietf-avtcore-rtp-haptics
    - draft-ietf-avtcore-rtp-j2k-scl
    - draft-ietf-avtcore-hevc-webrtc
    - draft-ietf-avtcore-abs-capture-time
    - draft-dawkins-avtcore-sdp-roq
    - draft-ietf-avtcore-rtp-v3c
  - Still to be transferred:
    - draft-ietf-avtcore-rtcp-green-metadata
    - draft-ietf-avtcore-rtp-volumetric-media-roi
    - *draft-ietf-avtcore-rtp-sframe*
- Once transferred, they should added to the “Activity this week” e-mail
  - A pull request against ietf-github-services/activity-summary

Agenda

1. Preliminaries (Chairs, 15 min)
   Note Well, Note Takers, Agenda Bashing, Draft status, Action items
2. RTP Payload for V3C (Lauri Ilola, 5 min)
   https://datatracker.ietf.org/doc/html/draft-ietf-avtcore-rtp-v3c
3. RTP over QUIC (Mathis Engelbart, 10 min)
   https://datatracker.ietf.org/doc/html/draft-ietf-avtcore-rtp-over-quic
4. SDP Offer/Answer for RTP over QUIC (Spencer Dawkins, 15 min)
   https://datatracker.ietf.org/doc/html/draft-dawkins-avtcore-sdp-roq
5. RTP Payload for Haptics (Hyunsik Yang, 5 min)
   https://datatracker.ietf.org/doc/html/draft-ietf-avtcore-rtp-haptics
6. RTP Payload for V-DMC (Hyunsik Yang, 10 min)
   https://datatracker.ietf.org/doc/html/draft-hsyang-avtcore-rtp-vdmc
7. RTP Payload for APV (Youngkwon Lim, 10 min)
   https://datatracker.ietf.org/doc/html/draft-lim-rtp-apv
8. Automatic Corruption Detection (Erik Språng, 10 min)
   https://datatracker.ietf.org/doc/draft-sprang-avtcore-corruption-detection
9. Frame Acknowledgement (Erik Språng, 5 min)
10. RTP Payload Format for Geometry-based Point Cloud Compression (Mathis Engelbart, as time permits)
    https://datatracker.ietf.org/doc/draft-engelbart-avtcore-rtp-gpcc
11. Wrapup and Next Steps (Chairs, 5 min)

RTP Payload for V3C

draft-ietf-avtcore-rtp-v3c

L. Ilola

L. Kondrad

Start time: 15:15
End time: 15:20

V3C RTP Payload format

- Second WGLC ongoing
  - Deadline midnight Pacific Time on Thursday, March 27, 2025
  - Latest changes to spec Sep 2024
- Document and editing history is available on github
  - The repository has been transferred under AVTCORE
- Summary of issues closed since first WGLC
  - [#16] Questions about SDP examples
  - [#17] Clarification of packetization-modes needed
  - [#18] Issues with SDP grouping
  - [#19] Clarification of 2-D video stream encapsulation
  - [#20] Reconsider the need for the three transmission modes (SRST, MRST and MRMT)
  - [#21] Evaluate the need for out-of-order decoding (i.e. decoding order number, DON)
  - [#22] V3C specific parameters can't be used with the 2D video media lines
  - [#23] Clarify the nature V3C specific parameters (send properties vs. receive capabilities)
  - [#24] Fix PT value for the BUNDLE example in 9.3
  - [#25] Incorrect PT type in answer example in 9.3
  - [#26] Clarifications of Offer&Answer model and backwards compatibility
- New issue
  - [#28] IANA Feedback

RTP over QUIC

https://datatracker.ietf.org/doc/html/draft-ietf-avtcore-rtp-over-quic

https://datatracker.ietf.org/doc/draft-dawkins-avtcore-sdp-rtp-quic/

https://datatracker.ietf.org/doc/draft-dawkins-avtcore-sdp-rtp-quic-issues/

Mathis Engelbart, Jörg Ott, Spencer Dawkins

Start time: 15:20
End time: 15:30

Updates

- 234: Update acknowledgment section (merged)
- 235: Replace "heading" with "registry group"
  - Result of early IANA review

RoQ and Interop Tests (Details in Wiki)

- New implementation in Gstreamer (gst-plugins-rs)
  - Interoperable with mengelbart/roq
  - Not (yet?) tested with other implementations

Next Steps

- Continue Interop tests
- SDP - see next presentation
- WGLC?

SDP Offer/Answer for RTP over QUIC

draft-dawkins-avtcore-sdp-roq

Spencer Dawkins, Victor Pascual

Start time: 15:30
End time: 15:45

What is in the draft now?

- Where the draft is now
  - Submitted as https://datatracker.ietf.org/doc/html/draft-hawkins-avtcore-sdp-roq-00
  - https://github.com/ietf-wg-avtcore/sdp-roq (now in the AVTCORE organization)
  - And, of course, there are further edits in the editor's copy here
- What the draft looks like
  - Definition of new protocol identifiers (added to IANA SDP protos registry)
  - Definitions of new attribute-names (added to IANA Attribute-Names registry)
  - Normative considerations for reusing existing attribute-names with RoQ
  - Non-normative Implementation Topics
  - A Non-Normative QUIC/RTP/AVPF Example
  - Security Considerations
  - IANA Considerations

Unchanged from -00

Definition of New Proto Identifiers

- QUIC transport proto
- QUIC/RTP/AVP
- QUIC/RTP/AVPF
- QUIC/RTP/SAVP
- QUIC/RTP/SAVPF

No surprises so far …

We don't think these are controversial, ***but***

see reasoning for QUIC/RTP/SAVP and QUIC/RTP/SAVPF

in Security Considerations

Unchanged from -00

Normative

Definition of New Attribute-Names for RoQ

- RoQ QUIC-DATAGRAMs Attribute
  - Because QUIC DATAGRAMs aren't part of the core QUIC protocol
  - This is advisory in SDP - the QUIC handshake actually controls support
- RoQ Flow Identifiers
  - This is required for RoQ over QUIC streams and QUIC DATAGRAMs

Unchanged from -00

Normative

Considerations for reusing existing SDP attribute-names

- setup - this says which RoQ endpoint opens a QUIC connection
- **tls-id - allows either end to reopen a QUIC connection**
  - **Replaces "connection" from -00, which did not allow this**
- fingerprint - this helps a RoQ connection resist person-in-the-middle attacks
- rtcp-mux - this allows a RoQ connection to carry both RTP and RTCP

Normative

Implementation Topics

- We don't THINK these require special consideration for RoQ (but we're happy to learn more!)
- Bundling Considerations
    - We don't think RoQ is special, as long as only one path is active
- Implications of Replacing RTCP Feedback with QUIC Feedback
    - We're not sure what we can say about standardized QUIC feedback via a QUIC API
- Implications of Congestion Control
    - We don't think RoQ is special, unless we need to recommend circuit breaker
- Implications of using ICE with RoQ
    - We assume RTCP makes QUIC PING unnecessary
    - We assume ICE Consent Freshness is a good thing

Minimal changes from -00

Informative

A Non-Normative QUIC/RTP/AVPF Example

- So, this is starting with an example in RFC 8833
  - We add a=quic-datagrams as a hint to the RoQ receiver
  - We add a=setup:passive to wait for a QUIC Initial packet
  - **We add a=tls-id so the endpoints can decide to reuse a QUIC connection**
  - We add a=fingerprint to resist person-in-the-middle attacks
  - We change m=from RTP/AVP to QUIC/RTP/AVPF
  - We add a=rtcp-mux so we can tell RTP and RTCP apart
  - We add roq-flow-ids so we can tell RoQ flows apart

Informative

Security Considerations

- We are including implicit security considerations from ~20 normative references
- We define "secure AVP" QUIC protos to avoid RTP middlebox media exposure
- **We describe the issues involved in translating QUIC/RTP/AVP(F) to SAVP(F)**
- **We describe using "some approach like SFRAME" for end-to-end media security**

Normative

IANA Considerations

- We are defining new [SDP-protos] and [SDP-attribute-name] values
- Most of the details are in the body of the document, at least for now

Unchanged from -00

Normative

Open Questions Identified in the Editor's Version

- ICE versus other NAT traversal mechanisms
- DATAGRAM signaling in SDP
- Interaction with MASQUE UDP-Connect
- Recommendations for RTCP feedback replacements with QUIC feedback
- **Recommendations for RoQ on multiple QUIC paths**

Feedback from Harald on editor's version

- Better phrasing in Abstract and Introduction
- Not changing **ABNF** to add new attribute values. Example: **RFC 7850**
- There's a mismatch on flow-id for RTCP between RoQ and SDP-RoQ
- The description of fingerprint's benefit is overstated
- The description of rtcp-mux needs to be clearer
- The description of interaction with ICE needs to be clearer
  - There's a knock-on effect about choosing between UDP, TCP, and QUIC
- The AVP profile security discussion should be in the body, not at the end

Feedback from the editors on Harald's feedback

- Better phrasing in Abstract and Introduction
- Not changing **ABNF** to add new attribute values. Example: RFC 7850
- There's a mismatch on flow-id for RTCP between RoQ and SDP-RoQ
- The description of fingerprint's benefit is overstated
- The description of rtcp-mux needs to be clearer
- The description of interaction with ICE needs to be clearer
  - There's a knock-on effect about choosing between UDP, TCP, and QUIC
- The AVP profile security discussion should be in the body, not at the end

All of these are clear and actionable

Feedback from Jonathan

- "Vastly underestimated complexity of interaction between RoQ and BUNDLE"
  - BUNDLE muxes multiple media descriptions onto a transport connection
  - But so does Flow-ID 😁
  - How do these work together, if at all? Examples, please!
- How can ICE distinguish between a UDP/RTP path and a UDP/QUIC path?
  - Jonathan isn't convinced this is possible
  - Harald allowed for the possibility this should be out of scope for now
  - We still **need** a way to use ICE to set up peer-to-peer RoQ connections

Next steps

- Obviously, respond to review comments from Harald and Jonathan!
  - Expect us to produce a -01 draft before the next AVTCORE interim
- **After that**, expect a request that the chairs poll for adoption of -01
  - We got really helpful comments at the interim
  - We asked for reviewers at the interim
  - We got really helpful review comments
  - It seems that having the working group work on the draft is useful!
- Also obviously, we invite other early reviewers
  - **If you'd like to help, please raise your hand now**
- See you at the interim!

RTP Payload for Haptics

draft-ietf-avtcore-rtp-haptics

HS Yang, X. de Foy

Start time: 15:45
End time: 15:50

Status

- **Published WG Draft Version 04 on Feb/March**
    - Updated SDP parameter section and FIR(Full Intra Request) support (based on Jonathan’s comment)

Updated section Summary

- Added FIR information(Detail in section 5.4)

- In some multimedia conference scenarios using an RTP video mixer (e.g., when adding or selecting a new source), it is recommended to use Full Intra Request (FIR) feedback messages with Haptic {{RFC5104}}. The purpose of the FIR message is to force an encoder to send a decoder refresh point at the earliest opportunity. In the context of haptics, an appropriate decoder refresh point is an initialization MIHS unit. The initialization MIHS unit point enables a decoder to be reset to a known state and be able decode all MIHS units following it.

Updated section Summary

- **Updated SDP section(detail in section 7.1)**
  - We have reviewed all SDP parameters again and precisely defined the usage of each parameter.

  - **Type of properties represented by the SDP parameters**

    When used for a unidirectional stream, the SDP parameters represent the properties of the sender (on the sending side) and of the receiver (on the receiving side). When used for a sendrecv stream, the SDP parameters represent the properties of the receiver.

Updated section Summary

- **Describe attribute of SDP parameters**
  - hmpg-profile/level/version in offer/answer: must use symmetric values, cannot be re-negotiated
  - Other SDP parameters: can be changed in answer, can be re-negotiated. A receiver will ignore parts of the stream that it does not support.

The properties expressed using SDP parameters other than 'hmpg-ver', 'hmpg-profile' and 'hmpg-lvl' are provided as recommendations for efficient data transmission and are not binding, meaning that a sender is encouraged but not required to conform to the parameters specified by the receiver. These properties may be set to different values in offers and answers. These properties may be updated in subsequent offers or answers. Any receiver compliant with [ISO.IEC.23090-31] must accept any stream with a compatible version, profile and level. A receiver supporting a more general profile will accept a stream corresponding to a same or less general profile (e.g., "main" is more general than "simple- parametric"). A receiver supporting a given level will accept a stream corresponding to a same or lower level. A receiver supporting a given version will accept a stream corresponding to the same version and may accept other versions. A receiver may ignore any part of a received stream, e.g., that it does not have support for rendering.

Next Step

- It is the opinion of the authors that the received comments are addressed and that the draft is ready for WGLC
- Feedback and any comments are always welcome.

RTP Payload for V-DMC

draft-hsyang-avtcore-rtp-vdmc

HS Yang, X. de Foy

Start time: 15:50
End time: 16:00

Status

the IETF 121 AVTCORE meeting, and additional comments received offline since then. Published a new Individual Draft Version 01 on February 28.

Here are comments we received from last IETF 121

- **Regarding the relationship with the V3C document(Jonathan)(updated)**
  The scope of the document has been revised to clarify it, and discussions with the V3C authors have been completed.
- **Clear definition of the intended use of SDP parameters.(Harald)(updated)**
  we clarified the distinction between the V3C parameters inherited from the V3C draft, which are unchanged for the most part (except the unit type which has a new value), and the new SDP parameters derived from the specifications, for the base mesh and for displacement.
- **Sync with timestamps(**Gurtej**) & DON (Mo Zonaty)**
  Currently, decoding order and rendering order are the same. In addition, the timestamp is sufficient for synchronization among bitstreams within a single RTP session.

Status

Here are comments we received offline since IETF 121. Thanks to Danilo (Sony) and Lukasz Kondrad (Nokia) for reviewing the draft.

- **MPEG specification-based updates:**
  - Definition of a dedicated unit type for the displacement sub-stream
  - Synchronization of parameters with the MPEG specification
  - Added a new unit type (V3C_ADD) for Arithmetic Coding-based displacement
  - MPEG specification will reach the FDIS (i.e., stable) stage around July. (Currently, It is in DIS stage)
  - Renamed the displacement sub-stream to distinguish AC-based displacement from video-based displacement.

Status

- **VDMC grouping has been added**
  - VDMC grouping has been added to distinguish VDMC groups from V3C groups since they have different sub-streams.

- **Reviewed all SDP parameters:**
  - Based on comments from IETF 121, we classified SDP parameters and re-arranged SDP parameters.

- **IANA requested update**
  - Registry → Registry group / Sub registry → registry

Next Step

- *Suggestions and feedback are welcome*
- *We are looking for people interested in reviewing or participating in the draft.*

RTP Payload for APV

draft-lim-rtp-apv

Y. Lim, M. Park, M. Budagavi, R. Joshi, K. Choi

Start time: 16:00
End time: 16:10

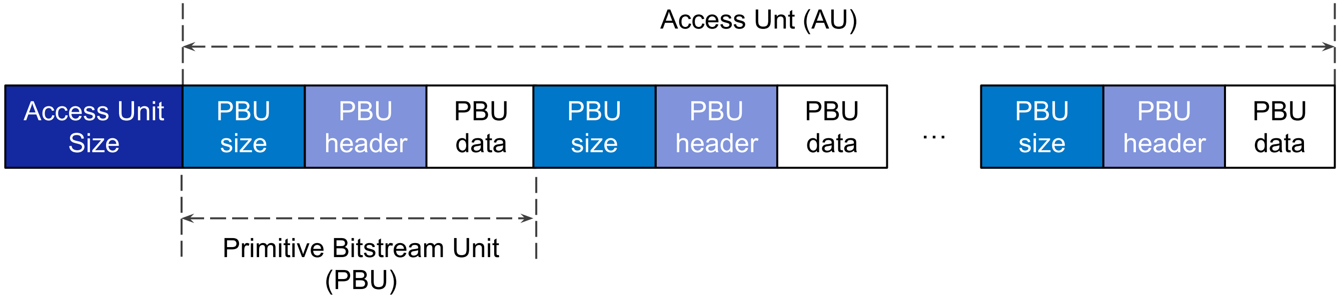
Advanced Professional Video

- Professional (mezzanine) video codec designed for resource constrained devices
- High throughput (up to several Gbps), high fidelity oriented
- Better compression efficiency than other professional video codec in the market
- IPR risk free and open source software provided
- Academy Software Foundation (ASWF) has started a open source project
  - https://github.com/AcademySoftwareFoundation/openapv

- I-D Submitted for Informational RFC (draft-lim-apv)
- I-D has been revised to -03
  - In ISE review

draft-lim-apv-03

- New bitstream structure




```
syntax code
---------------------------
pbu_header(){
    pbu_type
    group_id
    reserved_zero_8bits
}
```

- Two layer : An access unit is composed of one or more primitive bitstream unit(s)
  - Access Unit Size can be added for storage and delivery when necessary
- Type of PBUs
  - Coded frame: primary frame, non-primary frame, preview frame, depth frame, alpha frame
  - Metadata: access unit information, metadata
  - Filler: filler
- PBU header provides type and group_id for association with access unit information
  - Simple and fixed structure of PBU size and PBU header enables fast scanning and skip to the target data
- Filler can be added anywhere for fixed size AU or to avoid rewriting of entire bitstream while editing

draft-lim-apv-03

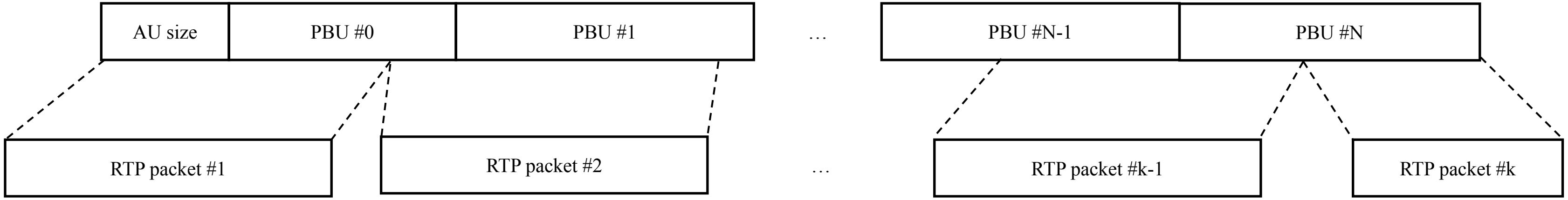
- PBU with coded frame data

PBU size	PBU header	frame header	tile size	tile data	tile size	tile data	…	tile size	tile data	filler

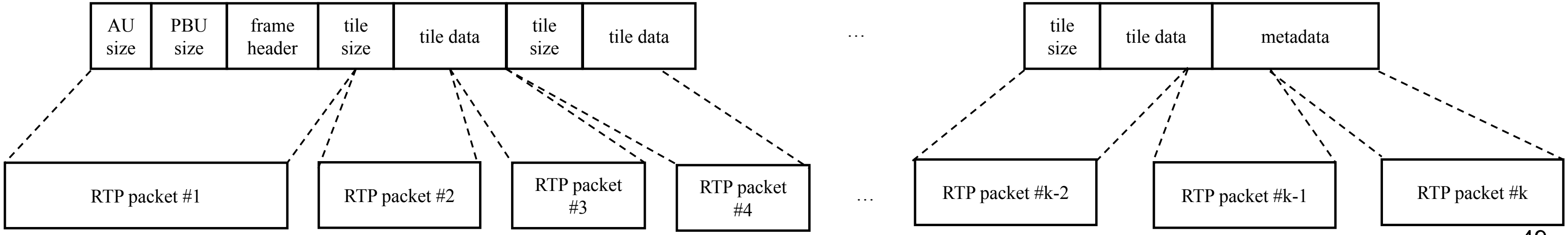
- Each frame is self-contained
    - Frame header provide all configuration information for a frame
- Frame is composed with one or more tiles
    - Information about the structure of tiles is provided by frame header
- Filler can be added optionally

Two packetization modes

- AU MUST not start in the middle of a RTP packet
- Simple mode
    - No alignment with internal data structure of APV bitstream



- Low delay mode
    - Beginning of tile size field MUST be aligned with beginning of RTP packet payloads.



Usage of RTP packet header

- Marker bit (M): 1 bit
  - set to 1 for the last packet of a frame.

- Timestamp: 32 bits
  - The RTP timestamp is set to the sampling time of a frame.
  - A 90 kHz clock rate MUST be used.

Design of payload header

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=0|OM |PT |H|S|     FRAGMENT COUNTER (FC)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Version (V) : 2 bits
  - This field indicates the version of the payload header.

- Operation Mode (OM) : 2 bits
  - This field indicates which operation mode is used
    - 00b : reserved
    - 01b : simple mode as defined in Section 5.2
    - 10b : low-delay mode as defined in Section 5.3
    - 11b : reserved

Design of payload header

```
0                   1                   2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=0|OM |PT |H|S|      FRAGMENT COUNTER (FC)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Payload Type (PT) : 2 bits
  - 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

Design of payload header

```
0                   1                   2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=0|OM |PT |H|S|     FRAGMENT COUNTER (FC)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Frame Header repeated (H) : 1 bit
  - This field indicates that the frame header is repeated in this payload.
  - only used in low delay mode
  - a copy of frame header is added to the end of the payload
- Static Frame Header (S) : 1 bit
  - the values of frame header data is identical except the value of capture_time_distance field with the immediately preceding frame data sent.
- Fragment Counter (FC) : 16 bit
  - the number of remaining payload excluding the current one carrying the current frame or tile

Payload format parameters

- Media type registration
  - Type name: video
  - Subtype name: apv
  - Required parameters: N/A
  - Optional parameters: profile-id, level-id, band-id

- SDP parameters
  - generally one way APV session is offered over RTP using SDP in a declarative style.

Next steps

- Please review and make comments/suggestions
- Considering to request for WG adoption when I-D on APV becomes stable

Automatic Corruption Detection

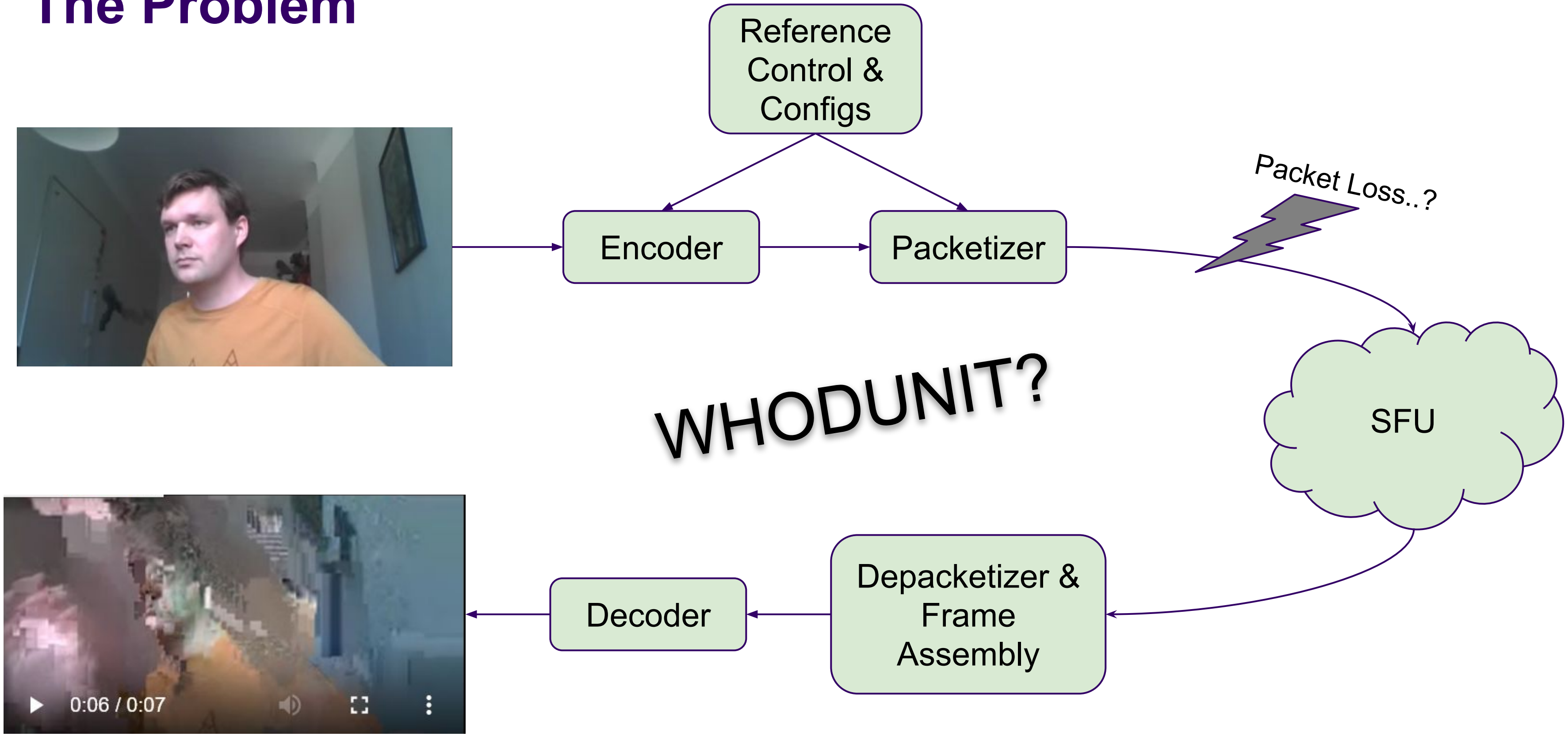
draft-sprang-avtcore-corruption-detection

Note: IPR disclosure in the process of being filed.

Erik Språng

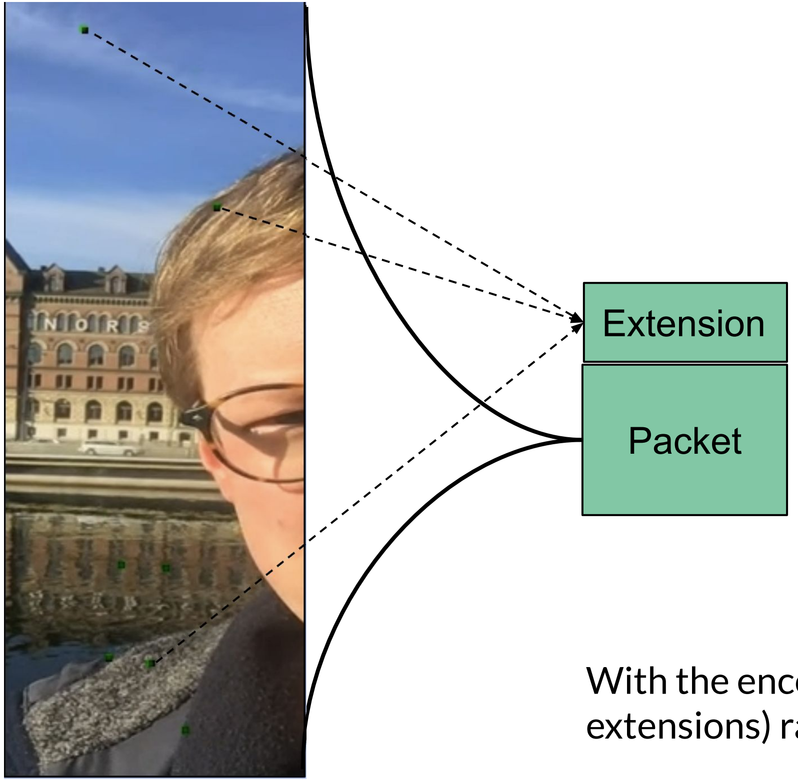
Start time: 16:10
End time: 16:20

The Problem



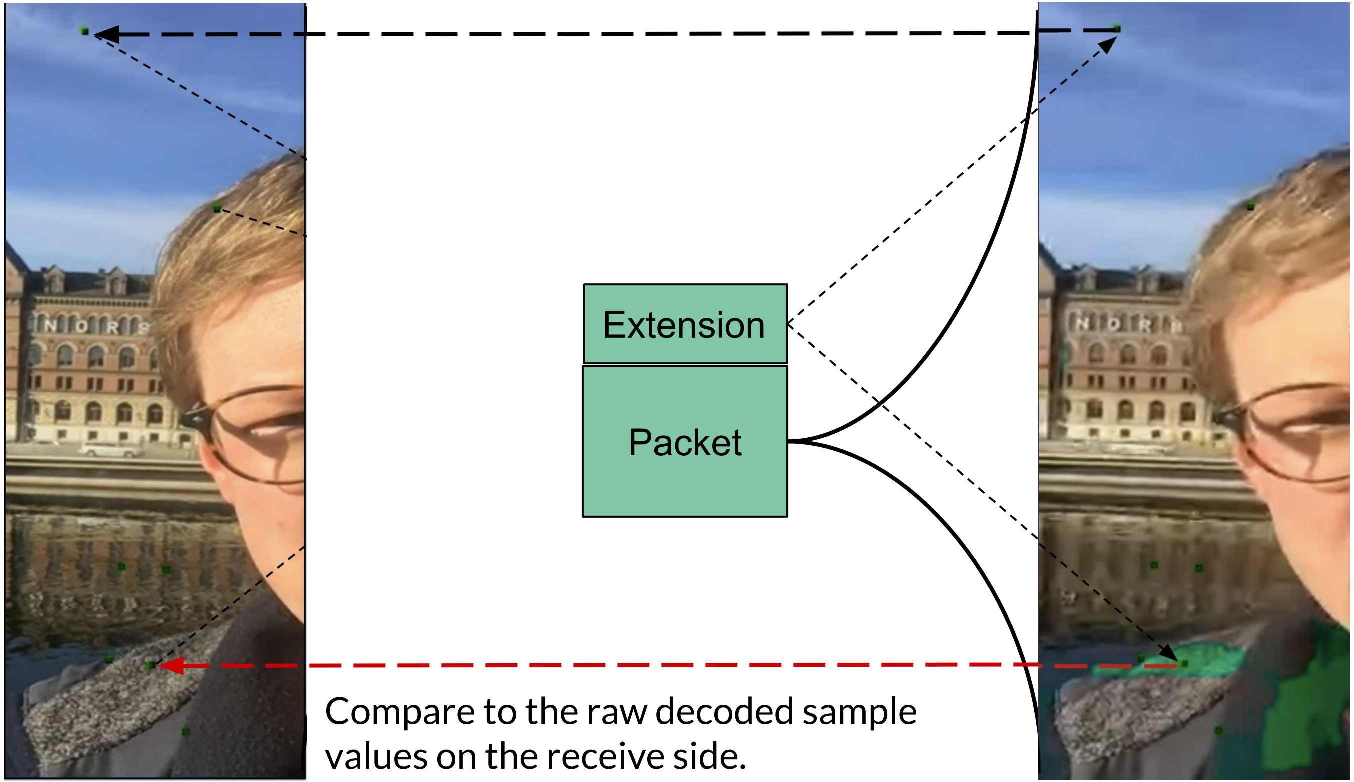
Reference Control & Configs
Encoder
Packetizer
Packet Loss..?
SFU
WHODUNIT?
Depacketizer & Frame Assembly
Decoder
0:06 / 0:07

Corruption Detection - General Concept



With the encoded images, piggy-back a few (13 with one-byte extensions) randomly selected samples as raw values.

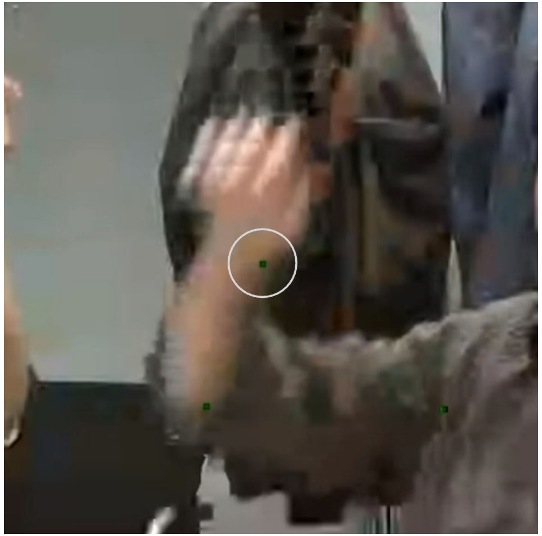
Corruption Detection - General Concept



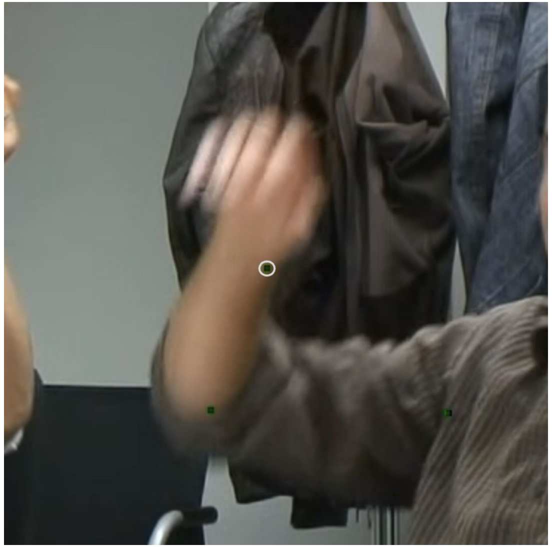
Compare to the raw decoded sample values on the receive side.

Corruption Detection - Filtering

The extension has fields indicating filter size and allowed error thresholds, so that expected distortions from lossy compression can be suppressed.



Large distortion - average over large area



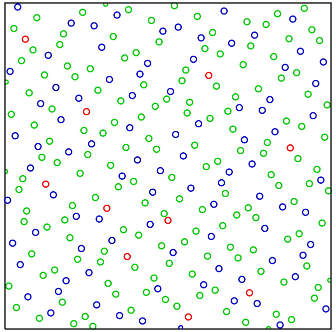
Low distortion - average over small area

Header Extension Layout

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID   | len=7 |B|  seq index  |     std dev     | Y err | UV err|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    sample 0   |    sample 1   |    ... up to sample <= 12     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

B + seq index: Index into pseudo-random Halton Sequence.

Used by both sender and receiver to determine coordinates to sample for this frame. Each sample belongs to a single image plane with a uniform distribution.
I420 => double the number of samples in Y vs U&V.



Header Extension Layout

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID   | len=7 |B|  seq index  |    std dev    | Y err | UV err|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    sample 0   |    sample 1   |    ... up to sample <= 12     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

std dev: Size of Gaussian blur kernel to apply (on both sides).
Y err + UV err: Allowed error thresholds for luma and chroma channels.

Header Extension Layout

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID   | len=7 |B|  seq index  |     std dev     | Y err | UV err|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    sample 0   |    sample 1   |    ... up to sample <= 12     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

sample N: A raw sample from the input, at location determined by (seq index + N) and filtered using a Gaussian with the given std dev.

Selecting the Thresholds

The allowed error thresholds and std dev should be selected such that the filtered input / output **diff <= the error threshold in for 99.5% of samples**.

No precise method is defined to do this. A straightforward way (implemented in Chromium) uses a mapping based on QP.

A priori knowledge of the codec implementation can yield tighter heuristics.

Using the Difference

Current implementation:

$$score = \frac{\sum_{i=0}^{N-1} max(|r_i - s_i| - d_i, 0)^2}{2N}$$

Where r is a receive-side sample, s is a send-side sample and d is the error threshold for that sample (depends on plane). Seems to work well, but can likely be improved upon without having to change the protocol.

See also WebRTC stats.

A Note on Security

This extension slowly leaks information since it is essentially a *really* sparse encoding of the original image. Given enough samples and static scene…

Therefore encryption of the header extension is vital. RFC6904 currently implemented and used in Chromium. RFC9335 is an even better choice and should be used instead, once implementations are ready.

Further Reading / Next Steps

Implemented in Chromium M132+, using an experimental header extension URI: http://www.webrtc.org/experiments/rtp-hdrext/corruption-detection
(Currently opt-in)

Relevant links:

- https://datatracker.ietf.org/doc/draft-sprang-avtcore-corruption-detection/
- https://www.w3.org/TR/webrtc-stats/#dom-rtcinboundrtpstreamstats-totalcorruptionprobability
- https://issues.webrtc.org/issues/358039777

Looking for interested parties to review and comment.

Frame Acknowledgment

draft-sprang-avtcore-frame-acknowledgement

Erik Språng

Start time: 16:20
End time: 16:25

Frame Acknowledgement

A method to request and provide feedback of whether a receiver has *decoded* given video frames. Main use case: Variations on **Long Term References (LTR)**

- Requests done in-band using an RTP Header Extension.
  - Request feedback from a given frame until “now”
  - Provides loss resilience
- Feedback provided via RTCP message.
  - A bit-vector of status for the requested frames
- Frame IDs provided by other means
  - Preferred: Dependency Descriptors

Frame Acknowledgement

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID   | len=1 |         First Frame ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P| FMT=12  |   PT = 205    |          length               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of RTCP packet sender                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SSRC of media source                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Start Frame ID         |L|   length    |  status + pad |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ...                                                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Frame Acknowledgement

Design Requirements:

- Codec agnostic
- Payload Invariant
- Uses Frame Identifiers
- Send-side Controlled
- Loss Resilient
- Low Delay
- Low Overhead

Frame Acknowledgement

Existing message formats:

- NACK / TWCC / RFC8888 / etc
  - Deals with packets, not frames.
  - No guarantee a frame will be decoded, even if it is received.
- RPSI
  - Highly codec specific. Not defined for AV1, forbidden for H.266.
- LNTF (Expired draft)
  - Better than any of the above, but…
  - Cannot handle out-of-order state, making some scenarios difficult, including multi-way LTR and single SSRC simulcast (“S-Modes”).

Frame Acknowledgement

This is a very early draft - main goal is to find who is interested in this problem domain and what your requirements are.

Please have a look at https://datatracker.ietf.org/doc/draft-sprang-avtcore-frame-acknowledgement/

Looking forward to hearing your feedback!

RTP Payload Format for Geometry-based Point Cloud Compression

draft-engelbart-avtcore-rtp-gpcc

Mathis Engelbart, Jörg Ott, Lukasz Kondrad

As time permits

Background: Point Clouds

- Data Structure used to represent three dimensional data
- List of points in three dimensional space
- Each point may be associated with zero or more attributes
  e.g., Color, Reflectance, …
- Acquired by, e.g., LiDAR, Radar, Multiple Camera setups
- Example use case: 3D representation of a vehicle's surrounding environment

Background: G-PCC

- ISO/IEC 23090-9: *Information technology — Coded representation of immersive media — Part 9: Geometry-based point cloud compression*
- Geometry data and attributes are coded independently
- 10 Types of Data Units (DUs)
    - Parameter Sets
    - Geometry Data Units
    - Attribute Data Units
    - …
- Annex-B describes Type-Length-Value Bitstream format

RTP Payload Format

- Standard RTP header usage
- Timestamp defined as the earliest sampling time of the point cloud frame
- Payload Header includes two type fields:
  - Single Unit / Aggregation Unit / Part of Fragmentation Unit
  - Type of Data Unit in packet payload
- Aggregation Unit packets additionally prefix each Data Unit with a ~~variable~~ **fixed size** length integer length field describing the length of the following Data Unit in bytes

Wrapup and Next Steps

- Action Items
- Next Steps (authors)

Start time: 16:25
End time: 16:30

Thank you

Special thanks to:

The Secretariat, WG Participants & ADs