AVTCORE WG

Virtual Interim

December 15, 2022

08:00 - 10:00 AM Pacific Time

Mailing list: avtcore@ietf.org

Meeting info: https://datatracker.ietf.org/meeting/interim-2022-avtcore-04/session/avtcore

This session is being recorded

Virtual Interim 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.
- Keep audio and video off unless you are chairing or presenting.
- Use of a headset is strongly recommended.

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)

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.

About this meeting

- Agenda and other meeting info:
  https://datatracker.ietf.org/meeting/interim-2022-avtcore-04/session/avtcore
- Secretariat: mtd@jabber.ietf.org
- WG Chairs: Jonathan Lennox & Bernard Aboba
- Zulip Scribe: Jonathan Lennox
- Note takers: ?

Agenda

1. Tips and resources, Note Well, Note Takers, Agenda Bashing, Draft status, CfAs (Chairs, 15 min)
2. RTP Control Protocol (RTCP) Messages for Green Metadata (Yong He, 10 min)
3. MAX_STREAMS and the Frame/Stream Model (B. Aboba, 10 min)
4. Scoping For RTP over QUIC (J. Ott, M. Engelbart, S. Dawkins, 20 min)
5. RTP over QUIC (J. Ott, M. Engelbart, 20 min)
6. Codec Agnostic Payload Format, (Youenn Fablet, 20 min)
7. Wrapup and Next Steps (Chairs, 15 min)

Draft Status

- Published
  - RFC 9071: was draft-ietf-avtcore-multi-party-rtt-mix
  - RFC 9134: was draft-ietf-payload-rtp-jpegxs
- RFC Editor Queue
  - draft-ietf-payload-vp9 (MISSREF)
  - draft-ietf-avtcore-cryptex (AUTH48)
  - draft-ietf-avtcore-rtp-vvc (AUTH48)
- In IETF Last Call
  - draft-ietf-avtcore-rtp-scip (Completes December 19, 2022)
- Waiting for AD Go-Ahead::Revised I-D Needed
  - draft-ietf-avtext-framemarking

Draft Status (cont'd)

- Publication Requested
  - draft-ietf-avtcore-rfc7983bis
- Adopted
  - draft-ietf-avtcore-rtp-over-quic
  - draft-ietf-avtcore-rtp-evc (expired on 8/08/2021. Can authors update?)
  - draft-ilola-avtcore-rtp-v3c (should be renamed draft-ietf-avtcore-rtp-v3c)

CfA on “RTP Control Protocol (RTCP) Messages for Green Metadata”

- CfA announcement: https://mailarchive.ietf.org/arch/msg/avt/ILvQ-P006O3IIwyiAC-OQeeHpiU/
- CfA summary: https://mailarchive.ietf.org/arch/msg/avt/tb9nUDIckwW34ZqIpBIz_GvtAIY/
- CfA concluded November 30, 2022. Replies:
  - Affirmative:
    - Yong He (October 31)
    - Christian Herglotz (November 2)
    - Shuai Zhao (November 8)
    - Lauri Ilola (November 23, with comment)
    - Srinivas Gudumasu (November 30)
  - No objection:
    - Magnus Westerlund (November 23, with comment)

Magnus's Comment

Re: [AVTCORE] Call for Adoption (CfA) of "RTP Control Protocol (RTCP) Messages for Green Metadata"

Magnus Westerlund <magnus.westerlund@ericsson.com> Wed, 23 November 2022 12:32 UTCShow header

Hi,

I don't see any issues with adopting this work. However, I think the work should be given a clearer title than Green Meta Data. From reading the draft it appears to be a mechanism that independently of intention of usage to save energy or simply to adopt better to dynamic events on the receiver endpoint that could be improved by adjusting temporal and spatial encoding is about requesting frame rate and resolution. Thus, I would suggest that the title is changed. I think including the energy conservation potential is a great introduction motivation for the work. But the title as currently given is unclear. I was expecting some feedback reporting on current energy consumption based on the current title.

I would also ask if the current data in the notification message is the right for this. The actual frame rate and resolution will be given by the media encoding itself and its meta data. Thus, from a functional perspective is it sufficient to indicate that the media sender has received feedback from a set of requestors. I would think a list of SSRC sending the TSRR and the sequence number would be sufficient. In addition, the FCI entry for the TSRN becomes redundant when you include multiple FCIs. As the Picture Width, Height and Frame rate are all for the SSRC sending the TSSN repeating them for each SSRC that sent a TSRR is wasteful.

Magnus's Comment (cont'd)

I also think the draft needs to consider what happens in the case the media sender adjust the resolution or frame rate due to congestion, resolution changes in the source etc. I guess they will just occur without any potential to send a TSRN.

I would also note that the RTP Middlebox section will need quite a lot of work to more to consider the various models and what it actually means for the middlebox. There is an important difference in just forwarding it which might make sense for a SFU use case, but for an RTP mixer that is combining multiple sources in some form depending on how it does it is needs to either act on it directly or make a decision on how to influence one or more upstream SSRCs that provides media to it for its mixing operation.

Cheers

Magnus Westerlund

Yong He's Response

Re: [AVTCORE] Call for Adoption (CfA) of "RTP Control Protocol (RTCP) Messages for Green Metadata"

Yong He <yonghe@qti.qualcomm.com> Wed, 23 November 2022 18:27 UTCShow header

Hi Magnus,

Thanks for your comments.

It makes sense to change the title as the messages may be deployed beyond the energy saving.

The proposed TSRN follows the Temporal-Spatial Trade-off Notification design in RFC5104, I'll look into it with the rest of your comments further.

Best Regards,
Yong

Lauri's Comment

Re: [AVTCORE] Call for Adoption (CfA) of "RTP Control Protocol (RTCP) Messages for Green Metadata"

"Lauri Ilola (Nokia)" <lauri.ilola@nokia.com> Wed, 23 November 2022 08:09 UTCShow header

I support the adoption with comment that it should be clarified how adapting the bitstream over RTCP feedback mechanism is used in collaboration with SDP updates. Adaptation over RTCP feedback messages invalidates the session description that was used to negotiate the original streaming parameters. It should be at least noted that the sender and receiver need to update the session description accordingly as indicated by the RTCP feedback messages.

Kind regards,
-Lauri

Yong He's Response

Re: [AVTCORE] Call for Adoption (CfA) of "RTP Control Protocol (RTCP) Messages for Green Metadata"

Yong He <yonghe@qti.qualcomm.com> Wed, 23 November 2022 18:12 UTCShow header

Hi Lauri,

Thanks for your comment.

I agree additional SDP subclause is needed to include the proposed messages to SDP rtcp-fb attribute and note the sender and receiver need to update the session description, will update the draft accordingly.

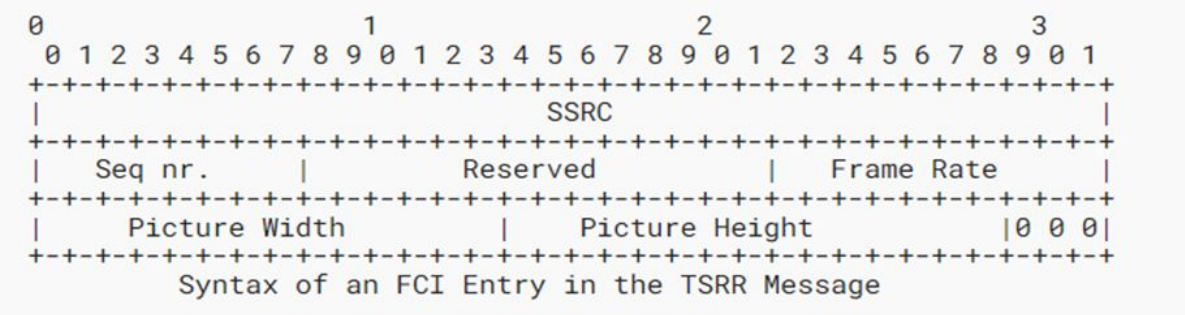
Br, Yong

RTP Control Protocol (RTCP) Messages for Green Metadata

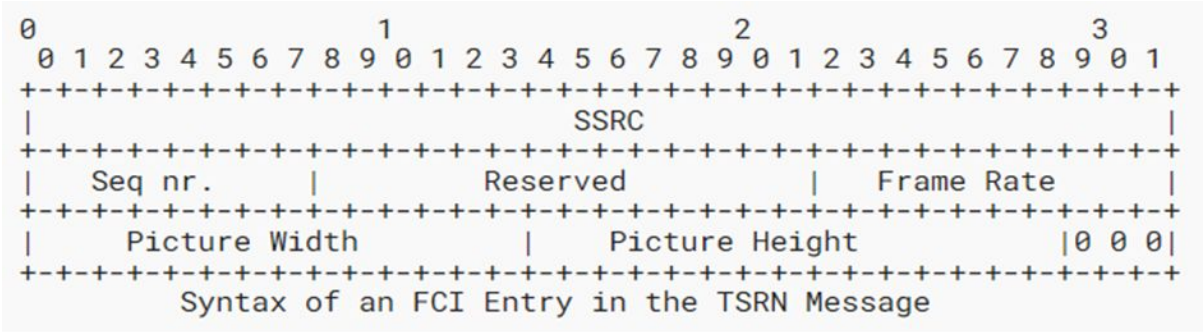
https://datatracker.ietf.org/doc/html/draft-he-avtcore-rtcp-green-metadata

Yong He

RTCP Messages



Temporal-Spatial Resolution Request (TSRR)



Temporal-Spatial Resolution Notification (TSRN)

CfA Comments

- Lauri comment on clarifying how adapting the bitstream over RTCP feedback mechanism is used in collaboration with SDP updates.
- Plan to add SDP definitions sub-clause similar to RFC5104 to define rtcp-fb attribute and parameter for the proposed messages
- Magus comment on the unclear title
- RTCP Message for Spatial and Temporal Resolutions?
- Codec Resolution Control Messages in RTP Audio-Visual Profile with Feedback (AVPF)?
- Magus comment on sending TSSN repeating picture width, height and frame rate is wasteful
- Sending a list of SSRC and Seq nr. saves 32-bits per FCI entry, prefer to be consistent with Temporal-Spatial Trade-off Notification (TSTN) format
- Magnus question on the media sender adjust the resolution or frame rate due to congestion, resolution changes in the source etc.
- Confirm Magnus' guess that they will just occur without any potential to send a TSRM

Next steps

- We ask for WG adoption of the draft.

MAX_STREAMS and the Frame/Stream Model

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

https://webrtc.internaut.com/wc/wtSender10/

Bernard Aboba

A Recent Posting…

- Christian Huitema on the MoQ list:
  - https://mailarchive.ietf.org/arch/msg/moq/hecXJfwaysqbyp85ZUwAT9I5CM0/
  - “Suppose that on the connection I am mixing audio frames (20 per sec, short), video frames (30 per sec, longuish, variable) and background data. The priorities are set to ensure audio streams are sent before video streams, and video streams before data stream. The data are always ready: when a data frame has been sent, the next one can begin. If the data is sent as a single stream, everything is fine, priorities work. But suppose now that the data is sent as a series of data frames, each one on its own stream. That's where I got stumped.”
- Reply from Luke Curley:
  - If I understand your email correctly, the problem is that the receiver doesn't know how many streams will be used. MAX_STREAMS 100 might seem reasonable for one application but prohibitive for another. As far as I understand, the stream limit exists to curtail stream state overhead and context switching. An exponential increase sounds reasonable, but so does starting at a significantly higher initial credit regardless of the application.

Some Questions

- Is there a potential issue with QUIC and WebTransport implementations?
  - There appear to be some use cases where a default MAX_STREAMS limit could be exceeded (e.g. conferencing)
    - Filed WebTransport API Issue 446
  - Experiments indicate that a single RTP stream sent over frame/stream transport will typically not be impacted by a well chosen default MAX_STREAMS limit.
  - **However**, applications that leak resources may encounter problems.
    - With frame/stream transport, it's very important to release resources (close streams, release memory, release reader and writer locks, etc.)
- Is text needed in the RTP over QUIC specification?
  - Filed Issue 49

The Frame/Stream Model

- Send pipeline
  - Sender opens a uni-directional stream.
  - Sender sets a timer with an expiration (RTO)
    - If the timer fires, sender resets the stream
    - RTO values can differ between discardable and non-discardable frames.
  - Sender writes the RTP header + payload to the uni-directional stream
  - Sender closes the stream.
- Receive pipeline
  - Receiver is notified of an incoming uni-directional stream.
  - Receiver reads from the incoming uni-directional stream until:
    - The stream is closed OR
    - The stream is reset.
    - Length field helpful to make sure the frame was completely received.

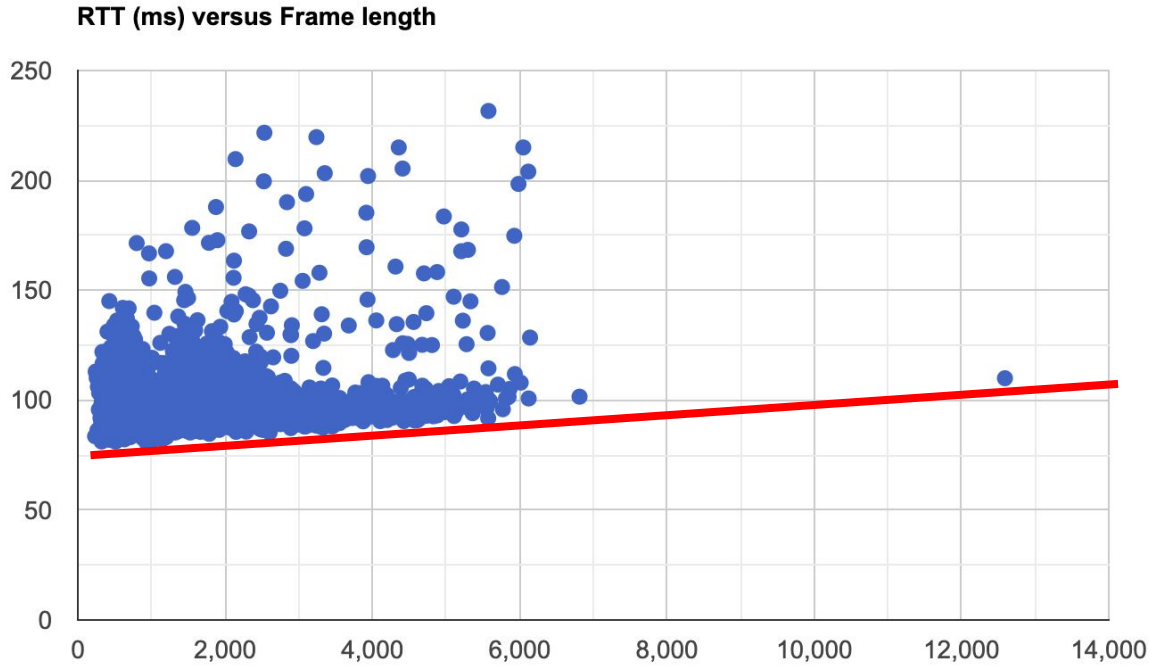
Frame/Stream Experiment

https://webrtc.internaut.com/wc/wtSender10/

- Observations
  - I-frames can be 10x the size of subsequent P-frames.
  - Concurrency desirable for lower glass-glass latency, but implies more open streams.
    - For concurrency, both send and receive pipelines need to avoid blocking.
    - In Javascript, be wary of **await!**
      - `promise.then(f).catch()` is ***not*** the same as **await f**!
  - A ***good*** sign: if multiple P-frames arrive on the receiver prior to complete receipt of the initial I-frame.
    - Lack of re-ordering is a ***symptom of blocking!***
- Results
  - After optimizing frame/stream sender, re-ordering now routinely observed (3-6 events per experiment).
    - I-frame closer to transmission line (cwind large enough to send in a single RTT)
    - See `async writeChunk()` function implementing frame/stream sender
    - More work needed on read pipeline

Frame RTT Graph

- AV1 @ full-Hd with 418 Kbps average bitrate and 30 fps, GoP = 3000, L1T3 scalability mode
- Largest (I-)frame = 12590 octets, median (P-)frame size = 1523 octets
- I-frame is close to the transmission line, indicating that cwind > 12590.



BWE report:
{"count":2283,"loss":0,"reorder":6,"bwe":0,"bwu":417956.483387237,"seqmin":0,"seqmax":2282,"lenmin":234,"lenfquart":727,"lenmedian":1523,"lentquart":2161,"lenmax":12590,"recvsum":3980116}

RTT report:
{"count":2283,"min":80.299,"fquart":92.9,"avg":101.6191081909768,"median":98.1,"tquart":105.399,"max":231.6,"stdev":15.421836998138598,"srtt":115.60558227812218,"rttvar":7.499192348045117,"rto":145.60235167030265}

MAX_STREAMS Limit

- Is it a problem?
  - Currently seeing < 5 concurrent open QUIC streams for each RTP stream (SSRC), assuming equal priority and no loss.
    - Many concurrent streams may be undesirable.
    - Good practice to close streams (and release writers) when done.
  - With loss or receive pipeline improvements, could be higher; with priority, lower.
  - 7x7 grids now common in conferencing applications. So not hard to imagine situations where MAX_STREAMS = 100 could be problematic.
- How does priority affect MAX_STREAMS limit?
  - Priority can be used to limit the number of concurrent streams.
    - Example: wait until I-frame is finished before sending any P-frames.
      - Problem: re-introduces head-of-line blocking.
    - Better: I-frame and N P-frames have equal priority (limit of N+1 concurrent streams)
  - Currently Chromium apportions bandwidth equally between open streams by default.
  - Partial reliability can be implemented independent of priority.

Scoping For 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

Scoping for RTP over QUIC

- Early discussions on RTP over QUIC started out small
  - "RTP over QUIC as a drop-in replacement for RTP over UDP"
  - This idea turns out to be unrealistic for a variety of reasons

- The authors want to Do The Right Thing with RTP over QUIC
  - Proposed goal - do enough for RTP over QUIC to be useful
  - Proposed non-goal - don't do everything that we can imagine

- For now, let's focus on what we know we will need now
  - Don't break anything that we might need later

Is this about right, so far?

How Many Pizza Toppings?

- From 1980s discussion on history of programming languages
  - "I order a pizza with toppings I want" - APL, Pascal
  - "We order a pizza with toppings we want" - COBOL
  - "We order a pizza with all the toppings anyone could want" - PL/I

- Fast forward to the 2020s discussion on RTP over QUIC
  - Let's order a QUIC/RTP pizza with only the toppings that we need!

Planned usages

- In SIP/VoIP environments currently using RTP
- In WebRTC environments currently using RTP
- In WebTransport environments carrying realtime media
  - More details on this in the next section
  - Let's defer discussion on WebTransport for now

Is this about right, so far?

Topologies

- We know we can't support all RFC 7667 topologies
  - Section 3.3 defines point to point using multicast
  - QUIC doesn't provide multicast (yet?)

- We know we want to support some RFC 7667 topologies
  - But we haven't discussed which topologies in detail yet

- -01 version describes supported RTP topologies in general terms
  - Proposal - we analyze the rest of the unicast RFC 7667 topologies
  - (I added Issue #47 for this)

Is this about right, so far?

Relevant QUIC Extensions

- Datagrams
  - RFC 9221 - has been part of the RTP over QUIC plan for a while
  - -01 assumes the use of datagrams

- QUIC Timestamps
  - Currently an Individual Draft
  - Would allow better loss detection and congestion control

Is this about right, so far?

AVT Profiles

- Proposals have ranged from "AVPF" to "[S]AVP[F]"
  - "AVPF"- provides extended RTCP-based feedback
  - "AVP" - minimizes RTCP-based feedback (relying on QUIC feedback)
  - "SAVP" and "SAVPF" - for ease of relay/forwarding to UDP/RTP

- Proposals for full proto definition
  - UDP/QUIC/[S]AVP[F] - following all of the conventions
  - TCP/QUIC/[S]AVP[F] - if all your ICE candidates are TCP candidates
  - stream/dgram/both as attributes, to limit the combinatorial-ness

- Next draft-dawkins-avtcore-sdp-rtp-quic-latest will register all of these

Is this about right, so far?

QUIC Congestion Control

- Proposal - define an ALPN that isn't "h3"
  - Currently "rtp-mux-quic" in -01
  - Avoid the most awkward aspects of nested RTP and "h3" control loops

- Rely on "rtp-mux-quic" endpoints to do "something appropriate"
  - -01 references SCReAM and NADA as examples

Is this about right, so far?

If we finish all of those … 😎

- Mapping between QUIC feedback and AVP/AVPF feedback (#48)
- Working out the details of QUIC connection interaction with ICE (#50)

Is this about right, so far?

Thank you for sharing your thoughts!

… and see you in Yokohama 😁

RTP over QUIC

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

Mathis Engelbart, Jörg Ott

Multiplexing

- Past discussions: How to signal the use of RTP-over-QUIC?
  - Which ALPN(s) to use?
  - Does the choice of an ALPN determine the multiplexing?
  - How much to do and define in this document?
  - To a certain extent a scoping discussion

- How to multiplex RTP-over-QUIC with non RTP streams and datagrams?
  - Prefixing explicit “session identifiers”?
  - How to ensure compatibility with other formats?

- Implications on the use of RTP-over-QUIC?
  - Want to remain lightweight and independent

Multiplexing

- RTP packets for one RTP session in a QUIC connection
  - One RTP session per QUIC connection (this is easy, but …)
  - Can be encapsulated in a stream, or in datagrams
- RTP packets for multiple RTP sessions in a QUIC connection
  - Can be encapsulated in a stream, in datagrams, or in both
  - Would require some sort of session identifier
- RTCP packets
  - In a separate RTP session
  - Multiplexed with RTP packets of the same RTP session
- With other protocols
  - rfc7983bis - QUIC, DTLS, RTP, RTCP, STUN, TURN, ZRTP

Multiplexing Options

- WebTransport
- Partial WebTransport
- RTP/RTCP only

WebTransport

- Transport Features: Sessions, Datagrams, Uni-/Bidirectional Streams

- Transport Properties
  - Unreliable data delivery not required
  - Aborting streams MAY result in retransmissions
  - Datagrams MAY be retransmitted

- Transport Requirements
  - TLS or equivalent
  - Congestion Control
  - **Multiple Sessions**
  - **Indicate client origin to server**
  - **Describe server endpoint location using a URI**

WebTransport

- Transport protocols used in WebTransport have to implement origins and URIs, but does that have any implications on the application layer (RTP)?
- What does *indicate client origin to server* mean for RTP over QUIC?
- What does *describe server endpoint location using a URI* mean for RTP over QUIC?
- WebTransport is currently defined for HTTP/2 and HTTP/3
  - => Does it make sense to depend on WebTransport (and thus HTTP/2 and HTTP/3), just to get RTP over QUIC done?

Partial WebTransport

- Reuse parts of the WebTransport encapsulation
- Avoid dependency on HTTP
- Ignore everything that is not relevant to RTP over QUIC
- Does this make sense?
  - Can we reference anything from WebTransport without saying HTTP?
  - How much different is it from current approach using flow IDs?

Only RTP/RTCP

- Only allow RTP/RTCP multiplexing, nothing else (for now)
- Keep flow identifier for multiplexing many RTP sessions and RTCP
- Register “rtp-quic” ALPN
- Other documents can specify new ALPN token(s) for multiplexing with other protocols
  - Need to define how to multiplex with RTP/RTCP
  - One option could be to assign (ranges of) flow identifiers to different protocols

Encrypted media RTP Architecture

Youenn Fablet

https://github.com/youennf/codec-agnostic-rtp-payload-format/blob/main/draft-encrypted-media-rtp-architecture.md

Goal

- Enable end-to-end encryption of media over RTP
- Define an architecture so as to
  - Identify requirements / what needs to be done
  - Provide hints at how it can be done
  - draft-encrypted-media-rtp-architecture.md

Sender side architecture (1/2)

- Existing architecture
  - Packetizer generates packets from
    - Frame metadata
    - Encoded frame content
  - Intermediaries extract information from packet headers and packet payloads
    - Small use of encoded frame metadata
- E2E encrypted architecture
  - Packetizer cannot extract information from encoded frame content
    - This content is opaque
  - Intermediaries cannot extract information fro
    - It has to be replaced by more encoded frame metadata

Sender side architecture (2/2)

- Traditionally, the packetizer is tied with the encoder
- In e2 encryption pipelines, the packetizer becomes tied to the media encryptor

- Define the relationship between the packetizer and media encryptor
  - Interface between media encryptor and encrypted media packetizer
  - Changes to the media encryptor should not require a new packetizer
    - SFrame format, pre-SFrame format & SPacket should all fit in

- WebCodecs is a Web API that defines interfaces for media decoders & encoders
  - Use WebCodecs interfaces to model the encryptor/packetizer relationship
  - Let's focus on video for now

Sender side requirements

- The media encryptor MUST provide
  - Content metadata useful to intermediaries (& no need for e2e encryption)
    - VideoFrame members: color space, orientation…
    - EncodedVideoChunk members: codec, frame type, timestamp…
    - EncodedVideoChunkMetadata: decoder config, SVC metadata…
  - Encrypted content as a byte array

- The media packetizer MUST transmit and generate
  - RTP headers from content metadata only
    - CVO from VideoFrame orientation, timestamp from EncodedVideoChunk…
  - RTP payloads without analysis of the encrypted content
    - Splitting based on MTU or information provided by Media encryptor
  - Underlying payload type corresponding to the codec used before encryption

Sender side SVC requirements

- The media encryptor MUST
  - Process the encoded frame in individually decodable units
  - Encrypt separately each decodable unit
  - Provide to the packetizer metadata to identify the decodable unit

- The packetizer MUST
  - transmit the decodable unit metadata
    - Dependency descriptor for instance

```
dictionary EncodedVideoChunkMetadata {

// Number for identifying this frame in |dependsOnIds| and |chainLinks| (for other chunks).
unsigned short frameNumber;

// List of frameNumbers that this chunk depends on.
// Used to detect/handle network loss. Decoding out of order is an error.
list<unsigned long> dependsOnIds;

// ID of the spatial layer this chunk belongs to.
unsigned long spatialLayerId;

// ID of the temporal layer this chunk belongs to.
unsigned long temporalLayerId;

// List of decoder targets this frame participates in.
// Used to know whether this frame should be sent (forwarded) to a given receiver depending
// on what decode targets the receiver is expecting.
// Decode target is a numerical index determined by the encoder. No commitment that a
// particular number implies a given layer.
list<unsigned long> decodeTargets;

// Mapping of decode target -> the last important frame to decode prior to "this"
// frame for the given decode target.
// Used to ensure we preserve decode order for the desired decode target. It is insufficient
// to simply satisfy the dependencies for the current frame. See example.
map<unsigned long, unsigned long> chainLinks;

};
```

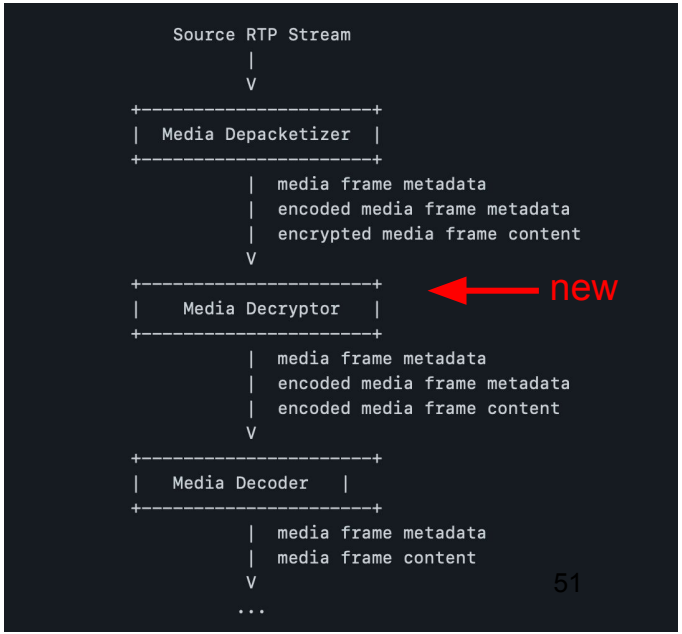
Packet-forwarding unit

- Packet forwarding unit processing requires
  - Computation of which decodable unit a received packet belongs to
    - Underlying codec type, marker bit, timestamp
  - Decodable unit metadata to decide whether to forward the decodable unit
    - Dependency descriptor
  - Computation of decodable unit encrypted content for recording purposes

Receiver side architecture

- Media depacketizer has similar requirements as packet-forwarding unit
  - Reconstruction of decodable unit encrypted content
  - Identification of underlying media decoder
  - Early processing of packets based on RTP headers
    - FIR, Retransmission requests

- Media depacketizer MUST be able to use existing RTP-Based redundancy mechanisms



Proof of feasibility

- SFrame applied on each EncodedVideoChunk provided by encoder
  - SFrame as pass-through for all related metadata
- RTP packetizer generates
  - Dependency descriptor RTP header extension
    - From EncodedVideoChunkMetadata information
  - Encrypted content split based on MTU
    - Underlying payload type prepended on each chunk

- SDP negotiation via additional types
  - `audio/encrypted`
  - `video/encrypted`

```
m=video 9 UDP/TLS/RTP/SAVPF 96 97 98 99 100 101
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=setup:actpass
a=mid:1
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:repaired-rtp-stream-id
a=extmap:4 https://aomediacodec.github.io/av1-rtp-spec/#dependency-descriptor-rtp-header-extension
a=sendrecv
a=rtpmap:96 vp9/90000
a=rtpmap:97 vp8/90000
a=rtpmap:98 encrypted/90000
a=rtpmap:99 rtx/90000
a=fmtp:99 apt=96
a=rtpmap:100 rtx/90000
a=fmtp:100 apt=97
a=rtpmap:101 rtx/90000
a=fmtp:101 apt=98
```

SFU repacketization

- SFU might have clients with different MTUs
- SFU might have to repacketize packets
  - It can wait for missing packets if received out of order
  - It can do suboptimal repacketization
  - It can use a packetization format that allows repacketization without waiting for out of order packets
  - It can use RTP header extensions for the same repacketization purposes
- Should it be in scope of this work?
  - Large to small packet repacketization, small to large packet repacketization

Temporary conclusion

- Supporting end-to-end encrypted content over RTP is desired
- It can be done even though the particular encrypted transform may vary
  - Be it SFrame or SPacket
- Current architecture model provides a basis for what and how we want to support packetization of encrypted content
  - Are we ready for starting defining solutions?

If We Are Ready… (Peter's Slides)

Major design decision is: what metadata to we include?

If We Are Ready… (Peter's Slides)

Major design decision is: what metadata to we include?

- **Payload Type** of decrypted media: helps avoid APT explosion
  - optional or required? (not need when your only codec is Opus)
  - header extension or payload prefix?
- **Frame Number**: helps repacketization of out-of-order packets
  - probably optional
  - header extension or payload prefix?
  - how many bits?
- **Offset/Index**: helps repacketization of out-of-order packets
  - probably optional
  - probably payload prefix
  - how many bits?
  - offset or index?

Wrapup and Next Steps

- Action Items
  - Item 1
  - Item 2
- Next Steps
  - Step 1
  - Step 2

Thank you

Special thanks to:

The Secretariat, WG Participants & ADs