Abstract

This document describes these new SDP "proto" attribute values: "QUIC", "QUIC/RTP/SAVP", "QUIC/RTP/AVPF", and "QUIC/RTP/SAVPF", and describes how SDP Offer/Answer can be used to set up an RTP connection using QUIC as a transport protocol.

These proto values are necessary to allow the use of QUIC as an underlying transport protocol for applications such as SIP and WebRTC that commonly use SDP as a session signaling protocol to set up RTP connections.

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 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components
This document describes these new SDP "proto" attribute values: "QUIC", "QUIC/RTP/SAVP", "QUIC/RTP/AVPF", and "QUIC/RTP/SAVPF", and describes how SDP Offer/Answer ([RFC3264]) can be used to set up an RTP ([RFC3550]) connection using QUIC ([RFC9000] and related specifications) as a transport protocol.

These proto values are necessary to allow the use of QUIC as an underlying transport protocol for applications such as SIP ([RFC3261]) and WebRTC ([RFC8825]) that commonly use SDP as a session signaling protocol to set up RTP connections.

1.1. Notes for Readers

(Note to RFC Editor - if this document ever reaches you, please remove this section)
This document is intended for publication as a standards-track RFC in the IETF stream, but has not been adopted by any IETF working group, and does not carry any special status within the IETF.

1.2. Terminology

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

1.3. Scope of this document

This document focuses on the IANA registration and description of the RTP sessions using SDP Offer/Answer, as would be the case for many current RTP applications in common use, such as SIP ([RFC3261]) and WebRTC ([RFC8825]).

This document is intended as complementary to drafts such as [I-D.engelbart-rtp-over-quic], which largely focus on RTP/RTCP encapsulation in QUIC, so that the SDP experts can focus on SDP offer/answer aspects, and the RTP experts can focus on RTP/RTCP encapsulation aspects.

1.4. Contribution and Discussion Venues for this draft.

(Note to RFC Editor - if this document ever reaches you, please remove this section)

With the concurrence of the AVTCORE and MMUSIC working group co-chairs, this document should be discussed in the AVTCORE working group, in the same venue where RTP over QUIC proposals are being discussed. When proposals for RTP over SIP have stabilized in AVTCORE, this document will be sent to the MMUSIC working group for review by SDP experts, but SDP-specific comments are welcomed at any time.

Readers are also invited to open issues and send pull requests with contributed text for this document in the GitHub repository at https://github.com/SpencerDawkins/sdp-rtp-quic. The direct link to the list of issues is https://github.com/SpencerDawkins/sdp-rtp-quic/issues.
1.5. Assumptions for this document

This document assumes that for RTP-over-QUIC, it is useful to register these AVP profiles using QUIC, in order to allow existing SIP and RTCWEB RTP applications to migrate more easily to QUIC:

* RTP/SAVP ("The Secure Real-time Transport Protocol (SRTP)") as defined in [RFC3711].
* RTP/AVPF ("Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)") as defined in [RFC4585].
* RTP/SAVPF ("Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)") as defined in [RFC5124].

This document assumes that any implementation adding support for RTP-over-QUIC could reasonably also add support for BUNDLE ([RFC8843]) and "rtcp-mux" ([RFC5761]), so these capabilities are not mentioned further in this document.

1.5.1. An Aside on Secure AVP Profiles in an RTP Over QUIC Context

Existing RTP implementations have the choice for any given RTP connection to exchange either unencrypted RTP streams (using AVP profiles such as RTP/AVPF) or encrypted RTP streams (using AVP profiles such as RTP/SAVPF).

An RTP implementation that uses QUIC as its underlying transport protocol will always send an RTP stream that is encrypted between the two QUIC endpoints, so some RTP implementations may be tempted to exchange unencrypted RTP as an encrypted QUIC payload, reasoning that QUIC protection will be sufficient.

One nuance here is that QUIC is heavily encrypted between two QUIC endpoints, with the very minimal exception of the invariant header fields described in [RFC8999], but as described in [RFC7667], many RTP applications use middleboxes for a variety of reasons, and some of these topologies (for example, media translation) require that the middlebox understand the RTP payload.

These middleboxes are explicitly addressed, and the QUIC cryptographic handshake described in [RFC9001] takes place between the RTP endpoint and the RTP middlebox. After the QUIC cryptographic handshake has succeeded, the RTP middlebox has access to the RTP in the QUIC payload, and can perform whatever translations are appropriate before forwarding the RTP stream to another RTP endpoint. However, if the
RTP sender uses one of the "insecure" AVPs, the middlebox does not have any indication that the RTP sender wants the translated RTP stream to be protected by encryption when the middlebox forwards it. That might be fine if the middlebox and RTP endpoint are both using RTP over QUIC, but if the middlebox is performing transport translation as well, the middlebox may also be translating an RTP-over-QUIC stream to RTP-over-UDP.

This specification tries to provide that indication by supporting both "secure" and "insecure" AVPs for RTP over QUIC, so the middlebox that is providing back-to-back RTP sessions as described in [RFC7667] can be aware of the sender’s desire that a translated RTP stream is encrypted regardless of the underlying transport protocol, without always requiring both SRTP and QUIC encryption between each pair of QUIC endpoints for all RTP traffic. That’s one strategy, and it’s certainly possible that other strategies might be safer, cleaner, and/or more useful.

1.6. Open Questions

The current contents of Section 2 and Section 3 would allow an existing RTP/RTCP implementation to make a relatively straightforward transition from "RTP over UDP" to "RTP over QUIC datagrams over UDP", and likewise from "RTCP over UDP" to "RTCP over QUIC datagrams over UDP".

Although it is still early days for RTP over QUIC, things may not be that straightforward. Just limiting our attention to various proposals for "RTP over QUIC" that have already been discussed on the Media Over QUIC IETF mailing list [MOQ] and in various IETF side meetings, we have seen

* a desire to make use of QUIC connection migration in case of path failure between two endpoints

* a desire to replace RTP Round Trip Time (RTT) measurement with something like a proposed QUIC extension for timestamps ([I-D.huitema-quic-ts]) that could be used to measure one-way delays

* a desire to make use of QUIC streams, potentially with QUIC datagrams in the same QUIC connection

* a desire to decouple the RTP state machine and the QUIC state machine, which currently assume they are solely responsible for managing sending rates, without any knowledge of what the other plans to do
* a desire to select a media-focused congestion control mechanism such as "Self-Clocked Rate Adaptation for Multimedia", or SCReAM ([RFC8298]), that can be included in QUIC implementations

* a desire to use RTP over QUIC in peer-to-peer applications, which likely would require extensions to the QUIC protocol for NAT traversal, at a bare minimum

Changes to the SDP signaling in Section 2 and Section 3 may be (and likely would be) needed in order to support any of these desires (or other desires that may surface in the future).

2. Identifiers and Attributes

As much as possible, these are reused from other specifications, with references to the original definitions.

2.1. Protocol Identifiers

2.1.1. The QUIC proto

The ‘QUIC’ protocol identifier is similar to the ‘UDP’ and ‘TCP’ protocol identifiers in that it only describes the transport protocol, and not the upper-layer protocol.

An ‘m’ line that specifies ‘QUIC’ MUST further qualify the application-layer protocol using an fmt identifier, such as "QUIC/RTP/AVPF". Media described using an ‘m’ line containing the ‘QUIC’ protocol identifier are carried using QUIC ([RFC9000]).

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC protocol.

```
media-field =         %s"m" "=" media SP port \
                      \"[/" integer\]\ CRLF
                      \"SP proto 1*(SP fmt) CRLF\"

m= line parameter        parameter value(s)

------------------------------------------------------------------
<media>:                 (unchanged from {{RFC8866}})
<proto>:                 'QUIC'
<port>:                  UDP port number
<fmt>:                   (unchanged from {{RFC8866}})
```

2.1.2. The QUIC/RTP/SAVP proto

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/SAVP protocol.
2.1.3. The QUIC/RTP/AVPF proto

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/AVPF protocol.

```
media-field = %s"m" "=" media SP port \["/" integer\]
             SP proto 1*(SP fmt) CRLF
m= line parameter parameter value(s)
```

```
<media>: (unchanged from {{RFC8866}})
<proto>: 'QUIC/RTP/AVPF'
<port>: UDP port number
<fmt>: (unchanged from {{RFC8866}})
```

2.1.4. The QUIC/RTP/SAVPF proto

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/SAVPF protocol.

```
media-field = %s"m" "=" media SP port \["/" integer\]
             SP proto 1*(SP fmt) CRLF
m= line parameter parameter value(s)
```

```
<media>: (unchanged from {{RFC8866}})
<proto>: 'QUIC/RTP/SAVPF'
<port>: UDP port number
<fmt>: (unchanged from {{RFC8866}})
```

2.2. A QUIC/RTP/AVPF Offer

A complete example of an SDP offer using QUIC/RTP/AVPF might look like:
<table>
<thead>
<tr>
<th>SDP line</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>v=0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>o=jdoe 3724394400 3724394405 IN IP4 198.51.100.1</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>s=Call to John Smith</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>i=SDP Offer #1</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>u=<a href="http://www.jdoe.example.com/home.html">http://www.jdoe.example.com/home.html</a></td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>e=Jane Doe</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>p=+1 617 555-6011</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>c=IN IP4 198.51.100.1</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>t=0 0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>m=audio 49170 RTP/AVP 0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>m=audio 49180 RTP/AVP 0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>m=video 51372 QUIC/RTP/AVPF 99</td>
<td>QUIC transport</td>
</tr>
<tr>
<td>a=setup:passive</td>
<td>will wait for QUIC handshake</td>
</tr>
<tr>
<td></td>
<td>(setup attribute from [RFC4145])</td>
</tr>
<tr>
<td>a=connection:new</td>
<td>don’t want to reuse an existing</td>
</tr>
<tr>
<td></td>
<td>QUIC connection (connection</td>
</tr>
<tr>
<td></td>
<td>attribute from [RFC4145])</td>
</tr>
<tr>
<td>c=IN IP6 2001:db8::2</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>a=rtpmap:99 h266/90000</td>
<td>H.266 VVC codec</td>
</tr>
<tr>
<td></td>
<td>[I-D.ietf-avtcore-rtp-vvc]</td>
</tr>
</tbody>
</table>

Table 1
This example is largely based on an example appearing in [RFC8866], Section 5, but is using QUIC/RTP/AVPF to support a newer codec.

Because QUIC uses connections for both streams and datagrams, we are reusing two session- and media-level SDP attributes from [SDP-attribute-name] that were defined in [RFC4145] for use with TCP: setup and connection.

This example SDP offer might be included in a SIP Invite.

3. IANA Considerations

This document registers these protocols in the proto registry ([SDP-parameters]).

* QUIC (Section 2.1.1)
* QUIC/RTP/SAVP (Section 2.1.2)
* QUIC/RTP/AVPF (Section 2.1.3)
* QUIC/RTP/SAVPF (Section 2.1.4)

3.1. Proto Registrations

IANA is requested to add these protocols to the Session Description Protocol (SDP) Parameters proto registry ([SDP-parameters]).

<table>
<thead>
<tr>
<th>Type</th>
<th>SDP Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>proto</td>
<td>QUIC</td>
<td>RFCXXXX</td>
</tr>
<tr>
<td>proto</td>
<td>QUIC/RTP/SAVP</td>
<td>RFCXXXX</td>
</tr>
<tr>
<td>proto</td>
<td>QUIC/RTP/AVPF</td>
<td>RFCXXXX</td>
</tr>
<tr>
<td>proto</td>
<td>QUIC/RTP/SAVPF</td>
<td>RFCXXXX</td>
</tr>
</tbody>
</table>

Table 2

*Note to the RFC Editor*

Please replace "RFCXXXX" with the assigned RFC number, when that is available, and remove this note.
4. Security Considerations

Security considerations for the QUIC protocol are described in the corresponding section in [RFC9000].

Security considerations for the TLS handshake used to secure QUIC are described in [RFC9001].

Security considerations for SDP are described in the corresponding section in [RFC8866].

Security considerations for SDP offer/answer are described in the corresponding section in [RFC3264].

5. Acknowledgments

My appreciation to the authors of [RFC4145], which served as a model for the initial structure of this document.

Thanks to these folks for helping to improve this draft:

* Colin Perkins

(Your name also could appear here. Please comment and contribute, as per Section 1.4).

6. References

6.1. Normative References


6.2. Informative References

[I-D.engelbart-rtp-over-quic]
Ott, J. and M. Engelbart, "RTP over QUIC", Work in Progress, Internet-Draft, draft-engelbart-rtp-over-quic-01, 25 October 2021,

[I-D.huitema-quic-ts]
Huitema, C., "Quic Timestamps For Measuring One-Way Delays", Work in Progress, Internet-Draft, draft-huitema-quic-ts-06, 12 September 2021,

[I-D.ietf-avtcore-rtp-vvc]
Zhao, S., Wenger, S., Sanchez, Y., Wang, Y., and M. M. Hannuksela, "RTP Payload Format for Versatile Video Coding (VVC)", Work in Progress, Internet-Draft, draft-ietf-
avtcore-rtp-vvc-13, 18 November 2021,

[RFC4145] Yon, D. and G. Camarillo, "TCP-Based Media Transport in
the Session Description Protocol (SDP)", RFC 4145,
DOI 10.17487/RFC4145, September 2005,

Author's Address

Spencer Dawkins
Tencent America LLC
United States of America

Email: spencerdawkins.ietf@gmail.com
Abstract

This document describes these new SDP "proto" attribute values: "QUIC", "QUIC/RTP/SAVP", "QUIC/RTP/AVPF", and "QUIC/RTP/SAVPF", and describes how SDP Offer/Answer can be used to set up an RTP connection using QUIC as a transport protocol.

These proto values are necessary to allow the use of QUIC as an underlying transport protocol for applications that commonly use SDP as a session signaling protocol to set up RTP connections with UDP as its underlying transport protocol, such as SIP and WebRTC.

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 12 March 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

This document describes these new SDP "proto" attribute values: "QUIC", "QUIC/RTP/SAVP", "QUIC/RTP/AVPF", and "QUIC/RTP/SAVPF", and describes how SDP Offer/Answer ([RFC3264]) can be used to set up an RTP ([RFC3550]) connection using QUIC ([RFC9000]) as a transport protocol.

These proto values are necessary to allow the use of QUIC as an underlying transport protocol for applications that commonly use SDP as a session signaling protocol to set up RTP connections with UDP as its underlying transport protocol, such as SIP ([RFC3261]) and WebRTC ([RFC8825]).
1.1. Notes for Readers

This document is intended for publication as a standards-track RFC in the IETF stream, but has not been adopted by any IETF working group, and does not carry any special status within the IETF.

1.2. Terminology

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

1.3. Contribution and Discussion Venues for this draft.

(Note to RFC Editor - if this document ever reaches you, please remove this section)

This document is under development in the Github repository at https://github.com/SpencerDawkins/sdp-rtp-quic.

Readers are invited to open issues and send pull requests with contributed text for this document, or to send them to the author via email.

1.4. Scope of this document

This document focuses on the IANA registration and description of the RTP sessions using SDP Offer/Answer, as would be the case for many current RTP applications in common use, such as SIP ([RFC3261]) and WebRTC ([RFC8825]).

This document is intended as complementary to [I-D.engelbart-rtp-over-quic], which largely focuses on RTP/RTCP encapsulation in QUIC datagrams, so that the SDP experts can focus on SDP offer/answer aspects, and the RTP experts can focus on RTP/RTCP encapsulation aspects.

1.5. Assumptions for this document

This document assumes that for RTP-over-QUIC, it is useful to register these AVP profiles using QUIC, in order to allow existing SIP and RTCWEB RTP applications to migrate more easily to QUIC:

* RTP/SAVP ("The Secure Real-time Transport Protocol (SRTP)"), as defined in [RFC3711].
* RTP/AVPF ("Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)"), as defined in [RFC4585].

* RTP/SAVPF ("Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)"), as defined in [RFC5124].

This document assumes that any implementation adding support for RTP-over-QUIC could reasonably add support for BUNDLE ([RFC8843]), "rtcp-mux" ([RFC5761]).

2. Open Questions (probably not for this draft, but could have implications on SDP Offer/Answer)

* RTP (and RTCP) headers and payloads will be entirely encrypted using QUIC ([RFC9000]), as secured by TLS 1.3 handshake ([RFC9001]), between QUIC endpoints. It’s worth thinking more about how that maps onto expected deployment scenarios like centralized multiparty conferencing, and also whether WebRTC really requires SAVPF with double encryption (i.e. SRTP encryption, and then QUIC encryption). No opinions here yet, just noting the question for now.

* When QUIC establishes connections, it uses IP addresses but then expects applications to use connection IDs to refer to connections, even if the underlying IP addresses change because of NAT binding, and even if the QUIC implementation performs QUIC connection migration itself, so the underlying IP addresses change. RTP applications expect to use IP addresses, not QUIC connection IDs. Must we specify an RTP/RTCP adaptation layer, similar to [I-D.ietf-quic-http] for HTTP/3?

3. Identifiers and Attributes

As much as possible, these are reused from other specifications, with references to the original definitions.

3.1. Protocol Identifiers

3.1.1. The QUIC proto

The 'QUIC' protocol identifier is similar to the 'UDP' and 'TCP' protocol identifiers in that it only describes the transport protocol, and not the upper-layer protocol.
An ‘m’ line that specifies ‘QUIC’ MUST further qualify the application-layer protocol using an fmt identifier, such as “QUIC/RTP/AVPF”. Media described using an ‘m’ line containing the ‘QUIC’ protocol identifier are carried using QUIC ([RFC9000]).

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC protocol.

```plaintext
media-field = %s"m" "=" media SP port 
["/"] integer\]
            SP proto 1*(SP fmt) CRLF
m= line parameter  parameter value(s)

--------------------------------------------------------------------------------------------------
<media>:                     (unchanged from {{RFC8866}})
<proto>:                    ‘QUIC’
<port>:                     UDP port number
<fmt>:                      (unchanged from {{RFC8866}})
```

3.1.2. The QUIC/RTP/SAVP proto

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/SAVP protocol.

```plaintext
media-field = %s"m" "=" media SP port 
["/"] integer\]
            SP proto 1*(SP fmt) CRLF
m= line parameter  parameter value(s)

--------------------------------------------------------------------------------------------------
<media>:                     (unchanged from {{RFC8866}})
<proto>:                    ‘QUIC/RTP/SAVP’
<port>:                     UDP port number
<fmt>:                      (unchanged from {{RFC8866}})
```

3.1.3. The QUIC/RTP/AVPF proto

The following is an update to the ABNF for an ‘m’ line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/AVPF protocol.
3.1.4. The QUIC/RTP/SAVPF proto

The following is an update to the ABNF for an 'm' line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/SAVPF protocol.

```plaintext
media-field = %s"m" "=" media SP port \[/" integer\]
               SP proto 1*(SP fmt) CRLF

m= line parameter parameter value(s)

<media>:                     (unchanged from {{RFC8866}})
<proto>:                     'QUIC/RTP/AVPF'
<port>:                      UDP port number
<fmt>:                       (unchanged from {{RFC8866}})
```

3.2. A QUIC/RTP/AVPF Offer

A complete example of an SDP offer using QUIC/RTP/AVPF might look like:

```plaintext
```
<table>
<thead>
<tr>
<th>SDP line</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>v=0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>o=jdoe 3724394400 3724394405 IN IP4 198.51.100.1</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>s=Call to John Smith</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>i=SDP Offer #1</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>u=<a href="http://www.jdoe.example.com/home.html">http://www.jdoe.example.com/home.html</a></td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>e=Jane Doe</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>p=+1 617 555-6011</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>c=IN IP4 198.51.100.1</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>t=0 0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>m=audio 49170 RTP/AVP 0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>m=audio 49180 RTP/AVP 0</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>m=video 51372 QUIC/RTP/AVPF 99</td>
<td>QUIC transport</td>
</tr>
<tr>
<td>a=setup:passive</td>
<td>will wait for QUIC handshake (setup attribute from [RFC4145])</td>
</tr>
<tr>
<td>a=connection:new</td>
<td>don’t want to reuse an existing QUIC connection (connection attribute from [RFC4145])</td>
</tr>
<tr>
<td>c=IN IP6 2001:db8::2</td>
<td>Same as [RFC8866]</td>
</tr>
<tr>
<td>a=rtpmap:99 h266/90000</td>
<td>H.266 VVC codec [I-D.ietf-avtcore-rtp-vvc]</td>
</tr>
</tbody>
</table>

Table 1
This example is largely based on an example appearing in [RFC8866], Section 5, but is using QUIC/RTP/AVPF to support a newer codec.

Because QUIC uses connections for both streams and datagrams, we are reusing two session- and media-level SDP attributes from [SDP-attribute-name] that were defined in [RFC4145] for use with TCP: setup and connection.

This example SDP offer might be included in a SIP Invite.

4. IANA Considerations

This document registers these protocols in the proto registry ([SDP-parameters]).

* QUIC (Section 3.1.1)
* QUIC/RTP/SAVP (Section 3.1.2)
* QUIC/RTP/AVPF (Section 3.1.3)
* QUIC/RTP/SAVPF (Section 3.1.4)

4.1. Proto Registrations

IANA is requested to add these protocols to the Session Description Protocol (SDP) Parameters proto registry ([SDP-parameters]).

| Type | SDP Name       | Reference |
|------|----------------+-----------|
| proto| QUIC           | RFCXXXX   |
| proto| QUIC/RTP/SAVP  | RFCXXXX   |
| proto| QUIC/RTP/AVPF  | RFCXXXX   |
| proto| QUIC/RTP/SAVPF | RFCXXXX   |

Table 2

*Note to the RFC Editor*

Please replace "RFCXXXX" with the assigned RFC number, when that is available, and remove this note.
5. Security Considerations

Security considerations for the QUIC protocol are described in the corresponding section in [RFC9000].

Security considerations for the TLS handshake used to secure QUIC are described in [RFC9001].

Security considerations for SDP are described in the corresponding section in [RFC8866].

Security considerations for SDP offer/answer are described in the corresponding section in [RFC3264].

6. Acknowledgments

My appreciation to the authors of [RFC4145], which served as a model for the initial structure of this document.

Your name could appear here. Please comment and contribute, as per Section 1.3.

7. References

7.1. Normative References


7.2. Informative References


Author’s Address

Spencer Dawkins
Tencent America LLC
United States of America

Email: spencerdawkins.ietf@gmail.com
Abstract

This document specifies a minimal mapping for encapsulating RTP and RTCP packets within QUIC. It also discusses how to leverage state from the QUIC implementation in the endpoints to reduce the exchange of RTCP packets.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/mengelbart/rtp-over-quic-draft.

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 13 November 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document.
1. Introduction

The Real-time Transport Protocol (RTP) [RFC3550] is generally used to carry real-time media for conversational media sessions, such as video conferences, across the Internet. Since RTP requires real-time delivery and is tolerant to packet losses, the default underlying transport protocol has been UDP, recently with DTLS on top to secure the media exchange and occasionally TCP (and possibly TLS) as a fallback. With the advent of QUIC [RFC9000] and, most notably, its unreliable DATAGRAM extension [RFC9221], another secure transport protocol becomes available. QUIC and its DATAGRAMs combine desirable properties for real-time traffic (e.g., no unnecessary retransmissions, avoiding head-of-line blocking) with a secure end-to-end transport that is also expected to work well through NATs and firewalls.
Moreover, with QUIC’s multiplexing capabilities, reliable and unreliable transport connections as, e.g., needed for WebRTC, can be established with only a single port used at either end of the connection. This document defines a mapping of how to carry RTP over QUIC. The focus is on RTP and RTCP packet mapping and on reducing the amount of RTCP traffic by leveraging state information readily available within a QUIC endpoint. This document also briefly touches upon how to signal media over QUIC using the Session Description Protocol (SDP) [RFC8866].

The scope of this document is limited to unicast RTP/RTCP.

Note that this draft is similar in spirit to but differs in numerous ways from [I-D.draft-hurst-quic-rtp-tunnelling].

2. Terminology and Notation

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

The following terms are used:

Congestion Controller: QUIC specifies a congestion controller in Section 7 of [RFC9002], but the specific requirements for interactive real-time media lead to the development of dedicated congestion control algorithms. In this document, the term congestion controller refers to these algorithms dedicated to real-time applications.

Datagram: Datagrams exist in UDP as well as in QUICs unreliable datagram extension. If not explicitly noted differently, the term datagram in this document refers to a QUIC Datagram as defined in [RFC9221].

Endpoint: A QUIC server or client that participates in an RTP over QUIC session.

Frame: A QUIC frame as defined in [RFC9000].

Media Encoder: An entity that is used by an application to produce a stream of encoded media, which can be packetized in RTP packets to be transmitted over QUIC.

Receiver: An endpoint that receives media in RTP packets and may send or receive RTCP packets.
Sender: An endpoint that sends media in RTP packets and may send or receive RTCP packets.

Packet diagrams in this document use the format defined in Section 1.3 of [RFC9000] to illustrate the order and size of fields.

3. Protocol Overview

This document introduces a mapping of the Real-time Transport Protocol (RTP) to the QUIC transport protocol. RTP over QUIC allows the use of QUIC streams and unreliable QUIC datagrams to transport real-time data, and thus, the QUIC implementation MUST support QUICs unreliable datagram extension, if RTP packets should be sent over QUIC datagrams. Since datagram frames cannot be fragmented, the QUIC implementation MUST also provide a way to query the maximum datagram size so that an application can create RTP packets that always fit into a QUIC datagram frame.

[RFC3550] specifies that RTP sessions need to be transmitted on different transport addresses to allow multiplexing between them. RTP over QUIC uses a different approach to leverage the advantages of QUIC connections without managing a separate QUIC connection per RTP session. QUIC does not provide demultiplexing between different flows on datagrams but suggests that an application implement a demultiplexing mechanism if required. An example of such a mechanism are flow identifiers prepended to each datagram frame as described in Section 2.1 of [I-D.draft-ietf-masque-h3-datagram]. RTP over QUIC uses a flow identifier to replace the network address and port number to multiplex many RTP sessions over the same QUIC connection.

A congestion controller can be plugged in to adapt the media bitrate to the available bandwidth. This document does not mandate any congestion control algorithm. Some examples include Network-Assisted Dynamic Adaptation (NADA) [RFC8698] and Self-Clocked Rate Adaptation for Multimedia (SCReAM) [RFC8298]. These congestion control algorithms require some feedback about the network’s performance to calculate target bitrates. Traditionally this feedback is generated at the receiver and sent back to the sender via RTCP. Since QUIC also collects some metrics about the network’s performance, these metrics can be used to generate the required feedback at the sender-side and provide it to the congestion controller to avoid the additional overhead of the RTCP stream.
4. Encapsulation

QUIC supports two transport methods: reliable streams [RFC9000] and unreliable datagrams [RFC9221]. This document specifies a mapping of RTP to both of the transport modes. The encapsulation format for RTP over QUIC is described in Figure 1.

Section 4.1 and Section 4.2 explain the specifics of mapping of RTP to QUIC streams and QUIC datagrams respectively.

Payload {
  Flow Identifier (i),
  RTP/RTCP Packet (..)
}

Figure 1: RTP over QUIC Payload Format

Flow Identifier: Flow identifier to demultiplex different data flows on the same QUIC connection.

RTP/RTCP Packet: The RTP/RTCP packet to transmit.

For multiplexing different RTP and other data streams on the same QUIC connection, each RTP/RTCP packet is prefixed with a flow identifier. A flow identifier is a QUIC variable-length integer which must be unique per stream.

RTP and RTCP packets of a single RTP session MAY be sent using the same flow identifier (following the procedures defined in [RFC5761], or they MAY be sent using different flow identifiers. The respective mode of operation MUST be indicated using the appropriate signaling, e.g., when using SDP as discussed in Section 7.

RTP and RTCP packets of different RTP sessions MUST be sent using different flow identifiers.

Differentiating RTP/RTCP packets of different RTP sessions from non-RTP/RTCP datagrams is the responsibility of the application by means of appropriate use of flow identifiers and the corresponding signaling.
4.1. QUIC Streams

An application MUST open a new QUIC stream for each Application Data Unit (ADU). Each ADU MUST be encapsulated in a single RTP packet and the application MUST not send more than one RTP packet per stream. Opening a new stream for each packet adds implicit framing to RTP packets, allows to receive packets without strict ordering and gives an application the possibility to cancel certain packets.

Large RTP packets sent on a stream will be fragmented in smaller QUIC frames, that are transmitted reliably and in order, such that a receiving application can read a complete packet from the stream. No retransmission has to be implemented by the application, since QUIC frames that are lost in transit are retransmitted by the QUIC connection. If it is known to either the sender or the receiver, that a packet, which was not yet successfully and completely transmitted, is no longer needed, either side can close the stream.

*Editor’s Note:* We considered adding a framing like the one described in [RFC4571] to send multiple RTP packets on one stream, but we don’t think it is worth the additional overhead only to reduce the number of streams. Moreover, putting multiple ADUs into a single stream would also require defining policies when to use the same (and which) stream and when to open a new one.

4.2. QUIC Datagrams

RTP packets can be sent in QUIC datagrams. QUIC datagrams are an extension to QUIC described in [RFC9221]. QUIC datagrams preserve frame boundaries, thus a single RTP packet can be mapped to a single QUIC datagram, without the need for an additional framing. Senders SHOULD consider the header overhead associated with QUIC datagrams and ensure that the RTP/RTCP packets, including their payloads, QUIC, and IP headers, will fit into path MTU.

If an application wishes to retransmit lost RTP packets, the retransmission has to be implemented by the application by sending a new datagram for the RTP packet, because QUIC datagrams are not retransmitted on loss (see also Section 5.1 for loss signaling).

5. RTCP

The RTP Control Protocol (RTCP) allows RTP senders and receivers to exchange control information to monitor connection statistics and to identify and synchronize streams. Many of the statistics contained in RTCP packets overlap with the connection statistics collected by a QUIC connection. To avoid using up bandwidth for duplicated control information, the information SHOULD only be sent at one protocol
QUIC relies on certain control frames to be sent.

In general, applications MAY send RTCP without any restrictions. This document specifies a baseline for replacing some of the RTCP packet types by mapping the contents to QUIC connection statistics. Future documents can extend this mapping for other RTCP format types. It is RECOMMENDED to expose relevant information from the QUIC layer to the application instead of exchanging additional RTCP packets, where applicable.

This section discusses what information can be exposed from the QUIC connection layer to reduce the RTCP overhead and which type of RTCP messages cannot be replaced by similar feedback from the transport layer. The list of RTCP packets in this section is not exhaustive and similar considerations SHOULD be taken into account before exchanging any other type of RTCP control packets.

*TODO*: Define parameters for SDP to signal RTCP vs. QUIC feedback. Could use RTCP by default and add parameters for "can use QUIC statistics for X".

5.1. Transport Layer Feedback

This section explains how some of the RTCP packet types which are used to signal reception statistics can be replaced by equivalent statistics that are already collected by QUIC. The following list explains how this mapping can be achieved for the individual fields of different RTCP packet types.

QUIC Datagrams are ack-eliciting packets, which means, that an acknowledgment is triggered when a datagram frame is received. Thus, a sender can assume that an RTP packet arrived at the receiver or was lost in transit, using the QUIC acknowledgments of QUIC Datagram frames. In the following, an RTP packet is regarded as acknowledged, when the QUIC Datagram frame that carried the RTP packet, was acknowledged. For RTP packets that are sent over QUIC streams, an RTP packet can be considered acknowledged, when all frames which carried fragments of the RTP packet were acknowledged.

Some of the transport layer feedback that can be implemented in RTCP contains information that is not included in QUIC by default, but can be added via QUIC extensions. One important example are arrival timestamps, which are not part of QUIC’s default acknowledgment frames, but can be added using [I-D.draft-smith-quic-receive-ts] or [I-D.draft-huitema-quic-ts]. Another extension, that can improve the precision of the feedback from QUIC is [I-D.draft-ietf-quic-ack-frequency], which allows a sender to control the delay of acknowledgments sent by the receiver.
* **Receiver Reports** (PT=201, Name=RR, [RFC3550])

  - _Fraction lost_: The fraction of lost packets can be directly inferred from QUIC’s acknowledgments. The calculation SHOULD include all packets up to the acknowledged RTP packet with the highest RTP sequence number. Later packets SHOULD be ignored, since they may still be in flight, unless other QUIC packets that were sent after the datagram frame, were already acknowledged.

  - _Cumulative lost_: Similar to the fraction of lost packets, the cumulative loss can be inferred from QUIC’s acknowledgments including all packets up to the latest acknowledged packet.

  - _Highest Sequence Number received_: The highest sequence number received is the sequence number of all RTP packets that were acknowledged.

  - Interarrival jitter: If QUIC acknowledgments carry timestamps as described in one of the extensions referenced above, senders can infer from QUIC acks the interarrival jitter from the arrival timestamps.

  - Last SR: Similar to RTP arrival times, the arrival time of RTCP Sender Reports can be inferred from QUIC acknowledgments, if they include timestamps.

  - Delay since last SR: This field is not required when the receiver reports are entirely replaced by QUIC feedback.

* **Negative Acknowledgments** (PT=205, FMT=1, Name=Generic NACK, [RFC4585])

  - The generic negative acknowledgment packet contains information about packets which the receiver considered lost. Section 6.2.1. of [RFC4585] recommends to use this feature only, if the underlying protocol cannot provide similar feedback. QUIC does not provide negative acknowledgments, but can detect lost packets through acknowledgments.

* **ECN Feedback** (PT=205, FMT=8, Name=RTCP-ECN-FB, [RFC6679])
- ECN feedback packets report the count of observed ECN-CE marks. [RFC6679] defines two RTCP reports, one packet type (with PT=205 and FMT=8) and a new report block for the extended reports which are listed below. QUIC supports ECN reporting through acknowledgments. If the connection supports ECN, the reporting of ECN counts SHOULD be done using QUIC acknowledgments.

* _Congestion Control Feedback_ (PT=205, FMT=11, Name=CCFB, [RFC8888])

- RTP Congestion Control Feedback contains acknowledgments, arrival timestamps and ECN notifications for each received packet. Acknowledgments and ECNs can be inferred from QUIC as described above. Arrival timestamps can be added through extended acknowledgment frames as described in [I-D.draft-smith-quic-receive-ts] or [I-D.draft-huitema-quic-ts].

* _Extended Reports_ (PT=207, Name=XR, [RFC3611])

- Extended Reports offer an extensible framework for a variety of different control messages. Some of the standard report blocks which can be implemented in extended reports such as loss RLE or ECNs can be implemented in QUIC, too. For other report blocks, it SHOULD be evaluated individually, if the contained information can be transmitted using QUIC instead.

5.2. Application Layer Repair and other Control Messages

While the previous section presented some RTCP packet that can be replaced by QUIC features, QUIC cannot replace all of the available RTCP packet types. This mostly affects RTCP packet types which carry control information that is to be interpreted by the application layer instead of the transport itself.

_sender Reports_ (PT=200, Name=SR, [RFC3550]) are similar to _receiver Reports_. They are send by media senders and additionally contain a NTP and a RTP timestamp and the number of packets and octets transmitted by the sender. The timestamps can be used by a receiver to synchronize streams. QUIC cannot provide a similar control information, since it does not know about RTP timestamps. A QUIC receiver can also not calculate the packet or octet counts, since it does not know about lost datagrams. Thus, sender reports are required in RTP over QUIC to synchronize streams at the receiver. The sender reports SHOULD not contain any receiver report blocks, as the information can be inferred from the QUIC transport as explained in the previous section.
Next to carrying transmission statistics, RTCP packets can contain application layer control information, that cannot directly be mapped to QUIC. This includes for example the _Source Description_ (PT=202, Name=SDES), _Bye_ (PT=203, Name=BYE) and _Application_ (PT=204, Name=APP) packet types from [RFC3550] or many of the payload specific feedback messages (PT=206) defined in [RFC4585], which can for example be used to control the codec behavior of the sender. Since QUIC does not provide any kind of application layer control messaging, these RTCP packet types SHOULD be used in the same way as they would be used over any other transport protocol.

6. Congestion Control

Like any other application on the internet, RTP over QUIC needs to perform congestion control to avoid overloading the network.

QUIC is a congestion controlled transport protocol. Senders are required to employ some form of congestion control. The default congestion control specified for QUIC is an algorithm similar to TCP NewReno, but senders are free to choose any congestion control algorithm as long as they follow the guidelines specified in Section 3 of [RFC8085].

RTP does not specify a congestion controller, but provides feedback formats for congestion control (e.g. [RFC8888]) as well as different congestion control algorithms in separate RFCs (e.g. [RFC8298] and [RFC8698]). The congestion control algorithms for RTP are specifically tailored for real-time transmissions at low latencies. The available congestion control algorithms for RTP expose a target_bitrate that can be used to dynamically reconfigure media codecs to produce media at a rate that can be sent in real-time under the observed network conditions.

This section defines two architectures for congestion control and bandwidth estimation for RTP over QUIC, but it does not mandate any specific congestion control algorithm to use.

It is assumed that the congestion controller in use provides a pacing mechanism to determine when a packet can be sent to avoid bursts. The currently proposed congestion control algorithms for real-time communications provide such pacing mechanisms. The use of congestion controllers which don’t provide a pacing mechanism is out of scope of this document.

*TODO:* Add considerations for bandwidth shares when a QUIC connection is shared between RTP and non-RTP streams?
6.1. Congestion Control at the QUIC layer

If congestion control is to be applied at the transport layer, it is RECOMMENDED to configure the QUIC Implementation to use a delay-based real-time congestion control algorithm instead of a loss-based algorithm. The currently available delay-based congestion control algorithms depend on detailed arrival time feedback to estimate the current one-way delay between sender and receiver. Since QUIC does not provide arrival timestamps in its acknowledgments, the QUIC implementations of the sender and receiver MUST use an extension to add this information to QUICs acknowledgment frames, e.g. [I-D.draft-smith-quic-receive-ts].

If congestion control is done by the QUIC implementation, the application needs a mechanism to query the currently available bandwidth to adapt media codec configurations. The employed congestion controller of the QUIC connection SHOULD expose such an API to the application. If a current bandwidth estimation is not available from the QUIC congestion controller, the sender can either implement an alternative bandwidth estimation at the application layer as described in Section 6.2 or a receiver can feedback the observed bandwidth through RTCP, e.g., using [I-D.draft-alvestrand-rmcat-remb].

*Editor's note:* An alternative to the hard requirement to use a timestamp extension could be to use RTCP, but that would mean, that an application has to negotiate RTCP congestion control feedback which would then have to be passed to the QUIC congestion controller.

*Editor’s note:* How can a QUIC connection be shared with non-RTP streams, when SCReAM/NADA/GCC is used as congestion controller? Can these algorithms be adapted to allow different streams including non-real-time streams? Do they even have to be adapted or _should_ this just work?

6.2. Congestion Control at the Application Layer

If an application cannot access a bandwidth estimation from the QUIC layer, or the QUIC implementation does not support a delay-based, low-latency congestion control algorithm, it can alternatively implement a bandwidth estimation algorithm at the application layer. Calculating a bandwidth estimation at the application layer can be done using the same bandwidth estimation algorithms as described in Section 6.1 (NADA, SCReAM). The bandwidth estimation algorithm typically needs some feedback on the transmission performance. This feedback can be collected following the guidelines in Section 5.
If the application implements full congestion control rather than just a bandwidth estimation at the application layer using a congestion controller that satisfies the requirements of Section 7 of [RFC9002], and the connection is only used to send real-time media which is subject to the application layer congestion control, it is RECOMMENDED to disable any other congestion control that is possibly running at the QUIC layer. Disabling the additional congestion controllers helps to avoid any interference between the different congestion controllers.

7. SDP Signalling

*Editor’s note:* See also [I-D.draft-dawkins-avtcore-sdp-rtp-quic].

QUIC is a connection-based protocol that supports connectionless transmissions of DATAGRAM frames within an established connection. As noted above, demultiplexing DATAGRAMS intended for different purposes is up to the application using QUIC.

There are several necessary steps to carry out jointly between the communicating peers to enable RTP over QUIC:

1. The protocol identifier for the m= lines MUST be "QUIC/RTP", combined as per [RFC8866] with the respective audiovisual profile: for example, "QUIC/RTP/AVP".

2. The peers need to decide whether to establish a new QUIC connection or whether to re-use an existing one. In case of establishing a new connection, the initiator and the responder (client and server) need to be determined. Signaling for this step MUST follow [RFC8122] on SDP attributes for connection-oriented media for the a=setup, a=connection, and a=fingerprint attributes. They MUST use the appropriate protocol identification as per 1.

3. The peers must provide a means for identifying RTP sessions carried in QUIC DATAGRAMS. To enable using a common transport connection for one, two, or more media sessions in the first place, the BUNDLE grouping framework MUST be used [RFC8843]. All media sections belonging to a bundle group, except the first one, MUST set the port in the m= line to zero and MUST include the a=bundle-only attribute.

For disambiguating different RTP session, a reference needs to be provided for each m= line to allow associating this specific media session with a flow identifier. This could be achieved following different approaches:
* Simply reusing the a=extmap attribute [RFC8285] and relying on RTP header extensions for demultiplexing different media packets carried in QUIC DATAGRAM frames.

* Defining a variant or different flavor of the a=extmap attribute [RFC8285] that binds media sessions to flow identifiers used in QUIC DATAGRAMS.

  *Editor’s note:* It is likely preferable to use multiplexing using QUIC DATAGRAM flow identifiers because this multiplexing mechanisms will also work across RTP and non-RTP media streams.

In either case, the corresponding identifiers MUST be treated independently for each direction of transmission, so that an endpoint MAY choose its own identifies and only uses SDP to inform its peer which RTP sessions use which identifiers.

To this end, SDP MUST be used to indicate the respective flow identifiers for RTP and RTCP of the different RTP sessions (for which we borrow inspiration from [RFC3605]).

4. The peers MUST agree, for each RTP session, whether or not to apply RTP/RTCP multiplexing. If multiplexing RTP and RTCP shall take place on the same flow identifier, this MUST be indicated using the attribute a=rtcp-mux.

A sample session setup offer (liberally borrowed and extended from [RFC8843] and [RFC8122] could look as follows:
v=0
o=alice 2890844526 2890844526 IN IP6 2001:db8::3
s=
c=IN IP6 2001:db8::3
t=0 0
a=group:BUNDLE abc xyz

m=audio 10000 QUIC/RTP/AVP 0 8 97
a=setup:actpass
a=connection:new
b=AS:200
a=mid:abc
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 iLBC/8000
a=extmap:1 urn:ietf:params:<tbd>

m=video 0 QUIC/RTP/AVP 31 32
b=AS:1000
a=bundle-only
a=mid:bar
a=rtpmap:31 H261/90000
a=rtpmap:32 MPV/90000
a=extmap:2 urn:ietf:params:<tbd>

---

Figure 2: SDP Offer

Signaling details to be worked out.

8. Discussion

8.1. Flow Identifier

[RFC9221] suggests to use flow identifiers to multiplex different streams on QUIC Datagrams, which is implemented in Section 4, but it is unclear how applications can combine RTP over QUIC with other data streams using the same QUIC connections. If the non-RTP data streams use the same flow identifies, too and the application can make sure, that flow identifiers are unique, there should be no problem. Flow identifiers could be problematic, if different specifications for RTP and non-RTP data streams over QUIC mandate different incompatible flow identifiers.
8.2. Impact of Connection Migration

9. Security Considerations

TBD

10. IANA Considerations

This document has no IANA actions.

11. References

11.1. Normative References

[I-D.draft-alvestrand-rmcat-remb]

[I-D.draft-huitema-quic-ts]

[I-D.draft-ietf-masque-h3-datagram]

[I-D.draft-ietf-quic-ack-frequency]

[I-D.draft-smith-quic-receive-ts]


Internet-Draft                RTP over QUIC                     May 2022

[RF8122] Lennox, J. and C. Holmberg, "Connection-Oriented Media
Transport over the Transport Layer Security (TLS) Protocol
in the Session Description Protocol (SDP)", RFC 8122,
DOI 10.17487/RFC8122, March 2017,

[RF8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,

Mechanism for RTP Header Extensions", RFC 8285,
DOI 10.17487/RFC8285, October 2017,

[RF8298] Johansson, I. and Z. Sarker, "Self-Clocked Rate Adaptation
for Multimedia", RFC 8298, DOI 10.17487/RFC8298, December

[RF8698] Zhu, X., Pan, R., Ramalho, M., and S. Mena, "Network-
Assisted Dynamic Adaptation (NADA): A Unified Congestion
Control Scheme for Real-Time Media", RFC 8698,
DOI 10.17487/RFC8698, February 2020,

[RF8843] Holmberg, C., Alvestrand, H., and C. Jennings,
"Negotiating Media Multiplexing Using the Session
Description Protocol (SDP)", RFC 8843,
DOI 10.17487/RFC8843, January 2021,

[RF8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP:
Session Description Protocol", RFC 8866,
DOI 10.17487/RFC8866, January 2021,

[RF9000] Sarker, Z., Perkins, C., Singh, V., and M. Ramalho, "RTP
Control Protocol (RTCP) Feedback for Congestion Control",
RFC 9000, DOI 10.17487/RFC9000, May 2021,
11.2. Informative References

[I-D.draft-dawkins-avtcore-sdp-rtp-quic]

[I-D.draft-hurst-quic-rtp-tunnelling]

Acknowledgments

TODO acknowledge.

Authors’ Addresses

Jörg Ott
Technical University Munich
Email: ott@in.tum.de

Mathis Engelbart
Technical University Munich
Email: mathis.engelbart@gmail.com
RTP Payload Format for the SCIP Codec
draft-hanson-avtcore-rtp-scip-00.txt

Abstract

This document describes the RTP payload format of the Secure Communication Interoperability Protocol (SCIP) as audio and video media subtypes. It provides RFC 6838 compliant media
The IANA registration of media subtype types in the IETF tree created two similar media subtypes "scip" under the audio and video media types [AUDIOSCIP], [VIDEOSCIP]. This document, as the common top-level reference, provides information on their similarities and differences and the usage of those media subtypes.

This document details usage of the scip pseudo-codec as a secure session establishment protocol and transport protocol over RTP. It provides a reference for network security policymakers, network equipment OEMs, procurement personnel, and government agency and commercial industry representatives.

1.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and
"OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Best current practices for writing an RTP payload format specification were followed [RFC2736] [RFC8088].

1.2. Abbreviations

The following abbreviations are used in this document.

- AVP: Audio/Video Profile
- DTX: Discontinuous Transmission
- FNBDT: Future Narrowband Digital Terminal
- ICWG: Interoperability Control Working Group
- IICWG: International Interoperability Control Working Group
- NATO: North Atlantic Treaty Organization
- SCIP: Secure Communication Interoperability Protocol
- SDP: Session Description Protocol

2. Background

The Secure Communication Interoperability Protocol (SCIP) allows the negotiation of several voice, data, and video applications using various encryption suites. SCIP also provides several important characteristics that have led to its broad acceptance in the international user community. These features include end-to-end security at the application layer, authentication of user identity, the ability to apply different security levels for each secure session, and secure communication over any end-to-end data connection.

SCIP began in the U.S. as the FNBDT (Future Narrowband Digital Terminal) Protocol. A combined Department of Defense and vendor consortium formed a governing organization named the ICWG (Interoperability Control Working Group). In time, the group expanded to include NATO, NATO partners and European vendors under the name IICWG (International Interoperability Control Working Group), which was later renamed the SCIP Working Group.

SCIP is presently implemented in U.S. and NATO secure voice, video, and data products operating on commercial, private, and tactical IP networks worldwide using the scip media subtype. First generation SCIP devices operated on circuit-switched
networks. SCIP was then expanded to radio and IP networks. The SCIP media subtype transports SCIP secure session establishment signaling and secure application traffic. The built-in negotiation and flexibility provided by the SCIP standards make it a natural choice for many scenarios that require various secure applications and associated encryption suites. SCIP has been endorsed by many nations as the secure end-to-end solution for secure voice, video, and data devices. SCIP standards are currently available to participating government/military communities and select OEMs of equipment that support SCIP.

However, SCIP must operate over global networks (including private and commercial networks). Without access to necessary information to support SCIP, some networks may not support the SCIP media subtypes. Issues may occur simply because information is not as readily available to OEMs, network administrators, and network architects.

This RFC provides essential information about audio/scip and video/scip media subtypes that enables network equipment manufacturers to include SCIP as a known audio and video media subtype in their equipment and enables network administrators to define and implement a compatible security policy.

All current IP-based SCIP devices support "scip" as a media subtype. Registration of SCIP as a media subtype provides a common reference for network equipment manufacturers to recognize SCIP in a payload declaration.

3. Media Format Description

The "scip" media subtype indicates support for and identifies SCIP traffic that is being transferred using RTP. SCIP traffic requires end-to-end bit integrity, therefore transcoding SHALL NOT be performed over the end-to-end IP connection. The audio/scip and video/scip media subtype data streams within the network, including the VoIP network, MUST be a transparent relay and be treated as "clear-channel data", similar to the Clearmode media subtype defined by RFC 4040. However, Clearmode is defined as a gateway protocol and limited to a sample rate of 8000 Hz and 64kbps bandwidth only [RFC4040]. Clearmode is not defined for the higher sample and data rates required for some SCIP traffic.
4. Payload Format

The RTP Packet content of SCIP traffic is dependent upon the SCIP session state. SCIP secure session establishment uses protocols defined in SCIP-210 [SCIP210] to negotiate an application. SCIP secure traffic may consist of the encrypted output of codecs such as MELPe [RFC8130], G.729D [RFC3551], H.264 [RFC6184], or other media encodings, based on the application negotiated during SCIP secure session establishment. SCIP traffic is highly variable and may include other SCIP signaling information in the media stream. SCIP traffic may not always be a continuous stream at the bit rate specified in the SDP [RFC8866] since discontinuous transmission (DTX) or other mechanisms may be used. The SCIP payload size will vary, especially during SCIP secure session establishment.

4.1. RTP Header Fields

The RTP header fields SHOULD conform to RFC 3550. This is a SHOULD rather than a SHALL in recognition that legacy SCIP-enabled products may not strictly adhere to RFC 3550.

SCIP traffic may be continuous or discontinuous. The Timestamp field increments based on the sampling clock for discontinuous transmission as described in [RFC3550], Section 5.1. The Timestamp field for continuous transmission applications is dependent on the sampling rate of the media as specified in the media subtype’s specification (e.g., MELPe [RFC8130]). Note that during a call, both discontinuous and continuous traffic is highly probable. Therefore, a jitter buffer MAY be implemented in endpoint devices only but SHOULD NOT be implemented in network devices.

The Marker bit SHOULD be set to zero for discontinuous traffic. The Marker bit for continuous traffic is based on the underlying media subtype specification. This specification is a SHOULD rather than a SHALL in recognition that legacy SCIP-enabled products may not strictly adhere to the media subtype specification.

5. Payload Format Parameters

The SCIP RTP payload format is identified using the scip media subtype, which is registered in accordance with [RFC4855] and per the media type registration template form [RFC6838]. A clock rate of 8000 Hz SHALL be used for "audio/scip". A clock rate of 90000 Hz SHALL be used for "video/scip".
5.1. Media Subtype "audio/scip"

Media type name: audio
Media subtype name: scip
Required parameters: N/A
Optional parameters: N/A

Encoding considerations: Binary. This media subtype is only defined for transfer via RTP. There SHALL be no encoding/decoding (transcoding) of the audio stream as it traverses the network.


Interoperability considerations: N/A

Published specifications: [SCIP214], [SCIP210]

Applications which use this media: N/A

Fragment Identifier considerations: none

Restrictions on usage: N/A

Additional information:

1. Deprecated alias names for this type: N/A
2. Magic number(s): N/A
3. File extension(s): N/A
4. Macintosh file type code: N/A
5. Object Identifiers: N/A

Person to contact for further information:

1. Name: Michael Faller and Daniel Hanson
2. Email: michael.faller@gd-ms.com and dan.hanson@gd-ms.com

Intended usage: Common, Government and Military
5.2. Media Subtype "video/scip"

Media type name: video
Media subtype name: scip
Required parameters: N/A
Optional parameters: N/A

Encoding considerations: Binary. This media subtype is only defined for transfer via RTP. There SHALL be no encoding/decoding (transcoding) of the video stream as it traverses the network.


Interoperability considerations: N/A

Published specifications: [SCIP214], [SCIP210]

Applications which use this media: N/A

Fragment Identifier considerations: none

Restrictions on usage: N/A

Additional information:

1. Deprecated alias names for this type: N/A
2. Magic number(s): N/A
3. File extension(s): N/A
4. Macintosh file type code: N/A
5. Object Identifiers: N/A

Person to contact for further information:

1. Name: Michael Faller and Daniel Hanson
2. Email: michael.faller@gd-ms.com and dan.hanson@gd-ms.com

Intended usage: Common, Government and Military

Authors:

Michael Faller - michael.faller@gd-ms.com

Daniel Hanson - dan.hanson@gd-ms.com

Change controller:

SCIP Working Group - ncia.cis3@ncia.nato.int

5.3. Mapping to SDP

The mapping of the above defined payload format media subtype and its parameters SHALL be done according to Section 3 of [RFC4855].

An example mapping for audio/scip is:

m=audio 50000 RTP/AVP 96
a=rtpmap:96 scip/8000

An example mapping for video/scip is:

m=video 50002 RTP/AVP 97
a=rtpmap:97 scip/90000

An example mapping for both audio/scip and video/scip is:

m=audio 50000 RTP/AVP 96
a=rtpmap:96 scip/8000
m=video 50002 RTP/AVP 97
a=rtpmap:97 scip/90000

The application negotiation between endpoints will determine whether the audio and video streams are transported as separate
5.4. SDP Offer/Answer Considerations

In accordance with the SDP Offer/Answer model [RFC3264], the SCIP device SHALL list the SCIP payload type in order of preference in the "m" media line.

6. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVPF [RFC3711], or RTP/SAVPF [RFC5124]. However, as "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format’s responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity, and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this Security Considerations section discusses the security impacting properties of the payload format itself.

This RTP payload format and its media decoder do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing, and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Nor does the RTP payload format contain any active content.

7. IANA Considerations

The audio/scip and video/scip media subtypes have been registered with IANA [AUDIOSCIP] [VIDEOSCIP].

8. References

8.1. Normative References

[AUDIOSCIP] Faller, M., and D. Hanson, "audio/scip", Internet Assigned Numbers Authority (IANA), 28 January 2021,
8.2. Informative References

[SCIP210] SCIP-210, "SCIP Signaling Plan", Revision 3.10, 26 October 2017, request access via email <ncia.cis3@ncia.nato.int>.


9. Authors’ Addresses

Daniel Hanson
General Dynamics Mission Systems, Inc.
150 Rustcraft Road
Dedham, MA 02026, USA
E-mail: dan.hanson@gd-ms.com

Michael Faller
General Dynamics Mission Systems, Inc.
150 Rustcraft Road
Dedham, MA 02026, USA
E-mail: michael.faller@gd-ms.com

Keith Maver
General Dynamics Mission Systems, Inc.
150 Rustcraft Road
Dedham, MA 02026, USA
E-mail: keith.maver@gd-ms.com

SCIP Working Group, CIS3 Partnership
NATO Communications and Information Agency
Oude Waalsdorperweg 61, 2597AK
The Hague, The Netherlands
E-mail: ncia.cis3@ncia.nato.int
Abstract

While the Secure Real-time Transport Protocol (SRTP) provides confidentiality for the contents of a media packet, a significant amount of metadata is left unprotected, including RTP header extensions and contributing sources (CSRCs). However, this data can be moderately sensitive in many applications. While there have been previous attempts to protect this data, they have had limited deployment, due to complexity as well as technical limitations.

This document defines Cryptex as a new mechanism that completely encrypts header extensions and CSRCs and uses simpler signaling with the goal of facilitating deployment.

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 8 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.
This document is subject to BCP 78 and the IETF Trust’s Legal
license-info) in effect on the date of publication of this document.
Please review these documents carefully, as they describe your rights
and restrictions with respect to this document. Code Components
extracted from this document must include Revised BSD License text as
described in Section 4.e of the Trust Legal Provisions and are
provided without warranty as described in the Revised BSD License.

Table of Contents

1.  Introduction ...................................................... 3
  1.1.  Problem Statement .......................................... 3
  1.2.  Previous Solutions .......................................... 3
  1.3.  Goals ......................................................... 4
2.  Terminology ...................................................... 5
3.  Design .......................................................... 5
4.  Signaling ......................................................... 5
5.  RTP Header Processing ........................................... 6
  5.1.  Sending ....................................................... 6
  5.2.  Receiving ..................................................... 7
6.  Encryption and Decryption ........................................ 7
  6.1.  Packet Structure ............................................ 7
  6.2.  Encryption Procedure ....................................... 8
  6.3.  Decryption Procedure ....................................... 9
7.  Backwards Compatibility ......................................... 9
8.  Security Considerations ......................................... 9
9.  IANA Considerations ............................................. 10
  9.1.  cryptex SDP Attribute ....................................... 10
10. Acknowledgements ................................................ 11
11. References ....................................................... 11
  11.1.  Normative References ..................................... 11
  11.2.  Informative References .................................... 12
Appendix A.  Test Vectors ........................................... 12
  A.1.  AES-CTR ...................................................... 12
    A.1.1.  RTP Packet with 1-byte header extension .............. 12
    A.1.2.  RTP Packet with 2-byte header extension ............ 13
    A.1.3.  RTP Packet with 1-byte header extension and CSRC
            fields ............................................... 14
    A.1.4.  RTP Packet with 2-byte header extension and CSRC
            fields ............................................... 14
    A.1.5.  RTP Packet with empty 1-byte header extension and CSRC
            fields ............................................... 15
    A.1.6.  RTP Packet with empty 2-byte header extension and CSRC
            fields ............................................... 15
  A.2.  AES-GCM ..................................................... 17
    A.2.1.  RTP Packet with 1-byte header extension ............ 17
    A.2.2.  RTP Packet with 2-byte header extension ............ 17
1.  Introduction

1.1.  Problem Statement

The Secure Real-time Transport Protocol [RFC3711] mechanism provides message authentication for the entire RTP packet, but only encrypts the RTP payload. This has not historically been a problem, as much of the information carried in the header has minimal sensitivity (e.g., RTP timestamp); in addition, certain fields need to remain as cleartext because they are used for key scheduling (e.g., RTP SSRC and sequence number).

However, as noted in [RFC6904], the security requirements can be different for information carried in RTP header extensions, including the per-packet sound levels defined in [RFC6464] and [RFC6465], which are specifically noted as being sensitive in the Security Considerations section of those RFCs.

In addition to the contents of the header extensions, there are now enough header extensions in active use that the header extension identifiers themselves can provide meaningful information in terms of determining the identity of the endpoint and/or application. Accordingly, these identifiers can be considered a fingerprinting issue.

Finally, the CSRCs included in RTP packets can also be sensitive, potentially allowing a network eavesdropper to determine who was speaking and when during an otherwise secure conference call.

1.2.  Previous Solutions

[RFC6904] was proposed in 2013 as a solution to the problem of unprotected header extension values. However, it has not seen significant adoption, and has a few technical shortcomings.

First, the mechanism is complicated. Since it allows encryption to be negotiated on a per-extension basis, a fair amount of signaling logic is required. And in the SRTP layer, a somewhat complex
transform is required to allow only the selected header extension values to be encrypted. One of the most popular SRTP implementations had a significant bug in this area that was not detected for five years.

Second, it only protects the header extension values, and not their ids or lengths. It also does not protect the CSRCs. As noted above, this leaves a fair amount of potentially sensitive information exposed.

Third, it bloats the header extension space. Because each extension must be offered in both unencrypted and encrypted forms, twice as many header extensions must be offered, which will in many cases push implementations past the 14-extension limit for the use of one-byte extension headers defined in [RFC8285]. Accordingly, implementations will need to use two-byte headers in many cases, which are not supported well by some existing implementations.

Finally, the header extension bloat combined with the need for backwards compatibility results in additional wire overhead. Because two-byte extension headers may not be handled well by existing implementations, one-byte extension identifiers will need to be used for the unencrypted (backwards compatible) forms, and two-byte for the encrypted forms. Thus, deployment of [RFC6904] encryption for header extensions will typically result in multiple extra bytes in each RTP packet, compared to the present situation.

1.3. Goals

From this analysis we can state the desired properties of a solution:

* Build on existing [RFC3711] SRTP framework (simple to understand)
* Build on existing [RFC8285] header extension framework (simple to implement)
* Protection of header extension ids, lengths, and values
* Protection of CSRCs when present
* Simple signaling
* Simple crypto transform and SRTP interactions
* Backward compatible with unencrypted endpoints, if desired
* Backward compatible with existing RTP tooling
The last point deserves further discussion. While we considered possible solutions that would have encrypted more of the RTP header (e.g., the number of CSRCs), we felt the inability to parse the resultant packets with current tools, as well as additional complexity incurred, outweighed the slight improvement in confidentiality.

2. Terminology

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 [BCP14] RFC2119 RFC8174 when, and only when, they appear in all capitals, as shown here.

3. Design

This specification proposes a mechanism to negotiate encryption of all RTP header extensions (ids, lengths, and values) as well as CSRC values. It reuses the existing SRTP framework, is accordingly simple to implement, and is backward compatible with existing RTP packet parsing code, even when support for the mechanism has been negotiated.

4. Signaling

In order to determine whether the mechanism defined in this specification is supported, this document defines a new "a=cryptex" Session Description Protocol (SDP) [RFC4566] attribute to indicate support.

This attribute takes no value, and can be used at the session level or media level.

The presence of this attribute in the SDP (either in an offer or answer) indicates that the endpoint is capable of receiving RTP packets encrypted with Cryptex, as defined below.

Once each peer has verified that the other party supports receiving RTP packets encrypted with Cryptex, senders can unilaterally decide whether to use the Cryptex mechanism or not.

If BUNDLE is in use and the a=cryptex attribute is present for a media line, it MUST be present for all media lines belonging to the same bundle group. This ensures that the encrypted MID header extensions used to demux BUNDLE can be processed correctly. When used with BUNDLE, this attribute is assigned to the TRANSPORT category [RFC8859].
It is possible to signal and negotiate both Encryption of Header Extensions as defined in [RFC6904] and cryptex in the SDP O/A, however if a packet is encrypted with cryptex, it MUST NOT use the [RFC6904] header extension encryption mechanisms.

5. RTP Header Processing

[RFC8285] defines two values for the "defined by profile" field for carrying one-byte and two-byte header extensions. In order to allow a receiver to determine if an incoming RTP packet is using the encryption scheme in this specification, two new values are defined:

* 0xC0DE for the encrypted version of the one-byte header extensions (instead of 0xBEDE).

* 0xC2DE for the encrypted versions of the two-byte header extensions (instead of 0x100).

In the case of using two-byte header extensions, the extension id with value 256 MUST NOT be negotiated, as the value of this id is meant to be contained in the "appbits" of the "defined by profile" field, which are not available when using the values above.

If the "a=extmap-allow-mixed" attribute defined in [RFC8285] is negotiated, either one-byte or two-byte header ids can be used (with the values above), as in [RFC8285].

5.1. Sending

When the mechanism defined by this specification has been negotiated, sending a RTP packet that has any CSRCs or contains any [RFC8285] header extensions follows the steps below. This mechanism MUST NOT be used with header extensions other than the [RFC8285] variety.

If the packet contains solely one-byte extension ids, the 16-bit RTP header extension tag MUST be set to 0xC0DE to indicate that the encryption has been applied, and the one-byte framing is being used. If the packet contains only two-byte extension ids, the header extension tag MUST be set to 0xC2DE to indicate encryption has been applied, and the two-byte framing is being used.

If the packet contains CSRCs but no header extensions, an empty extension block consisting of the 0xC0DE tag and a 16-bit length field set to zero (explicitly permitted by [RFC3550]) MUST be appended, and the X bit MUST be set to 1 to indicate an extension block is present. This is necessary to provide the receiver an indication that the CSRCs in the packet are encrypted.
The RTP packet MUST then be encrypted as described in Encryption Procedure.

5.2. Receiving

When receiving an RTP packet that contains header extensions, the "defined by profile" field MUST be checked to ensure the payload is formatted according to this specification. If the field does not match one of the values defined above, the implementation MUST instead handle it according to the specification that defines that value. The implementation MAY stop and report an error if it considers use of this specification mandatory for the RTP stream.

If the RTP packet passes this check, it is then decrypted according to Decryption Procedure, and passed to the the next layer to process the packet and its extensions. In the event that a zero-length extension block was added as indicated above, it can be left as-is and will be processed normally.

6. Encryption and Decryption

6.1. Packet Structure

When this mechanism is active, the SRTP packet is protected as follows:
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**6.2. Encryption Procedure**

The encryption procedure is identical to that of [RFC3711] except for the region to encrypt, which is as shown in the section above.

To minimize changes to surrounding code, the encryption mechanism can choose to replace a "defined by profile" field from [RFC8285] with its counterpart defined in RTP Header Processing above and encrypt at the same time.

For AEAD ciphers (e.g., GCM), the 12-byte fixed header and the four-byte header extension header (the "defined by profile" field and the length) are considered AAD, even though they are non-contiguous in the packet if CSRCs are present.

---

Specifically, the encrypted portion MUST include any CSRC identifiers, any RTP header extension (except for the first 4 bytes), and the RTP payload.
6.3. Decryption Procedure

The decryption procedure is identical to that of [RFC3711] except for the region to decrypt, which is as shown in the section above.

To minimize changes to surrounding code, the decryption mechanism can choose to replace the "defined by profile" field with its no-encryption counterpart from [RFC8285] and decrypt at the same time.

7. Backwards Compatibility

This specification attempts to encrypt as much as possible without interfering with backwards compatibility for systems that expect a certain structure from an RTPv2 packet, including systems that perform demultiplexing based on packet headers. Accordingly, the first two bytes of the RTP packet are not encrypted.

This specification also attempts to reuse the key scheduling from SRTP, which depends on the RTP packet sequence number and SSRC identifier. Accordingly these values are also not encrypted.

8. Security Considerations

This specification extends SRTP by expanding the portion of the packet that is encrypted, as shown in Packet Structure. It does not change how SRTP authentication works in any way. Given that more of the packet is being encrypted than before, this is necessarily an improvement.

The RTP fields that are left unencrypted (see rationale above) are as follows:

* RTP version
* padding bit
* extension bit
* number of CSRCs
* marker bit
* payload type
* sequence number
* timestamp
6.  SRTP header

   * SSRC identifier
   * number of [RFC8285] header extensions

These values contain a fixed set (i.e., one that won’t be changed by extensions) of information that, at present, is observed to have low sensitivity. In the event any of these values need to be encrypted, SRTP is likely the wrong protocol to use and a fully-encapsulating protocol such as DTLS is preferred (with its attendant per-packet overhead).

9.  IANA Considerations

9.1.  cryptex SDP Attribute

This document updates the "Session Description Protocol Parameters" registry as specified in Section 8.2.4 of [RFC8866]. Specifically, it adds the SDP ‘cryptex’ attribute to the table for SDP media-level attributes.

Contact name: TBD
Contact email address: TBD
Attribute name: cryptex
Attribute syntax: This attribute takes no values.
Attribute semantics: N/A
Attribute value: N/A
Usage level: media-level
Charset dependent: No
Purpose: The presence of this attribute in the SDP indicates that the endpoint is capable of receiving RTP packets encrypted with Cryptex as described in this document. O/A procedures: SDP O/A procedures are described in Section 4 of this document.

Mux Category: TRANSPORT
10. Acknowledgements

The authors wish to thank Lennart Grahl for pointing out many of the issues with the existing header encryption mechanism, as well as suggestions for this proposal. Thanks also to Jonathan Lennox, Inaki Castillo, and Bernard Aboba for their review and suggestions.

11. References

11.1. Normative References


<https://www.rfc-editor.org/info/bcp14>


11.2. Informative References


Appendix A. Test Vectors

All values are in hexadecimal and represented in network order (big endian).

A.1. AES-CTR

Common values are organized as follows:

- Rollover Counter: 00000000
- Master Key: elf97a0d3e018be0d64fa32c06de4139
- Master Salt: 0ec675ad498afeeb6960b3aabe6
- Crypto Suite: AES_CM_128_HMAC_SHA1_80
- Session Key: c61e7a93744f39ee10734afe3ff7a087
- Session Salt: 30cbbc08863d8c85d49db34a9ae1
- Authentication Key: cebe321f6ff7716b6fd4ab49af256a156d38baa4

A.1.1. RTP Packet with 1-byte header extension

RTP Packet:
Encrypted RTP Packet:

```
900f1235
decafbad
cafebabe
bede0001
51000200
abababab
abababab
abababab
abababab
```

A.1.2. RTP Packet with 2-byte header extension

RTP Packet:

```
900f1236
decafbad
cafebabe
c0de0001
eb923652
51c3e036
f8de27e9
c27ee3e0
b4651d9f
bc4218a7
0244522f
34a5
```

Encrypted RTP Packet:
A.1.3. RTP Packet with 1-byte header extension and CSRC fields

RTP Packet:

920f1238
decafbad
cafebabe
c0de0001
4ed9cc4e
6a712b30
96c5ca77
339d4204
ce0d7739
6cab6958
5fbce381
94a5

Encrypted RTP Packet:

920f1238
decafbad
cafebabe
8bb6e12b
5cff16dd
c0de0001
92838c8c
09e58393
e1de3a9a
74734d67
45671338
c3acff11d
a2df8423
bee0
A.1.4. RTP Packet with 2-byte header extension and CSRC fields

RTP Packet:

920f1239
decafbad
cafebabe
0001e240
0000b26e
10000001
05020002
abababab
abababab
abababab
abababab

Encrypted RTP Packet:

920f1239
decafbad
cafebabe
f70e513e
b90b9b25
c2de0001
bbed4848
faa64466
5f3d7f34
125914e9
f4d0ae92
3c6f479b
95a0f7b5
3133

A.1.5. RTP Packet with empty 1-byte header extension and CSRC fields

RTP Packet:

920f123a
decafbad
cafebabe
0001e240
0000b26e
bede0000
abababab
abababab
abababab
abababab
abababab
Encrypted RTP Packet:
920f123a
decafbad
cafebabe
7130b6ab
fe2ab0e3
c0de0000
e3d9f64b
25c9e74c
b4cf8e43
fb92e378
1c2c0cea
b6b3a499
a14c

A.1.6. RTP Packet with empty 2-byte header extension and CSRC fields

RTP Packet:
920f123b
decafbad
cafebabe
0001e240
0000b26e
10000000
abababab
abababab
abababab
abababab

Encrypted RTP Packet:
920f123b
decafbad
cafebabe
cbf24c12
4330e1c8
c2de0000
599dd45b
c9d687b6
03e8b59d
771fd38e
88b170e0
cd31e125
eabe
A.2. AES-GCM

Common values are organized as follows:

- Rollover Counter: 00000000
- Master Key: 00102030405060708090a0b0c0d0e0f
- Master Salt: a0a1a2a3a4a5a6a7a8a9aaab
- Crypto Suite: AEAD_AES_128_GCM
- Session Key: 077c6143cb221bc355ff23d5f984a16e
- Session Salt: 9af3e95364ebac9c99c5a7c4

A.2.1. RTP Packet with 1-byte header extension

RTP Packet:

900f1235
decafbad
cafebabe
bede0001
51000200
abababab
abababab
abababab
abababab

Encrypted RTP Packet:

900f1235
decafbad
cafebabe
c0de0001
39972dc9
572c4d99
e8fc355d
e743fb2e
94f94ff
54e72f41
93bcb5c7
4ffab0fa
9fa0fbeb

A.2.2. RTP Packet with 2-byte header extension

RTP Packet:
900f1236
decafbad
cafebabe
10000001
05020002
abababab
abababab
abababab
abababab

Encrypted RTP Packet:
900f1236
decafbad
cafebabe
c2de0001
bb75a4c5
45cd1f41
3bdb7daa
2b1e3263
de313667
c9632490
81b35a65
f5cb6c88
b394235f

A.2.3. RTP Packet with 1-byte header extension and CSRC fields

RTP Packet:
920f1238
decafbad
cafebabe
0001e240
0000b26e
bede0001
51000200
abababab
abababab
abababab
abababab

Encrypted RTP Packet:
A.2.4. RTP Packet with 2-byte header extension and CSRC fields

RTP Packet:

920f1239
decafbad
cafebabe
63bccc4
a7f695c4
c0de0001
8ad7c71f
ac70a80c
92866b4c
6ba98546
ef913586
e95ffaaf
fe956885
bb0647a8
bc094ac8

Encrypted RTP Packet:
A.2.5. RTP Packet with empty 1-byte header extension and CSRC fields

RTP Packet:

920f123a
decafbad
cafebabe
3680524f
8d312b00
c2de0001
c78d1200
38422bc1
11a7187a
18246f98
0c059cc6
bc9df8b6
26394eca
344e4b05
d80fe83

Encrypted RTP Packet:

920f123a
decafbad
cafebabe
15b6bb43
37906fff
c0de0000
b7b96453
7a2b03ab
7ba5389c
e9331712
6b5d974d
f30c6884
dcb651c5
e120c1da
A.2.6. RTP Packet with empty 2-byte header extension and CSRC fields

RTP Packet:

```
920f123b
decafbad
cafebabe
0001e240
0000b26e
10000000
abababab
abababab
abababab
abababab
```

Encrypted RTP Packet:

```
920f123b
decafbad
cafebabe
dcb38c9e
48bf95f4
c2de0000
61ee432c
f9203170
76613258
d3ce4236
06a429
681ad084
13512dc9
8b5207d8
```

Authors’ Addresses

Justin Uberti
Clubhouse
Email: justin@uberti.name

Cullen Jennings
Cisco
Email: fluffy@iii.ca

Sergio Garcia Murillo
CoSMo
Email: sergio.garcia.murillo@cosmosoftware.io
Multiplexing Scheme Updates for QUIC
draft-ietf-avtcore-rfc7983bis-04.txt

Abstract

This document defines how QUIC, Datagram Transport Layer Security (DTLS), Real-time Transport Protocol (RTP), RTP Control Protocol (RTCP), Session Traversal Utilities for NAT (STUN), Traversal Using Relays around NAT (TURN), and ZRTP packets are multiplexed on a single receiving socket.

This document updates RFC 7983 and RFC 5764.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 13, 2022.
Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ................................................. 3
   1.1. Terminology ........................................... 3
2. Multiplexing of TURN Channels ............................. 3
3. Updates to RFC 7983 ......................................... 4
4. Security Considerations .................................... 5
5. IANA Considerations ....................................... 6
6. References .................................................. 6
   6.1. Normative References ................................... 6
   6.2. Informative References ................................. 7
Acknowledgements .............................................. 8
Authors’ Addresses ............................................ 8
1. Introduction

"Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)"

[RFC7983] defines a scheme for a Real-time Transport Protocol (RTP) [RFC3550] receiver to demultiplex DTLS [RFC9147], Session Traversal Utilities for NAT (STUN) [RFC8489], Secure Real-time Transport Protocol (SRTP) / Secure Real-time Transport Control Protocol (SRTCP) [RFC3711], ZRTP [RFC6189] and TURN Channel packets arriving on a single port. This document updates [RFC7983] and [RFC5764] to also allow QUIC [RFC9000] to also be multiplexed on the same port.

The multiplexing scheme described in this document enables multiple usage scenarios. Peer-to-peer QUIC in WebRTC scenarios, described in [P2P-QUIC] [P2P-QUIC-TRIAL], uses RTP for transport of audio and video along with QUIC for data exchange. For this use case, SRTP [RFC3711] is keyed using DTLS-SRTP [RFC5764] and therefore SRTP/SRTCP [RFC3550], STUN, TURN, DTLS and QUIC need to be multiplexed on the same port. Were SRTP to be keyed using QUIC-SRTP, SRTP/SRTCP, STUN, TURN and QUIC would need to be multiplexed on the same port. Where QUIC is used for peer-to-peer transport of data as well as RTP [I-D.ietf-quic-over-rtp-03] STUN, TURN and QUIC need to be multiplexed on the same port.

The scheme described in this document is compatible with QUIC version 2 [I-D.ietf-quic-v2]. However, it is not compatible with QUIC Bit greasing, as defined in [I-D.ietf-quic-bit-grease]. Therefore, in situations where multiplexing is desired, QUIC Bit greasing MUST NOT be negotiated.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Multiplexing of TURN Channels

TURN channels are an optimization where data packets are exchanged with a 4-byte prefix instead of the standard 36-byte STUN overhead (see Section 3.5 of [RFC8656]). [RFC7983] allocated the values from 64 to 79 in order to allow TURN channels to be demultiplexed when the TURN Client does the channel binding request in combination with the demultiplexing scheme described in [RFC7983].

When QUIC Bit greasing is not negotiated, the first octet of a QUIC short header packet falls in the range 64 to 127, thereby overlapping with the allocated range for TURN channels of 64 to 79.
Where QUIC Bit greating is not negotiated, the first octet of QUIC long header packets fall in the range 192 to 255. Since QUIC long header packets precede QUIC short header packets, if no packets with a first octet in the range of 192 to 255 have been received, a packet whose first octet is in the range of 64 to 79 can be demultiplexed unambiguously as TURN Channel traffic. Since WebRTC implementations supporting QUIC data exchange do not utilize TURN Channels, once packets with a first octet in the range of 192 to 255 have been received, a packet whose first octet is in the range of 64 to 127 can be demultiplexed as QUIC traffic.

3. Updates to RFC 7983

This document updates the text in Section 7 of [RFC7983] (which in turn updates [RFC5764]) as follows:

OLD TEXT

The process for demultiplexing a packet is as follows. The receiver looks at the first byte of the packet. If the value of this byte is in between 0 and 3 (inclusive), then the packet is STUN. If the value is between 16 and 19 (inclusive), then the packet is ZRTP. If the value is between 20 and 63 (inclusive), then the packet is DTLS. If the value is between 64 and 79 (inclusive), then the packet is TURN Channel. If the value is in between 128 and 191 (inclusive), then the packet is RTP (or RTCP, if both RTCP and RTP are being multiplexed over the same destination port). If the value does not match any known range, then the packet MUST be dropped and an alert MAY be logged. This process is summarized in Figure 3.

```
+----------------+
|        [0..3] -+--> forward to STUN
|                |
|      [16..19] -+--> forward to ZRTP
|   packet -->   |      [20..63] -+--> forward to DTLS
|                |
|    [64..79] -+--> forward to TURN Channel
|                |
|    [128..191] -+--> forward to RTP/RTCP
+----------------+
```

Figure 3: The DTLS-SRTP receiver’s packet demultiplexing algorithm.

END OLD TEXT
NEW TEXT

The process for demultiplexing a packet is as follows. The receiver looks at the first byte of the packet. If the value of this byte is in between 0 and 3 (inclusive), then the packet is STUN. If the value is between 16 and 19 (inclusive), then the packet is ZRTP. If the value is between 20 and 63 (inclusive), then the packet is DTLS. If the value is in between 128 and 191 (inclusive) then the packet is RTP (or RTCP, if both RTCP and RTP are being multiplexed over the same destination port). If the value is between 80 and 127 or between 192 and 255 (inclusive) then the packet is QUIC. If the value is between 64 and 79 inclusive, then if a packet has been previously forwarded that is in the range of 192 and 255, then the packet is QUIC, otherwise it is TURN Channel.

If the value does not match any known range, then the packet MUST be dropped and an alert MAY be logged. This process is summarized in Figure 3.

Figure 3: The receiver’s packet demultiplexing algorithm.

Note: The demultiplexing of QUIC packets requires that QUIC Bit greasing [I-D.ietf-quic-bit-grease] not be negotiated.

END NEW TEXT

4. Security Considerations

The solution discussed in this document could potentially introduce some additional security considerations beyond those detailed in [RFC7983]. Due to the additional logic required, if mis-implemented, heuristics have the potential to mis-classify packets.

Aboba, et. al Standards Track [Page 5]
When QUIC is used only for data exchange, the TLS-within-QUIC exchange [RFC9001] derives keys used solely to protect the QUIC data packets. If properly implemented, this should not affect the transport of SRTP nor the derivation of SRTP keys via DTLS-SRTP. However, were the TLS-within-QUIC exchange to be used to derive SRTP keys, both transport and SRTP key derivation could be adversely impacted by a vulnerability in the QUIC implementation.

5. IANA Considerations

This document does not require actions by IANA.

6. References

6.1. Normative References


6.2. Informative References

[I-D.engelbart-rtp-over-quic]
Ott, J. and M. Engelbart, "RTP over QUIC", draft-engelbart-rtp-over-quic-02 (work in progress), March 7, 2022.

[I-D.ietf-quic-v2]

[RFC6189]

[P2P-QUIC]

[P2P-QUIC-TRIAL]
Hampson, S., "RTCQuicTransport Coming to an Origin Trial Near You (Chrome 73)", January 2019,
Acknowledgments

We would like to thank Martin Thomson, Roni Even and other participants in the IETF QUIC and AVTCORE working groups for their discussion of the QUIC multiplexing issue, and their input relating to potential solutions.

Authors’ Addresses

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA  98052
USA

Email: bernard.aboba@gmail.com

Gonzalo Salgueiro
Cisco Systems
7200-12 Kit Creek Road
Research Triangle Park, NC  27709
United States of America

Email: gsalguei@cisco.com

Colin Perkins
School of Computing Science
University of Glasgow
Glasgow  G12 8QQ
United Kingdom

Email: csp@csperkins.org
RTP Payload Format for the SCIP Codec
draft-ietf-avtcore-rtp-scip-00.txt

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 22, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

This document describes the RTP payload format of the Secure Communication Interoperability Protocol (SCIP) as audio and video media subtypes. It provides RFC 6838 compliant media subtype definitions. SCIP-214.2 and SCIP-210 describe the protocols that comprise the SCIP RTP packet payload. This document follows the registration for related media types called "audio/scip" and "video/scip" with IANA and formatted according to RFC 4855.

Table of Contents

1. Introduction..............................................2
   1.1. Conventions..........................................3
   1.2. Abbreviations........................................3
2. Background................................................3
3. Media Format Description..................................4
4. Payload Format............................................5
   4.1. RTP Header Fields....................................5
5. Payload Format Parameters.................................5
   5.1. Media Subtype "audio/scip"...........................6
   5.2. Media Subtype "video/scip"...........................7
   5.3. Mapping to SDP.......................................8
   5.4. SDP Offer/Answer Considerations......................9
6. Security Considerations...................................9
7. IANA Considerations......................................9
8. References................................................9
   8.1. Normative References................................9
   8.2. Informative References..............................11
9. Authors' Addresses.......................................12

1. Introduction

The IANA registration of media subtype types in the IETF tree created two similar media subtypes "scip" under the audio and video media types [AUDIOSCIP], [VIDEOSCIP]. This document, as the common top-level reference, provides information on their similarities and differences and the usage of those media subtypes.

This document details usage of the SCIP pseudo-codec as a secure session establishment protocol and transport protocol over RTP. It provides a reference for network security policymakers, network equipment OEMs, procurement personnel, and government agency and commercial industry representatives.
1.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Best current practices for writing an RTP payload format specification were followed [RFC2736] [RFC8088].

1.2. Abbreviations

The following abbreviations are used in this document.

- AVP: Audio/Video Profile
- DTX: Discontinuous Transmission
- FNBDT: Future Narrowband Digital Terminal
- ICWG: Interoperability Control Working Group
- IICWG: International Interoperability Control Working Group
- NATO: North Atlantic Treaty Organization
- SCIP: Secure Communication Interoperability Protocol
- SDP: Session Description Protocol

2. Background

The Secure Communication Interoperability Protocol (SCIP) allows the negotiation of several voice, data, and video applications using various encryption suites. SCIP also provides several important characteristics that have led to its broad acceptance in the international user community. These features include end-to-end security at the application layer, authentication of user identity, the ability to apply different security levels for each secure session, and secure communication over any end-to-end data connection.

SCIP began in the U.S. as the Future Narrowband Digital Terminal (FNBDT) Protocol. A combined Department of Defense and vendor consortium formed a governing organization named the Interoperability Control Working Group (ICWG) to manage the protocol. In time, the group expanded to include NATO, NATO partners and European vendors under the name International Interoperability Control Working Group (IICWG), which was later renamed the SCIP Working Group.
SCIP is presently implemented in U.S. and NATO secure voice, video, and data products operating on commercial, private, and tactical IP networks worldwide using the scip media subtype. First generation SCIP devices operated on circuit-switched networks. SCIP was then expanded to radio and IP networks. The scip media subtype transports SCIP secure session establishment signaling and secure application traffic. The built-in negotiation and flexibility provided by the SCIP standards make it a natural choice for many scenarios that require various secure applications and associated encryption suites. SCIP has been endorsed by many nations as the secure end-to-end solution for secure voice, video, and data devices. SCIP standards are currently available to participating government/military communities and select OEMs of equipment that support SCIP.

However, SCIP must operate over global networks (including private and commercial networks). Without access to necessary information to support SCIP, some networks may not support the SCIP media subtypes. Issues may occur simply because information is not as readily available to OEMs, network administrators, and network architects.

This RFC provides essential information about audio/scip and video/scip media subtypes that enables network equipment manufacturers to include scip as a known audio and video media subtype in their equipment and enables network administrators to define and implement a compatible security policy. All current IP-based SCIP devices support "scip" as a media subtype. Registration of scip as a media subtype provides a common reference for network equipment manufacturers to recognize SCIP in a payload declaration.

3. Media Format Description

The "scip" media subtype indicates support for and identifies SCIP traffic that is being transferred using RTP. SCIP traffic requires end-to-end bit integrity, therefore transcoding SHALL NOT be performed over the end-to-end IP connection. The audio/scip and video/scip media subtype data streams within the network, including the VoIP network, MUST be a transparent relay and be treated as "clear-channel data", similar to the Clearmode media subtype defined by RFC 4040. However, Clearmode is defined as a gateway protocol and limited to a sample rate of 8000 Hz and 64kbps bandwidth only [RFC4040].
Clearmode is not defined for the higher sample and data rates required for some SCIP traffic.

4. Payload Format

The RTP Packet content of SCIP traffic is dependent upon the SCIP session state. SCIP secure session establishment uses protocols defined in SCIP-210 [SCIP210] to negotiate an application. SCIP secure traffic may consist of the encrypted output of codecs such as MELPe [RFC8130], G.729D [RFC3551], H.264 [RFC6184], or other media encodings, based on the application negotiated during SCIP secure session establishment. SCIP traffic is highly variable and may include other SCIP signaling information in the media stream. SCIP traffic may not always be a continuous stream at the bit rate specified in the SDP [RFC8866] since discontinuous transmission (DTX) or other mechanisms may be used. The SCIP payload size will vary, especially during SCIP secure session establishment.

4.1. RTP Header Fields

The SCIP RTP header fields SHALL conform to RFC 3550.

SCIP traffic may be continuous or discontinuous. The Timestamp field increments based on the sampling clock for discontinuous transmission as described in [RFC3550], Section 5.1. The Timestamp field for continuous transmission applications is dependent on the sampling rate of the media as specified in the media subtype’s specification (e.g., MELPe [RFC8130]). Note that during a call, both discontinuous and continuous traffic are highly probable. Therefore, a jitter buffer MAY be implemented in endpoint devices only but SHOULD NOT be implemented in network devices. Additionally, network devices SHOULD NOT repacketize SCIP packets.

The Marker bit SHALL be set to zero for discontinuous traffic. The Marker bit for continuous traffic is based on the underlying media subtype specification. The underlying media is opaque within SCIP RTP packets.

5. Payload Format Parameters

The SCIP RTP payload format is identified using the scip media subtype, which is registered in accordance with [RFC4855] and per the media type registration template form [RFC6838]. A clock rate of 8000 Hz SHALL be used for "audio/scip". A clock rate of 90000 Hz SHALL be used for "video/scip".
5.1. Media Subtype "audio/scip"

Media type name: audio
Media subtype name: scip
Required parameters: N/A
Optional parameters: N/A

Encoding considerations: Binary. This media subtype is only defined for transfer via RTP. There SHALL be no encoding/decoding (transcoding) of the audio stream as it traverses the network.

Interoperability considerations: N/A
Published specifications: [SCIP214], [SCIP210]
Applications which use this media: N/A
Fragment Identifier considerations: none
Restrictions on usage: N/A

Additional information:
1. Deprecated alias names for this type: N/A
2. Magic number(s): N/A
3. File extension(s): N/A
4. Macintosh file type code: N/A
5. Object Identifiers: N/A

Person to contact for further information:
1. Name: Michael Faller and Daniel Hanson
2. Email: michael.faller@gd-ms.com and dan.hanson@gd-ms.com

Intended usage: Common, Government and Military
5.2. Media Subtype "video/scip"

Media type name: video
Media subtype name: scip
Required parameters: N/A
Optional parameters: N/A

Encoding considerations: Binary. This media subtype is only defined for transfer via RTP. There SHALL be no encoding/decoding (transcoding) of the video stream as it traverses the network.

Interoperability considerations: N/A

Published specifications: [SCIP214], [SCIP210]
Applications which use this media: N/A
Fragment Identifier considerations: none
Restrictions on usage: N/A
Additional information:

1. Deprecated alias names for this type: N/A
2. Magic number(s): N/A
3. File extension(s): N/A
4. Macintosh file type code: N/A
5. Object Identifiers: N/A

Person to contact for further information:

1. Name: Michael Faller and Daniel Hanson
2. Email: michael.faller@gd-ms.com and dan.hanson@gd-ms.com

Intended usage: Common, Government and Military

Authors:

Michael Faller - michael.faller@gd-ms.com
Daniel Hanson - dan.hanson@gd-ms.com

Change controller:

SCIP Working Group - ncia.cis3@ncia.nato.int

5.3. Mapping to SDP

The mapping of the above defined payload format media subtype and its parameters SHALL be done according to Section 3 of [RFC4855].

An example mapping for audio/scip is:

```plaintext
m=audio 50000 RTP/AVP 96
a=rtpmap:96 scip/8000
```

An example mapping for video/scip is:

```plaintext
m=video 50002 RTP/AVP 97
a=rtpmap:97 scip/90000
```

An example mapping for both audio/scip and video/scip is:

```plaintext
m=audio 50000 RTP/AVP 96
a=rtpmap:96 scip/8000
m=video 50002 RTP/AVP 97
a=rtpmap:97 scip/90000
```

The application negotiation between endpoints will determine whether the audio and video streams are transported as separate
streams over the audio and video payload types or as a single media stream on the video payload type.

5.4. SDP Offer/Answer Considerations

In accordance with the SDP Offer/Answer model [RFC3264], the SCIP device SHALL list the SCIP payload type in order of preference in the "m" media line.

6. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. However, as "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format’s responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity, and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this Security Considerations section discusses the security impacting properties of the payload format itself.

This RTP payload format and its media decoder do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing, and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Nor does the RTP payload format contain any active content.

7. IANA Considerations

The audio/scip and video/scip media subtypes have been registered with IANA [AUDIOSCIP] [VIDEOSCIP].

8. References

8.1. Normative References

[AUDIOSCIP] Faller, M., and D. Hanson, "audio/scip", Internet Assigned Numbers Authority (IANA), 28 January 2021,
<https://www.iana.org/assignments/media-types/audio/scip>.


8.2. Informative References


9. Authors’ Addresses

Daniel Hanson
General Dynamics Mission Systems, Inc.
150 Rustcraft Road
Dedham, MA 02026, USA
E-mail: dan.hanson@gd-ms.com

Michael Faller
General Dynamics Mission Systems, Inc.
150 Rustcraft Road
Dedham, MA 02026, USA
E-mail: michael.faller@gd-ms.com

Keith Maver
General Dynamics Mission Systems, Inc.
150 Rustcraft Road
Dedham, MA 02026, USA
E-mail: keith.maver@gd-ms.com

SCIP Working Group, CIS3 Partnership
NATO Communications and Information Agency
Oude Waalsdorperweg 61, 2597AK
The Hague, The Netherlands
E-mail: ncia.cis3@ncia.nato.int
Abstract

This document describes a Frame Marking RTP header extension used to convey information about video frames that is critical for error recovery and packet forwarding in RTP middleboxes or network nodes. It is most useful when media is encrypted, and essential when the middlebox or node has no access to the media decryption keys. It is also useful for codec-agnostic processing of encrypted or unencrypted media, while it also supports extensions for codec-specific information.

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 5 May 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

1. Introduction

Many widely deployed RTP [RFC3550] topologies [RFC7667] used in modern voice and video conferencing systems include a centralized component that acts as an RTP switch. It receives voice and video streams from each participant, which may be encrypted using SRTP [RFC3711], or extensions that provide participants with private media [RFC8871] via end-to-end encryption where the switch has no access to media decryption keys. The goal is to provide a set of streams back to the participants which enable them to render the right media content. In a simple video configuration, for example, the goal will be that each participant sees and hears just the active speaker. In that case, the goal of the switch is to receive the voice and video streams from each participant, determine the active speaker based on energy in the voice packets, possibly using the client-to-mixer audio level RTP header extension [RFC6464], and select the corresponding video stream for transmission to participants; see Figure 1.
In this document, an "RTP switch" is used as a common short term for the terms "switching RTP mixer", "source projecting middlebox", "source forwarding unit/middlebox" and "video switching MCU" as discussed in [RFC7667].

```
+---+      +------------+      +---+
| A |<---->|            |<---->| B |
++---+      +            +      +---+
|   RTP  |      Switch    +      +---+
++---+      +------------+      +---+
| C |<---->|            |<---->| D |
++---+      +------------+      +---+
```

Figure 1: RTP switch

In order to properly support switching of video streams, the RTP switch typically needs some critical information about video frames in order to start and stop forwarding streams.

* Because of inter-frame dependencies, it should ideally switch video streams at a point where the first frame from the new speaker can be decoded by recipients without prior frames, e.g. switch on an intra-frame.
* In many cases, the switch may need to drop frames in order to realize congestion control techniques, and needs to know which frames can be dropped with minimal impact to video quality.
* For scalable streams with dependent layers, the switch may need to selectively forward specific layers to specific recipients due to recipient bandwidth or decoder limits.
* Furthermore, it is highly desirable to do this in a payload format-agnostic way which is not specific to each different video codec. Most modern video codecs share common concepts around frame types and other critical information to make this codec-agnostic handling possible.
* It is also desirable to be able to do this for SRTP without requiring the video switch to decrypt the packets. SRTP will encrypt the RTP payload format contents and consequently this data is not usable for the switching function without decryption, which may not even be possible in the case of end-to-end encryption of private media [RFC8871].

By providing meta-information about the RTP streams outside the encrypted media payload, an RTP switch can do codec-agnostic selective forwarding without decrypting the payload. This document specifies the necessary meta-information in an RTP header extension.
2. Key Words for Normative Requirements

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].

3. Frame Marking RTP Header Extension

This specification uses RTP header extensions as defined in [RFC8285]. A subset of meta-information from the video stream is provided as an RTP header extension to allow an RTP switch to do generic selective forwarding of video streams encoded with potentially different video codecs.

The Frame Marking RTP header extension is encoded using the one-byte header or two-byte header as described in [RFC8285]. The one-byte header format is used for examples in this memo. The two-byte header format is used when other two-byte header extensions are present in the same RTP packet, since mixing one-byte and two-byte extensions is not possible in the same RTP packet.

This extension is only specified for Source (not Redundancy) RTP Streams [RFC7656] that carry video payloads. It is not specified for audio payloads, nor is it specified for Redundancy RTP Streams. The (separate) specifications for Redundancy RTP Streams often include provisions for recovering any header extensions that were part of the original source packet. Such provisions SHALL be followed to recover the Frame Marking RTP header extension of the original source packet. Source packet frame markings may be useful when generating Redundancy RTP Streams; for example, the I and D bits can be used to generate extra or no redundancy, respectively, and redundancy schemes with source blocks can align source block boundaries with Independent frame boundaries as marked by the I bit.

A frame, in the context of this specification, is the set of RTP packets with the same RTP timestamp from a specific RTP synchronization source (SSRC). A frame within a layer is the set of RTP packets with the same RTP timestamp, SSRC, Temporal ID (TID), and Layer ID (LID).
3.1. Long Extension for Scalable Streams

The following RTP header extension is RECOMMENDED for scalable streams. It MAY also be used for non-scalable streams, in which case TID, LID and TL0PICIDX MUST be 0 or omitted. The ID is assigned per [RFC8285], and the length is encoded as L=2 which indicates 3 octets of data when nothing is omitted, or L=1 for 2 octets when TL0PICIDX is omitted, or L=0 for 1 octet when both LID and TL0PICIDX are omitted.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=? |  L=2  |S|E|I|D|B| TID |   LID         |    TL0PICIDX  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
or
```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=? |  L=1  |S|E|I|D|B| TID |   LID         | (TL0PICIDX omitted)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
or
```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=? |  L=0  |S|E|I|D|B| TID | (LID and TL0PICIDX omitted)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The following information are extracted from the media payload and sent in the Frame Marking RTP header extension.

* S: Start of Frame (1 bit) - MUST be 1 in the first packet in a frame within a layer; otherwise MUST be 0.
* E: End of Frame (1 bit) - MUST be 1 in the last packet in a frame within a layer; otherwise MUST be 0. Note that the RTP header marker bit MAY be used to infer the last packet of the highest enhancement layer, in payload formats with such semantics.
* I: Independent Frame (1 bit) - MUST be 1 for a frame within a layer that can be decoded independent of temporally prior frames, e.g. intra-frame, VPX keyframe, H.264 IDR [RFC6184], H.265 IDR/CRA/BLA/RAPI [RFC7798]; otherwise MUST be 0. Note that this bit only signals temporal independance, so it can be 1 in spatial or quality enhancement layers that depend on temporally co-located layers but not temporally prior frames.
* D: Discardable Frame (1 bit) - MUST be 1 for a frame within a layer the sender knows can be discarded, and still provide a decodable media stream; otherwise MUST be 0.
* B: Base Layer Sync (1 bit) - When TID is not 0, this MUST be 1 if the sender knows this frame within a layer only depends on the base temporal layer; otherwise MUST be 0. When TID is 0 or if no scalability is used, this MUST be 0.
* TID: Temporal ID (3 bits) - Identifies the temporal layer/sub-layer encoded, starting with 0 for the base layer, and increasing with higher temporal fidelity. If no scalability is used, this MUST be 0. It is implicitly 0 in the short extension format.

* LID: Layer ID (8 bits) - Identifies the spatial and quality layer encoded, starting with 0 for the base layer, and increasing with higher fidelity. If no scalability is used, this MUST be 0 or omitted to reduce length. When omitted, TL0PICIDX MUST also be omitted. It is implicitly 0 in the short extension format or when omitted in the long extension format.

* TL0PICIDX: Temporal Layer 0 Picture Index (8 bits) - When TID is 0 and LID is 0, this is a cyclic counter labeling base layer frames. When TID is not 0 or LID is not 0, this indicates a dependency on the given index, such that this frame within this layer depends on the frame with this label in the layer with TID 0 and LID 0. If no scalability is used, or the cyclic counter is unknown, this MUST be omitted to reduce length. Note that 0 is a valid index value for TL0PICIDX.

The layer information contained in TID and LID convey useful aspects of the layer structure that can be utilized in selective forwarding.

Without further information about the layer structure, these TID/LID identifiers can only be used for relative priority of layers and implicit dependencies between layers. They convey a layer hierarchy with TID=0 and LID=0 identifying the base layer. Higher values of TID identify higher temporal layers with higher frame rates. Higher values of LID identify higher spatial and/or quality layers with higher resolutions and/or bitrates. Implicit dependencies between layers assume that a layer with a given TID/LID MAY depend on layer(s) with the same or lower TID/LID, but MUST NOT depend on layer(s) with higher TID/LID.

With further information, for example, possible future RTCP SDES items that convey full layer structure information, it may be possible to map these TIDs and LIDs to specific absolute frame rates, resolutions and bitrates, as well as explicit dependencies between layers. Such additional layer information may be useful for forwarding decisions in the RTP switch, but is beyond the scope of this memo. The relative layer information is still useful for many selective forwarding decisions even without such additional layer information.
3.2. Short Extension for Non-Scalable Streams

The following RTP header extension is RECOMMENDED for non-scalable streams. It is identical to the shortest form of the extension for scalable streams, except the last four bits (B and TID) are replaced with zeros. It MAY also be used for scalable streams if the sender has limited or no information about stream scalability. The ID is assigned per [RFC8285], and the length is encoded as L=0 which indicates 1 octet of data.

```
0                   1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=? |  L=0  |S|E|I|D|0 0 0 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The following information are extracted from the media payload and sent in the Frame Marking RTP header extension.

* S: Start of Frame (1 bit) - MUST be 1 in the first packet in a frame; otherwise MUST be 0.
* E: End of Frame (1 bit) - MUST be 1 in the last packet in a frame; otherwise MUST be 0. SHOULD match the RTP header marker bit in payload formats with such semantics for marking end of frame.
* I: Independent Frame (1 bit) - MUST be 1 for frames that can be decoded independent of temporally prior frames, e.g. intra-frame, VPX keyframe, H.264 IDR [RFC6184], H.265 IDR/CRA/BLA/IRAP [RFC7798]; otherwise MUST be 0.
* D: Discardable Frame (1 bit) - MUST be 1 for frames the sender knows can be discarded, and still provide a decodable media stream; otherwise MUST be 0.
* The remaining (4 bits) - are reserved/fixed values and not used for non-scalable streams; they MUST be set to 0 upon transmission and ignored upon reception.

3.3. Layer ID Mappings for Scalable Streams

This section maps the specific Layer ID information contained in specific scalable codecs to the generic LID and TID fields.

Note that non-scalable streams have no Layer ID information and thus no mappings.

3.3.1. VP9 LID Mapping

The following shows the VP9 [I-D.ietf-payload-vp9] Spatial Layer ID (SID, 3 bits) and Temporal Layer ID (TID, 3 bits) from the VP9 payload descriptor mapped to the generic LID and TID fields.
The S bit MUST match the B bit in the VP9 payload descriptor.

The E bit MUST match the E bit in the VP9 payload descriptor.

The I bit MUST match the inverse of the P bit in the VP9 payload descriptor.

The D bit MUST be 1 if the refresh_frame_flags in the VP9 payload uncompressed header are all 0, otherwise it MUST be 0.

The B bit MUST be 0 if TID is 0; otherwise, if TID is not 0, it MUST match the U bit in the VP9 payload descriptor. Note: When using temporally nested scalability structures as recommended in Section 3.5.2, the B bit and VP9 U bit will always be 1 if TID is not 0, since it is always possible to switch up to a higher temporal layer in such nested structures.

TID and TL0PICIDX MUST match the correspondingly named fields in the VP9 payload descriptor.

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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=? |  L=2  |S|E|I|D|B| TID |0|0|0|0|0| SID |    TL0PICIDX  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

3.3.2.  H265 LID Mapping

The following shows the H265 [RFC7798] LayerID (6 bits) and TID (3 bits) from the NAL unit header mapped to the generic LID and TID fields.

The S and E bits MUST match the correspondingly named bits in PACI:PHES:TSCI payload structures.

The I bit MUST be 1 when the NAL unit type is 16-23 (inclusive) or 32-34 (inclusive), or an aggregation packet or fragmentation unit encapsulating any of these types, otherwise it MUST be 0. These ranges cover intra (IRAP) frames as well as critical parameter sets (VPS, SPS, PPS).

The D bit MUST be 1 when the NAL unit type is 0, 2, 4, 6, 8, 10, 12, 14, or 38, or an aggregation packet or fragmentation unit encapsulating only these types, otherwise it MUST be 0. These ranges cover non-reference frames as well as filler data.
The B bit can not be determined reliably from simple inspection of payload headers, and therefore is determined by implementation-specific means. For example, internal codec interfaces may provide information to set this reliably.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ID=? | L=2 | S | E | I | D | B | TID | 0 | 0 | LayerID | TL0PICIDX |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

### 3.3.3. H264-SVC LID Mapping

The following shows H264-SVC [RFC6190] Layer encoding information (3 bits for spatial/dependency layer, 4 bits for quality layer and 3 bits for temporal layer) mapped to the generic LID and TID fields.

The S, E, I and D bits MUST match the correspondingly named bits in PACSI payload structures.

The I bit MUST be 1 when the NAL unit type is 5, 7, 8, 13, or 15, or an aggregation packet or fragmentation unit encapsulating any of these types, otherwise it MUST be 0. These ranges cover intra (IDR) frames as well as critical parameter sets (SPS/PPS variants).

The D bit MUST be 1 when the NAL unit header NRI field is 0, or an aggregation packet or fragmentation unit encapsulating only NAL units with NRI=0, otherwise it MUST be 0. The NRI=0 condition signals non-reference frames.

The B bit can not be determined reliably from simple inspection of payload headers, and therefore is determined by implementation-specific means. For example, internal codec interfaces may provide information to set this reliably.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ID=? | L=2 | S | E | I | D | B | TID | 0 | DID | QID | TL0PICIDX |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

### 3.3.4. H264 (AVC) LID Mapping

The following shows the header extension for H264 (AVC) [RFC6184] that contains only temporal layer information.

The I bit MUST be 1 when the NAL unit type is 5, 7, 8, 13, or 15, or an aggregation packet or fragmentation unit encapsulating any of these types, otherwise it MUST be 0. These ranges cover intra (IDR) frames as well as critical parameter sets (SPS/PPS variants).

The D bit MUST be 1 when the NAL unit header NRI field is 0, or an aggregation packet or fragmentation unit encapsulating only NAL units with NRI=0, otherwise it MUST be 0. The NRI=0 condition signals non-reference frames.

The B bit can not be determined reliably from simple inspection of payload headers, and therefore is determined by implementation-specific means. For example, internal codec interfaces may provide information to set this reliably.
The S bit MUST be 1 when the timestamp in the RTP header differs from the timestamp in the prior RTP sequence number from the same SSRC, otherwise it MUST be 0.

The E bit MUST match the M bit in the RTP header.

The I bit MUST be 1 when the NAL unit type is 5, 7, or 8, or an aggregation packet or fragmentation unit encapsulating any of these types, otherwise it MUST be 0. These ranges cover intra (IDR) frames as well as critical parameter sets (SPS/PPS).

The D bit MUST be 1 when the NAL unit header NRI field is 0, or an aggregation packet or fragmentation unit encapsulating only NAL units with NRI=0, otherwise it MUST be 0. The NRI=0 condition signals non-reference frames.

The B bit can not be determined reliably from simple inspection of payload headers, and therefore is determined by implementation-specific means. For example, internal codec interfaces may provide information to set this reliably.

```
+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+
| ID=? | L=2 |S|E|I|D|B| TID |0|0|0|0|0|0|0|0|    TL0PICIDX  |
+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+-------------------+
```

3.3.5. VP8 LID Mapping

The following shows the header extension for VP8 [RFC7741] that contains only temporal layer information.

The S bit MUST match the correspondingly named bit in the VP8 payload descriptor when PID=0, otherwise it MUST be 0.

The E bit MUST match the M bit in the RTP header.

The I bit MUST match the inverse of the P bit in the VP8 payload header.

The D bit MUST match the N bit in the VP8 payload descriptor.

The B bit MUST match the Y bit in the VP8 payload descriptor. Note: When using temporally nested scalability structures as recommended in Section 3.5.2, the B bit and VP8 Y bit will always be 1 if TID is not 0, since it is always possible to switch up to a higher temporal layer in such nested structures.
TID and TL0PICIDX MUST match the correspondingly named fields in the VP8 payload descriptor.

<table>
<thead>
<tr>
<th>ID=?</th>
<th>L=2</th>
<th>S</th>
<th>E</th>
<th>I</th>
<th>D</th>
<th>B</th>
<th>TID</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>TL0PICIDX</th>
</tr>
</thead>
</table>
+-----------------------------------------------------------------+

3.3.6. Future Codec LID Mapping

The RTP payload format specification for future video codecs SHOULD include a section describing the LID mapping and TID mapping for the codec.

3.4. Signaling Information

The URI for declaring this header extension in an extmap attribute is "urn:ietf:params:rtp-hdrext:framemarking". It does not contain any extension attributes.

An example attribute line in SDP:

    a=extmap:3 urn:ietf:params:rtp-hdrext:framemarking

3.5. Usage Considerations

The header extension values MUST represent what is already in the RTP payload.

When an RTP switch needs to discard a received video frame due to congestion control considerations, it is RECOMMENDED that it preferably drop frames marked with the D (Discardable) bit set, or the highest values of TID and LID, which indicate the highest temporal and spatial/quality enhancement layers, since those typically have fewer dependences on them than lower layers.

When an RTP switch wants to forward a new video stream to a receiver, it is RECOMMENDED to select the new video stream from the first switching point with the I (Independent) bit set in all spatial layers and forward the same. An RTP switch can request a media source to generate a switching point by sending Full Intra Request (RTCP FIR) as defined in [RFC5104], for example.
3.5.1. Relation to Layer Refresh Request (LRR)

Receivers can use the Layer Refresh Request (LRR) [I-D.ietf-avtext-lrr] RTCP feedback message to upgrade to a higher layer in scalable encodings. The TID/LID values and formats used in LRR messages MUST correspond to the same values and formats specified in Section 3.1.

Because frame marking can only be used with temporally-nested streams, temporal-layer LRR refreshes are unnecessary for frame-marked streams. Other refreshes can be detected based on the I bit being set for the specific spatial layers.

3.5.2. Scalability Structures

The LID and TID information is most useful for fixed scalability structures, such as nested hierarchical temporal layering structures, where each temporal layer only references lower temporal layers or the base temporal layer. The LID and TID information is less useful, or even not useful at all, for complex, irregular scalability structures that do not conform to common, fixed patterns of inter-layer dependencies and referencing structures. Therefore it is RECOMMENDED to use LID and TID information for RTP switch forwarding decisions only in the case of temporally nested scalability structures, and it is NOT RECOMMENDED for other (more complex or irregular) scalability structures.

4. Security Considerations

In the Secure Real-Time Transport Protocol (SRTP) [RFC3711], RTP header extensions are authenticated but usually not encrypted. When header extensions are used some of the payload type information are exposed and visible to middle boxes. The encrypted media data is not exposed, so this is not seen as a high risk exposure.

5. Acknowledgements

Many thanks to Bernard Aboba, Jonathan Lennox, Stephan Wenger, Dale Worley, and Magnus Westerlund for their inputs.

6. IANA Considerations

This document defines a new extension URI to the RTP Compact HeaderExtensions sub-registry of the Real-Time Transport Protocol (RTP) Parameters registry, according to the following data:
7. References

7.1. Normative References


7.2. Informative References


Authors’ Addresses
Mo Zanaty  
Cisco Systems  
170 West Tasman Drive  
San Jose, CA 95134  
United States of America  
Email: mzanaty@cisco.com

Espen Berger  
Cisco Systems  
Email: espeberg@cisco.com

Suhas Nandakumar  
Cisco Systems  
170 West Tasman Drive  
San Jose, CA 95134  
United States of America  
Email: snandaku@cisco.com
Abstract

This memo describes an RTP payload format for visual volumetric video-based coding (V3C) [ISO.IEC.23090-5]. A V3C bitstream is composed of V3C units that contain V3C video sub-bitstreams, V3C atlas sub-bitstreams, or a V3C parameter set. The RTP payload format for V3C video sub-bitstreams is defined by appropriate Internet Standards for the applicable video codec. The RTP payload format for V3C atlas sub-bitstreams is described by this memo. The RTP payload format allows for packetization of one or more V3C Network Abstraction Layer (NAL) units in a RTP packet payload as well as fragmentation of a V3C NAL unit into multiple RTP packets. The memo also describes the mechanisms for grouping RTP streams of V3C component sub-bitstreams, providing a complete solution for streaming V3C bitstream.

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 28 October 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.
This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction ........................................... 3
2. Conventions ........................................... 3
3. Definitions, and Abbreviations ......................... 4
   3.1. Definitions ........................................ 4
   3.1.1. General ......................................... 4
   3.1.2. Definitions from the V3C Specification .......... 4
   3.1.3. Definitions Specific to This Memo ............. 6
3.2. Abbreviations ....................................... 6
4. Media Format Description ............................... 6
   4.1. Overview of the V3C codec .......................... 7
   4.2. V3C parameter set .................................. 8
   4.3. V3C atlas and video components .................... 8
   4.3.1. General ......................................... 8
   4.3.2. Atlas NAL units ................................ 10
4.4. Systems and transport interfaces ..................... 11
5. V3C Atlas RTP payload format ........................... 11
   5.1. General ........................................... 11
   5.2. RTP Header ........................................ 11
   5.3. RTP payload header ................................ 13
   5.4. Transmission modes ................................ 14
   5.5. Payload structures ................................ 14
   5.5.1. General ......................................... 15
   5.5.2. Single NAL unit packet ......................... 15
   5.5.3. Aggregation packet .............................. 17
   5.5.4. Fragmentation unit .............................. 19
   5.6. Decoding Order Number ............................. 21
6. Packetization and de-packetization rules ............... 22
7. Payload Examples ...................................... 23
   7.1. General ........................................... 23
   7.2. V3C fragmentation unit ............................. 23
8. Payload Format Parameters ................................ 25
   8.1. Media Type Definition .............................. 25
9. Congestion Control Considerations ....................... 29
10. Session Description Protocol .......................... 29
   10.1. Mapping of payload type parameters to SDP ........ 29
   10.1.1. For V3C atlas components ...................... 29
   10.1.2. For V3C video components ...................... 30
Volumetric video, similar to traditional 2D video, when uncompressed, is represented by a large amount of data. The Visual Volumetric Video-based Coding (V3C) specification [ISO.IEC.23090-5] leverages the compression efficiency of existing 2D video codecs to reduce the amount of data needed for storage and transmission of volumetric video.

V3C encoder converts volumetric frames, 3D volumetric information, into a collection of 2D images and associated data, known as atlas data. The converted 2D images are subsequently coded using existing video or image codecs, e.g. ISO/IEC International Standard 14496-10 [ISO.IEC.14496-10], ISO/IEC International Standard 23008-2 [ISO.IEC.23008-2] or ISO/IEC International Standard 23090-3 [ISO.IEC.23090-3]. The atlas data is coded with mechanisms specified in [ISO.IEC.23090-5]. V3C is a generic mechanism for volumetric video coding and it can be used by applications targeting volumetric content, such as point clouds, immersive video with depth, mesh representations of visual volumetric frames, etc. Examples of such applications are Video-based Point Cloud Compression (V-PCC) [ISO.IEC.23090-5], and MPEG Immersive Video (MIV) [ISO.IEC.23090-12].

V3C utilizes high level syntax (HLS) syntax design known from traditional 2D video codecs to represent the associated coded data, i.e. atlas data. The atlas data is represented by Network Abstraction Layer (NAL) units. Consequently, RTP payload format for V3C atlas data described in this memo shares design philosophy, security, congestion control, and overall implementation complexity with other the NAL unit-based RTP payload formats such as the ones defined in [RFC6184], [RFC6190], and [RFC7798].

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].
All fields defined in this specification related to RTP payload structures SHALL be considered in network order.

3. Definitions, and Abbreviations

3.1. Definitions

3.1.1. General

This document uses the definitions of [ISO.IEC.23090-5]. The following terms, defined in [ISO.IEC.23090-5], are provided here for convenience:

3.1.2. Definitions from the V3C Specification

atlas: collection of 2D bounding boxes and their associated information placed onto a rectangular frame and corresponding to a volume in 3D space on which volumetric data is rendered.

atlas bitstream: sequence of bits that forms the representation of atlas frames and associated data forming one or more CASs.

atlas coding layer NAL unit: collective term for coded atlas tile layer NAL units and the subset of NAL units that have reserved values of nal_unit_type that are classified as being of type class equal to ACL in this document.

atlas frame: 2D rectangular array of atlas samples onto which patches are projected and additional information related to the patches, corresponding to a volumetric frame.

atlas frame parameter set: syntax structure containing syntax elements that apply to zero or more entire coded atlas frames as determined by the content of a syntax element found in each tile header.

atlas sequence parameter set: syntax structure containing syntax elements that apply to zero or more entire coded atlas sequences as determined by the content of a syntax element found in the AFPS referred to by a syntax element found in each tile header.

attribute: scalar or vector property optionally associated with each point in a volumetric frame such as colour, reflectance, surface normal, time stamps, material ID, etc.

coded atlas sequence: sequence of coded atlas access units, in decoding order, of an IRAP coded atlas access unit, followed by zero or more coded atlas access units that are not IRAP coded atlas access
units, including all subsequent access units up to but not including any subsequent coded atlas access unit that is an IRAP coded atlas access unit.

coded atlas access unit: set of atlas NAL units that are associated with each other according to a specified classification rule, are consecutive in decoding order, and contain all atlas NAL units pertaining to one particular output time.

intra random access point coded atlas: coded atlas for which each ACL NAL unit has nal_unit_type in the range of NAL_BLA_W_LP to NAL_RSV_IRAP_ACL_29, inclusive.

intra random access point coded atlas access unit: access unit in which the coded atlas with nal_layer_id equal to 0 is a IRAP coded atlas.

network abstraction layer unit: syntax structure containing an indication of the type of data to follow and bytes containing that data in the form of an RBSP.

patch: rectangular region within an atlas associated with volumetric information.

raw byte sequence payload: syntax structure containing an integer number of bytes that is encapsulated in a NAL unit and that is either empty or has the form of a string of data bits containing syntax elements followed by an RBSP stop bit and zero or more subsequent bits equal to 0.

tile: independently decodable rectangular region of an atlas frame

visual volumetric video-based coding atlas sub-bitstream: extracted sub-bitstream from the V3C bitstream containing whole or portion of an atlas bitstream.

visual volumetric video-based coding video sub-bitstream: extracted sub-bitstream from the V3C bitstream containing whole or portion of an video bitstream.

visual volumetric video-based coding component: atlas, occupancy, geometry, or attribute of a particular type that is associated with a V3C volumetric content representation.

visual volumetric video-based coding parameter set: syntax structure containing syntax elements that apply to zero or more entire CVSs and may be referred to by syntax elements found in the V3C unit header.
volumetric frame: set of 3D points specified by their cartesian coordinates and zero or more corresponding sets of attributes at a particular time instance.

3.1.3. Definitions Specific to This Memo

Placeholder

3.2. Abbreviations

ACL atlas coding layer
AFPS atlas frame parameter set
AP aggregation packet
ASPS atlas sequence parameter set
AU aggregation unit
CAS coded atlas sequence
DON decoding order number
IRAP intra random access point
MRMT Multiple RTP streams on Multiple media Transports
MRST Multiple RTP streams over a Single media Transport
MTU maximum transmission unit
NAL network abstraction layer
NALU NAL unit
RBSP raw byte sequence payload
SRST Single RTP stream on a Single media Transport
V3C visual volumetric video-based coding
VPS V3C parameter set

4. Media Format Description
4.1. Overview of the V3C codec

ISO/IEC International Standards 23090-5 [ISO.IEC.23090-5] enables encoding and decoding processes of volumetric video which utilizes 2D video coding technologies and associated data. V3C encoding of volumetric frame is achieved through a conversion of volumetric frame from its 3D representation to multiple 2D representations and a generation of associated data.

2D representations, known as V3C video components, of volumetric frame are encoded using traditional 2D video codecs. V3C video component may, for example, include occupancy, geometry, or attribute data. The occupancy data informs a V3C decoder which pixels in other V3C video components contribute to reconstructed 3D representation. The geometry data describes information on the position of the reconstructed voxels, while attribute data provides properties of that voxel, e.g. color or material information.

Atlas data, known as V3C atlas component, provides information to interpret V3C video components and enables the reconstruction from a 2D representation back into a 3D representation of volumetric frame. Atlas data is composed of a collection of patches. Each patch identifies a region in all V3C video components and provides information necessary to perform the appropriate inverse projection of the indicated region back into 3D space. The shape of the patch region is determined by a 2D bounding box associated with each patch as well as their coding order. The shape of these patches is also further refined based on occupancy data.

To enable parallelization, random access, as well as a variety of other functionalities, an atlas frame can be divided into one or more rectangular partitions referred to as tiles. Tiles are not allowed to overlap and SHOULD be independently decodeable. An atlas frame may contain regions that are not associated with any tile or patch.

The binary form of V3C video components, i.e. video bitstream, and V3C atlas components, i.e. V3C atlas bitstream, can be grouped and represented by a single V3C bitstream. The V3C bitstream is composed of a set of V3C units. Each V3C unit has a V3C unit header and a V3C unit payload. The V3C unit header describes the V3C unit type for the payload. V3C unit payload contains V3C video components, V3C atlas components or V3C parameter set. V3C video component, i.e. occupancy, geometry, and attribute, corresponds to video data units (e.g. NAL units defined in [ISO.IEC.23008-2]) that could be decoded by an appropriate video decoder.
4.2. V3C parameter set

While this memo intends to describe encapsulation of V3C atlas data, aspects related to signaling of V3C parameter set need to be considered. V3C parameter set is signaled in its own V3C unit, which allows decoupling the transmission of V3C parameter set from the V3C video and atlas components. V3C parameter set can be transmitted by external means (e.g., as a result of the capability exchange) or through a (reliable or unreliable) control protocol. This memo provides information how V3C parameter set can be signaled as part of session description protocol, see Section 10.

4.3. V3C atlas and video components

4.3.1. General

In V3C bitstream the atlas component is identified by vuh_unit_type equal to V3C_AD in the V3C unit header. The V3C atlas component consists of atlas NAL units that define header and payload pairs and are described in Section 4.3.2. V3C video components are identified by vuh_unit_type equal to V3C_OVD, V3C_GVD, V3C_AVD, and V3C_PVD respectively. V3C video components can be further separated by other values in the V3C unit header such as vuh_attribute_index, vuh_attribute_partition_index, vuh_map_index and vuh_auxiliary_video_flag. By mapping V3C parameter set information to vuh_attribute_index, a V3C decoder identifies which attribute a given V3C video component contains, e.g. color.

The information supplied by V3C unit header SHOULD be provided in one form or another to a V3C decoder, e.g. as part of SDP as described in this memo in Section 10. The four-byte V3C unit header syntax and semantics are copied below as defined in [ISO.IEC.23090-5].
```c
v3c_unit_header( ) {
    unsigned int(5) vuh_unit_type;
    if( vuh_unit_type == V3C_AVD || vuh_unit_type == V3C_GVD ||
        vuh_unit_type == V3C_OVD || vuh_unit_type == V3C_AD ||
        vuh_unit_type == V3C_CAD || vuh_unit_type == V3C_PVD ) {
        unsigned int(4) vuh_v3c_parameter_set_id;
    }
    if( vuh_unit_type == V3C_AVD || vuh_unit_type == V3C_GVD ||
        vuh_unit_type == V3C_OVD || vuh_unit_type == V3C_AD ||
        vuh_unit_type == V3C_PVD ) {
        unsigned int(6) vuh_atlas_id;
    }
    if( vuh_unit_type == V3C_AVD ) {
        unsigned int(7) vuh_attribute_index;
        unsigned int(5) vuh_attribute_partition_index;
        unsigned int(4) vuh_map_index;
        unsigned int(1) vuh_auxiliary_video_flag;
    }
    if( vuh_unit_type == V3C_GVD ) {
        unsigned int(4) vuh_map_index;
        unsigned int(1) vuh_auxiliary_video_flag;
        bit(12) vuh_reserved_zero_12bits;
    }
    if( vuh_unit_type == V3C_OVD || vuh_unit_type == V3C_AD ||
        vuh_unit_type == V3C_PVD) {
        bit(17) vuh_reserved_zero_17bits;
    } else {
        bit(27) vuh_reserved_zero_27bits;
    }
}
```

vuh_unit_type indicates the V3C unit type for the V3C component as specified in [ISO.IEC.23090-5].

vuh_v3c_parameter_set_id specifies the value of vps_v3c_parameter_set_id for the active V3C VPS.

vuh_atlas_id specifies the ID of the atlas that corresponds to the current V3C unit.

vuh_attribute_index indicates the index of the attribute data carried in the Attribute Video Data unit.

vuh_attribute_partition_index indicates the index of the attribute dimension group carried in the attribute video data unit.
vuh_map_index when present indicates the map index of the current
geometry or attribute stream. When not present, the map index of the
current geometry or attribute sub-bitstream is derived based on the
type of the sub-bitstream.

vuh_auxiliary_video_flag equal indicates if the associated geometry
or attribute video data unit is a RAW and/or EOM coded points video
only sub-bitstream.

4.3.2. Atlas NAL units

Atlas NAL unit (nal_unit(NumBytesInNalUnit)) is a byte-aligned syntax
structure defined by [ISO.IEC.23090-5] to carry atlas data. Atlas NAL
unit always contains a 16-bit NAL unit header (nal_unit_header()),
which indicates among other things the type of the NAL unit
(nal_unit_type). The sample code below describes the NAL unit
syntax, including the NAL unit header.

```c
nal_unit_header(){
    bit(1) nal_forbidden_zero_bit;
    bit(6) nal_unit_type;
    bit(6) nal_layer_id;
    bit(3) nal_temporal_id_plus1;
}
nal_unit(NumBytesInNalUnit){
    nal_unit_header();
    NumBytesInRbsp = 0;
    for( i = 2; i < NumBytesInNalUnit; i++ )
        bit(8) rbsp_byte[ NumBytesInRbsp++ ];
}
```

`nal_forbidden_zero_bit` MUST be equal to 0. (F)

`nal_unit_type` indicates the type of the RBSP data structure contained
in the NAL unit (NUT)

`nal_layer_id` indicates the identifier of the layer to which an ACL
NAL unit belongs or the identifier of a layer to which a non-ACL NAL
unit applies. (NLI)

`nal_temporal_id_plus1` minus 1 indicates a temporal identifier for the
NAL unit. The value of `nal_temporal_id_plus1` MUST NOT be equal to 0.
(TID)
4.4. Systems and transport interfaces

In addition to releasing specifications on V3C [ISO.IEC.23090-5] and [ISO.IEC.23090-12], MPEG is conducting further systems level work on file format level to encapsulate compressed V3C content. The seventh edition of the ISOBMFF specification [ISO.IEC.14496-12] introduces a new media handler ‘volv’, intended to support volumetric visual media. It also specifies other structures to enable development of derived specifications detailing how various volumetric visual media may be stored in ISOBMFF.

One of such derived specifications is [ISO.IEC.23090-10], which focuses on defining how V3C content SHOULD be stored in a file and streamed over DASH. To a large extent ISO/IEC 23090-10 focuses on describing how ISOBMFF boxes and syntax elements may be used to store volumetric media, but in some cases new boxes and syntax elements are introduced to accommodate the fundamentally different type of new media. While the specification is not directly relevant for defining RTP payload format for V3C atlas data, it is a useful resource that SHOULD be considered especially when designing ingestion of encoded V3C content into RTP streaming pipelines.

5. V3C Atlas RTP payload format

5.1. General

This section describes details related to V3C atlas RTP payload definitions. Aspects related to RTP header, RTP payload header and general payload structure are considered along with different packetization modes.

5.2. RTP Header

The format of the RTP header is specified in [RFC3550] and replicated below in Figure 1 for convenience. This payload format uses the fields of the header in a manner consistent with that specification.
The RTP header information to be set according to this RTP payload format is set as follows:

**Marker bit (M): 1 bit**

Set for the last packet of the access unit, carried in the current RTP stream. This is in line with the normal use of the M bit in video formats to allow an efficient playout buffer handling.

When MRST or MRMT is in use, if an access unit appears in multiple RTP streams, the marker bit is set on each RTP stream’s last packet of the access unit.

**Payload Type (PT): 7 bits**

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

**NOTE: (informative) It is not required to use different payload type values for different RTP streams in MRST or MRMT.**

**Sequence Number (SN): 16 bits**

Set and used in accordance with [RFC3550]

**Timestamp (32 bits):**

The RTP timestamp is set to the sampling timestamp of the content. A 90 kHz clock rate MUST be used.
If the NAL unit has no timing properties of its own (e.g., parameter set and SEI NAL units), the RTP timestamp MUST be set to the RTP timestamp of the coded picture of the access unit in which case the NAL unit (according to Section 8.4.5.3 of [ISO.IEC.23090-5]) is included.

Receivers MUST use the RTP timestamp for the display process, even when the bitstream contains atlas frame timing SEI messages as specified in [ISO.IEC.23090-5].

Synchronization source (SSRC): 32 bits

Used to identify the source of the RTP packets.

When using SRST, by definition a single SSRC is used for all parts of a single bitstream. In MRST or MRMT, different SSRCs are used for each RTP stream containing a subset of the sub-layers of the single (temporally scalable) bitstream. A receiver is required to correctly associate the set of SSRCs that are included parts of the same bitstream.

The remaining RTP header fields are used as specified in [RFC3550].

5.3. RTP payload header

The first two bytes of the payload of an RTP packet are referred to as the payload header. The payload header consists of the same fields (F, NUT, NLI, and TID) as the NAL unit header as shown in Section 4.3.2, irrespective of the type of the payload structure. For convenience the structure of RTP payload header is described below in Figure 2.

```
  0 1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|F|    NUT    |    NLI    | TID |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 2: RTP Payload Header

F: nal_forbidden_zero_bit as specified in [ISO.IEC.23090-5] MUST be equal to 0.

NUT: nal_unit_type as specified in [ISO.IEC.23090-5] defines the type of the RBSP data structure contained in the NAL unit. NUT value could carry other meaning depending on the RTP packet type.
NLI: nal_layer_id as specified in [ISO.IEC.23090-5] defines the identifier of the layer to which an ACL NAL unit belongs or the identifier of a layer to which a non-ACL NAL unit applies.

TID: nal_temporal_id_plus1 minus 1 as specified in [ISO.IEC.23090-5] defines a temporal identifier for the NAL unit. The value of nal_temporal_id_plus1 MUST NOT be equal to 0.

5.4. Transmission modes

This memo enables transmission of an V3C atlas bitstream over:

* a Single RTP stream on a Single media Transport (SRST),
* Multiple RTP streams over a Single media Transport (MRST), or
* Multiple RTP streams on Multiple media Transports (MRMT).

When in MRST or MRMT, multiple RTP streams may be grouped together as specified in [RFC5888] and [RFC9143].

SRST or MRST SHOULD be used for point-to-point unicast scenarios, whereas MRMT SHOULD be used for point-to-multipoint multicast scenarios where different receivers require different operation points of the same V3C atlas bitstream, to improve bandwidth utilizing efficiency.

NOTE: A multicast may degrade to a unicast at some point when only one receiver has left. This is a justification of the first "SHOULD" instead of "MUST". There might be scenarios where MRMT is desirable but not possible, e.g., when IP multicast is not deployed in certain network. This is a justification of the second "SHOULD" instead of "MUST".

The transmission mode is indicated by the tx-mode media parameter. If tx-mode is equal to "SRST", SRST MUST be used. Otherwise, if tx-mode is equal to "MRST", MRST MUST be used. Otherwise (tx-mode is equal to "MRMT"), MRMT MUST be used.

NOTE: (informative) When an RTP stream does not depend on other RTP streams, any of SRST, MRST, or MRMT may be in use for the RTP stream.

Receivers MUST support all of SRST, MRST, and MRMT. The required support of MRMT by receivers does not imply that multicast must be supported by receivers.

5.5. Payload structures
5.5.1. General

Three different types of RTP packet payload structures are specified. A receiver can identify the payload structure by the first two bytes of the RTP packet payload, which co-serves as the RTP payload header. These two bytes are always structured as a NAL unit header. The NAL unit type field indicates which structure is present in the payload.

The three different payload structures are as follows:

* Single NAL Unit Packet: Contains a single NAL unit in the payload. This payload structure is specified in Section 5.5.2.

* Aggregation Packet: Contains multiple NAL units in a single RTP payload. This payload structure is specified in Section 5.5.3.

* Fragmentation Unit: Contains a subset of a single NAL unit. This payload structure is specified in Section 5.5.4.

NOTE: (informative) This specification does not limit the size of NAL units encapsulated in NAL unit packets and fragmentation units. [ISO.IEC.23090-5] does not restrict the maximum size of a NAL unit directly either. Instead a NAL unit sample stream format may be used, which provides flexibility to signal NAL unit size up to UINT64_MAX bytes.

5.5.2. Single NAL unit packet

Single NAL unit packet contains exactly one NAL unit, and consists of a RTP payload header and following conditional fields: 16-bit DONL and 16-bit v3c-tile-id. The rest of the payload data contain the NAL unit payload data (excluding the NAL unit header). Single NAL unit packet may contain atlas NAL units of the types defined in Table 4 of [ISO.IEC.23090-5]. The structure of the single NAL unit packet is shown below in Figure 3.
RTP payload header SHOULD be an exact copy of the NAL unit header of the contained NAL unit.

A NAL unit stream composed by de-packetizing single NAL unit packets in RTP sequence number order MUST conform to the NAL unit decoding order, when DONL is not present.

The DONL field, when present, specifies the value of the 16-bit decoding order number of the contained NAL unit. If sprop-max-don-diff is greater than 0 for any of the RTP streams, the DONL field MUST be present, and the variable DONL for the contained NAL unit is derived as equal to the value of the DONL field. Otherwise (sprop-max-don-diff is equal to 0 for all the RTP streams), the DONL field MUST NOT be present.

The v3c-tile-id field, when present, specifies the 16-bit tile identifier for the NAL unit, as signaled in V3C atlas tile header defined in [ISO.IEC.23090-5]. If v3c-tile-id-pres is equal to 1 and RTP payload header NUT is in range 0-35, inclusive, the v3c-tile-id field MUST be present. Otherwise, the v3c-tile-id field MUST NOT be present.

NOTE: (informative) Only values for NAL unit type (NUT) in range 0-35, inclusive, are allocated for atlas tile layer data, defined in [ISO.IEC.23090-5], which means that NAL unit types outside of the range are not specific to atlas tiles and SHOULD NOT contain v3c-tile-ids.
5.5.3. Aggregation packet

Aggregation Packets (APs) enable the reduction of packetization overhead for small NAL units, such as most of the non-ACL NAL units, which are often only a few octets in size.

Aggregation packets may wrap multiple NAL units belonging to the same access unit in a single RTP payload. The first two bytes of the AP MUST contain RTP payload header. The NAL unit type (NUT) for the NAL unit header contained in the RTP payload header MUST be equal to 56, which falls in the unspecified range of the NAL unit types defined in [ISO.IEC.23090-5]. AP may contain a conditional v3c-tile-id field. AP MUST contain two or more aggregation units. The structure of AP is shown in Figure 4.

```
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 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  RTP payload header (NUT=56)  |      v3c-tile-id (cond)       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                  Two or more aggregation units                |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Figure 4: Aggregation Packet (AP)
```

The fields in the payload header are set as follows. The F bit MUST be equal to 0 if the F bit of each aggregated NAL unit is equal to zero; otherwise, it MUST be equal to 1. The NUT field MUST be equal to 56. The value of NLI MUST be equal to the lowest value of NLI of all the aggregated NAL units. The value of TID MUST be the lowest value of TID of all the aggregated NAL units.

All ACL NAL units in an aggregation packet have the same TID value since they belong to the same access unit. However, the packet may contain non-ACL NAL units for which the TID value in the NAL unit header may be different than the TID value of the ACL NAL units in the same AP.

The v3c-tile-id field, when present, specifies the 16-bit tile identifier for all ACL NAL units in the AP. If v3c-tile-id-pres is equal to 1, the v3c-tile-id field MUST be present. Otherwise, the v3c-tile-id field MUST NOT be present.
AP MUST carry at least two aggregation units (AU) and can carry as many aggregation units as necessary. However, the total amount of data in an AP MUST fit into an IP packet, and the size SHOULD be chosen so that the resulting IP packet is smaller than the MTU size so to avoid IP layer fragmentation. The structure of the AU depends both on the presence of the decoding order number, the sequence order of the AU in the AP and the presence of v3c-tile-id field. The structure of an AU is shown in Figure 5.

```
+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------
|  DOND (cond)  |  DONL (cond)  |      v3c-tile-id (cond)       |
|               |               |                             |
+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------|
|  NALU size    |               |                             |
|               |               |                             |
+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------|
|  NAL unit     |               |                             |
|               |               |                             |
+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------|
```

Figure 5: Aggregation Unit (AU)

If sprop-max-don-diff is greater than 0 for any of the RTP streams, an AU begins with the DOND / DONL field. The first AU in the AP contains DONL field, which specifies the 16-bit value of the decoding order number of the aggregated NAL unit. The variable DON for the aggregated NAL unit is derived as equal to the value of the DONL field. All subsequent AUs in the AP MUST contain an (8-bit) DOND field, which specifies the difference between the decoding order number values of the current aggregated NAL unit and the preceding aggregated NAL unit in the same AP. The variable DON for the aggregated NAL unit is derived as equal to the DON of the preceding aggregated NAL unit in the same AP plus the value of the DOND field plus 1 modulo 65536.

When sprop-max-don-diff is equal to 0 for all the RTP streams, DOND / DONL fields MUST NOT be present in an aggregation unit. The aggregation units MUST be stored in the aggregation packet so that the decoding order of the containing NAL units is preserved. This means that the first aggregation unit in the aggregation packet SHOULD contain the NAL unit that SHOULD be decoded first.
If `v3c-tile-id-pres` is equal to 2 and the AU NAL unit header type is in range 0-35, inclusive, the 16-bit `v3c-tile-id` field MUST be present in the aggregation unit after the conditional DONL/DONL field. Otherwise `v3c-tile-id` field MUST NOT be present in the aggregation unit.

The conditional fields of the aggregation unit are followed by a 16-bit NALU size field, which provides the size of the NAL unit (in bytes) in the aggregation unit. The remainder of the data in the aggregation unit SHOULD contain the NAL unit (including the unmodified NAL unit header).

5.5.4. Fragmentation unit

Fragmentation Units (FUs) are introduced to enable fragmenting a single NAL unit into multiple RTP packets, possibly without cooperation or knowledge of the encoder. A fragment of a NAL unit consists of an integer number of consecutive octets of that NAL unit. Fragments of the same NAL unit MUST be sent in consecutive order with ascending RTP sequence numbers (with no other RTP packets within the same RTP stream being sent between the first and last fragment.

When a NAL unit is fragmented and conveyed within FUs, it is referred to as a fragmented NAL unit. Aggregation packets MUST NOT be fragmented. FUs MUST NOT be nested; i.e., an FU MUST NOT contain a subset of another FU. The RTP header timestamp of an RTP packet carrying an FU is set to the NALU-time of the fragmented NAL unit.

A FU consists of a RTP payload header with NUT equal to 58, an 8-bit FU header, a conditional 16-bit DONL field, a conditional 16-bit `v3c-tile-id` field and an FU payload. The structure of an FU is illustrated below in Figure 6.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  RTP payload header (NUT=58)  |   FU header   |  DONL (cond)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|  DONL (cond)  |    v3c-tile-id (cond)         |               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               |
|                                                               |
|                          FU payload                           |
|                                                               |
|                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                               :...OPTIONAL RTP padding        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: Fragmentation Unit
The fields in the RTP payload header are set as follows. The NUT field MUST be equal to 58. The rest of the fields MUST be equal to the fragmented NAL unit.

The FU header consists of an S bit, an E bit, and a 6-bit FUT field. The structure of FU header is illustrated below in Figure 7.

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

Figure 7: Fragmentation unit header

When set to 1, the S bit indicates the start of a fragmented NAL unit, i.e., the first byte of the FU payload is also the first byte of the payload of the fragmented NAL unit. When the FU payload is not the start of the fragmented NAL unit payload, the S bit MUST be set to 0.

When set to 1, the E bit indicates the end of a fragmented NAL unit, i.e., the last byte of the payload is also the last byte of the fragmented NAL unit. When the FU payload is not the last fragment of a fragmented NAL unit, the E bit MUST be set to 0.

The field FUT MUST be equal to the nal_unit_type field of the fragmented NAL unit.

A non-fragmented NAL unit MUST NOT be transmitted in one FU; i.e., the Start bit and End bit MUST NOT both be set to 1 in the same FU header.

The DONL field, when present, specifies the value of the 16-bit decoding order number of the fragmented NAL unit. If sprop-max-don-diff is greater than 0 for any of the RTP streams, and the S bit is equal to 1, the DONL field MUST be present in the FU, and the variable DON for the fragmented NAL unit is derived as equal to the value of the DONL field. Otherwise (sprop-max-don-diff is equal to 0 for all the RTP streams, or the S bit is equal to 0), the DONL field MUST NOT be present in the FU.

The v3c-tile-id field, when present, specifies the 16-bit tile identifier for the fragmented NAL unit. If v3c-tile-id-pres is equal to 1, FUT is in range 0-35, and the S bit is equal to 1, the v3c-tile-id field MUST be present after the conditional DONL field. Otherwise, the v3c-tile-id field MUST NOT be present.
The FU payload consists of fragments of the payload of the fragmented NAL unit so that if the FU payloads of consecutive FUs, starting with an FU with the S bit equal to 1 and ending with an FU with the E bit equal to 1, are sequentially concatenated, the payload of the fragmented NAL unit can be reconstructed.

The NAL unit header of the fragmented NAL unit is not included as such in the FU payload, but rather the information of the NAL unit header of the fragmented NAL unit is conveyed in F, NLI, and TID fields of the RTP payload headers of the FUs and the FUT field of the FU header. An FU payload MUST NOT be empty.

If an FU is lost, the receiver SHOULD discard all following fragmentation units in transmission order corresponding to the same fragmented NAL unit, unless the decoder in the receiver is known to be prepared to gracefully handle incomplete NAL units.

5.6. Decoding Order Number

For each atlas NAL unit, the variable AbsDon is derived, representing the decoding order number that is indicative of the NAL unit decoding order. Let NAL unit n be the n-th NAL unit in transmission order within an RTP stream.

If sprop-max-don-diff is equal to 0 for all the RTP streams carrying the V3C atlas bitstream, AbsDon[n], the value of AbsDon for NAL unit n, is derived as equal to n.

Otherwise (sprop-max-don-diff is greater than 0 for any of the RTP streams), AbsDon[n] is derived as follows, where DON[n] is the value of the variable DON for NAL unit n:

* If n is equal to 0 (i.e., NAL unit n is the very first NAL unit in transmission order), AbsDon[0] is set equal to DON[0].

* Otherwise (n is greater than 0), the following applies for derivation of AbsDon[n]:
  - If DON[n] == DON[n-1], AbsDon[n] = AbsDon[n-1]
If \( \text{DON}[n] < \text{DON}[n-1] \) and \( \text{DON}[n-1] - \text{DON}[n] < 32768 \), \( \text{AbsDon}[n] = \text{AbsDon}[n-1] - (\text{DON}[n-1] - \text{DON}[n]) \)

For any two NAL units \( m \) and \( n \), the following applies:

* \( \text{AbsDon}[n] \) greater than \( \text{AbsDon}[m] \) indicates that NAL unit \( n \) follows NAL unit \( m \) in NAL unit decoding order.

* When \( \text{AbsDon}[n] \) is equal to \( \text{AbsDon}[m] \), the NAL unit decoding order of the two NAL units can be in either order.

* \( \text{AbsDon}[n] \) less than \( \text{AbsDon}[m] \) indicates that NAL unit \( n \) precedes NAL unit \( m \) in decoding order.

6. Packetization and de-packetization rules

The following packetization rules apply:

* If \( \text{sprop-max-don-diff} \) is greater than 0 for any of the RTP streams, the transmission order of NAL units carried in the RTP stream MAY be different than the NAL unit decoding order and the NAL unit output order. Otherwise (\( \text{sprop-max-don-diff} \) is equal to 0 for all the RTP streams), the transmission order of NAL units carried in the RTP stream MUST be the same as the NAL unit decoding order and, when \( \text{tx-mode} \) is equal to "MRST" or "MRMT", MUST also be the same as the NAL unit output order.

* A NAL unit of a small size SHOULD be encapsulated in an aggregation packet together with one or more other NAL units in order to avoid the unnecessary packetization overhead for small NAL units. For example, non-ACL NAL units such as access unit delimiters, parameter sets, or SEI NAL units are typically small and can often be aggregated with ACL NAL units without violating MTU size constraints.

* Each non-ACL NAL unit SHOULD, when possible from an MTU size perspective, be encapsulated in an aggregation packet together with its associated ACL NAL unit, as typically a non-ACL NAL unit would be meaningless without the associated ACL NAL unit being available.

* For carrying exactly one NAL unit in an RTP packet, a single NAL unit packet MUST be used

The general concept behind de-packetization is to get the NAL units out of the RTP packets in an RTP stream and all RTP streams the RTP stream depends on, if any, and pass them to the decoder in the NAL unit decoding order.
The de-packetization process is implementation dependent. Therefore, the following de-packetization rules SHOULD be taken as an example.

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

* NAL units with NAL unit type values in the range of 0 to 55, inclusive, may be passed to the decoder. NAL-unit-like structures with NAL unit type values in the range of 55 to 63, inclusive, MUST NOT be passed to the decoder.

* When sprop-max-don-diff is equal to 0 for the received RTP stream, the NAL units carried in the RTP stream may be directly passed to the decoder in their transmission order, which is identical to their decoding order.

* When sprop-max-don-diff is greater than 0 for any of the received RTP streams, the received NAL units need to be arranged into decoding order before handing them over to the decoder.

* For further de-packetization examples, the reader is referred to Section 6 of [RFC7798].

7. Payload Examples

7.1. General

Examples describing the different payload formats is provided.

7.2. V3C fragmentation unit

This example illustrates how fragmentation unit may be used to divide one NAL unit into to RTP packets. The Figure 8 illustrates the structure of the first packet with the first part of the fragmented NAL unit.
Figure 8: First packet of fragmented NAL unit

The Figure 9 illustrates the structure of the second packet with the rest of the fragmented NAL unit.

Figure 9: Second packet of fragmented NAL unit
8. Payload Format Parameters

This section specifies the parameters that MAY be used to select optional features of the payload format and certain features or properties of the bitstream or the RTP stream. The parameters are specified here as part of the media type registration for the V3C codec. A mapping of the parameters into the Session Description Protocol (SDP) [RFC8866] is also provided for applications that use SDP. Equivalent parameters could be defined elsewhere for use with control protocols that do not use SDP.

8.1. Media Type Definition

Type name: application

Subtype name: v3c


v3c-unit-header:

provides a V3C unit header bytes defined in [ISO.IEC.23090-5]. The value contains base16 [RFC4648] (hexadecimal) representation of the 4 bytes of V3C unit header.

v3c-unit-type:

v3c-unit-type provides a V3C unit type value corresponding to vuh_unit_type defined in [ISO.IEC.23090-5], i.e. defines V3C sub-bitstream type.

v3c-vps-id:

v3c-vps-id provides a value corresponding to vuh_v3c_parameter_set_id defined in [ISO.IEC.23090-5].

v3c-atlas-id:

v3c-atlas-id provides a value corresponding to vuh_atlas_id defined in [ISO.IEC.23090-5].

v3c-attr-idx:

v3c-attr-idx:

v3c-attr-part-idx:

v3c-attr-part-idx:

v3c-map-idx:

v3c-map-idx:

v3c-aux-video-flag:

v3c-aux-video-flag:

v3c-parameter-set:

v3c-parameter-set:

v3c-tile-id:

v3c-tile-id:

v3c-tile-id-pres:

v3c-tile-id-pres:

v3c-atlas-data:

v3c-atlas-data:

v3c-common-atlas-data:

v3c-common-atlas-data:

v3c-sei:

v3c-sei:

v3c-ptl-level-idxc:

v3c-ptl-level-idxc:

v3c-ptl-tier-flag:

v3c-ptl-tier-flag:

v3c-ptl-codec-idxc:

v3c-ptl-codec-idxc:

v3c-ptl-toolset-idxc:

v3c-ptl-toolset-idxc:

v3c-ptl-rec-idxc:

v3c-ptl-rec-idxc:

tx-mode:

tx-mode and sprop-max-don-diff:
v3c-attr-idx provides a value corresponding to vuh_attribute_index
defined in [ISO.IEC.23090-5].

v3c-attr-part-idx:

v3c-attr-part-idx provides a value corresponding to
vuh_attribute_partition_index defined in [ISO.IEC.23090-5].

v3c-map-idx:

v3c-map-idx provides a value corresponding to vuh_map_index defined
in [ISO.IEC.23090-5].

v3c-aux-video-flag:

v3c-aux-video-flag provides a value corresponding to
vuh_auxiliary_video_flag defined in [ISO.IEC.23090-5].

v3c-parameter-set:

v3c-parameter-set provides V3C parameter set bytes as defined in
[ISO.IEC.23090-5]. The value contains base16 [RFC4648] (hexadecimal)
representation of the V3C parameter set bytes.

v3c-tile-id:

v3c-tile-id indicates that the RTP stream contains only portion of
the tiles in the atlas. v3c-tile-id is a comma-separated (',') list
of integer values, which indicate the v3c-tile-ids that are present
in the RTP stream.

v3c-tile-id-pres:

v3c-tile-id-pres indicates that the RTP packets contain v3c-tile-id
field.

v3c-atlas-data:

v3c-atlas-data may be used to convey any atlas data NAL units of the
V3C atlas sub bitstream for out-of-band transmission. The value is a
comma-separated (',') list of encoded representations of the atlas
NAL units as specified in [ISO.IEC.23090-5]. The NAL units SHOULD be
encoded as base16 [RFC4648] (hexadecimal) representations.

v3c-common-atlas-data:
v3c-common-atlas-data may be used to convey common atlas data NAL units of the V3C common atlas sub bitstream for out-of-band transmission. The value is a comma-separated (',') list of encoded representations of the common atlas NAL units (i.e. NAL_CASPS and NAL_CAF_IDR) as specified in [ISO.IEC.23090-5]. The NAL units SHOULD be encoded as base16 [RFC4648] (hexadecimal) representations.

v3c-sei:

v3c-sei may be used to convey SEI NAL units of V3C atlas and common atlas sub bitstreams for out-of-band transmission. The value is a comma-separated (',') list of encoded representations of SEI NAL units (i.e. NAL_PREFIX_NSEI and NAL_SUFFIX_NSEI, NAL_PREFIX_ESEI, NAL_SUFFIX_ESEI) as specified in [ISO.IEC.23090-5]. The SEI NAL units SHOULD be encoded as base16 [RFC4648] (hexadecimal) representations.

v3c-ptl-level-idc:

v3c-ptl-level-idc provides a value corresponding to ptl_level_idc defined in [ISO.IEC.23090-5].

v3c-ptl-tier-flag:

v3c-ptl-tier-flag provides a value corresponding to ptl_tier_flag defined in [ISO.IEC.23090-5].

v3c-ptl-codec-idc:

v3c-ptl-codec-idc provides a value corresponding to ptl_profile_codec_group_idc defined in [ISO.IEC.23090-5].

v3c-ptl-toolset-idc:

v3c-ptl-toolset-idc provides a value corresponding to ptl_profile_toolset_idc defined in [ISO.IEC.23090-5].

v3c-ptl-rec-idc:

v3c-ptl-rec-idc provides a value corresponding to ptl_profile_reconstruction_idc defined in [ISO.IEC.23090-5].

tx-mode:

This parameter indicates whether the transmission mode is SRST, MRST, or MRMT.
The value of tx-mode MUST be equal to "SRST", "MRST" or "MRMT". When not present, the value of tx-mode is inferred to be equal to "SRST".

If the value is equal to "MRST", MRST MUST be in use. Otherwise, if the value is equal to "MRMT", MRMT MUST be in use. Otherwise (the value is equal to "SRST"), SRST MUST be in use.

The value of tx-mode MUST be equal to "MRST" for all RTP streams in an MRST.

The value of tx-mode MUST be equal to "MRMT" for all RTP streams in an MRMT.

**sprop-max-don-diff:**

If the transmission order of NAL units in the RTP stream(s) is the same as the decoding and NAL unit output order, this parameter must be equal to 0.

Otherwise, if the decoding order of the NAL units of the RTP stream(s) is the same as the NAL unit transmission order but not the same as NAL unit output order, the value of this parameter MUST be equal to 1.

Otherwise, this parameter specifies the maximum absolute difference between the decoding order number (i.e., AbsDon) values of any two NAL units naluA and naluB, where naluA follows naluB in decoding order and precedes naluB in transmission order.

The value of sprop-max-don-diff MUST be an integer in the range of 0 to 32767, inclusive.

When not present, the value of sprop-max-don-diff is inferred to be equal to 0.

**Encoding considerations:**

This media type is framed and binary; see Section 4.8 in [RFC6838].

**Security considerations:**

Please see Section 12.

**Interoperability considerations:** N/A

**Published specification:**

Applications that use this media type: N/A
9. Congestion Control Considerations

This section is to describe the possibility to vary the bitrate as a response to congestion. Below is also a proposal for an initial text that reference RTP and profiles definition of congestion control.

Congestion control for RTP SHALL be used in accordance with [RFC3550], and with any applicable RTP profile: e.g., [RFC3551]. An additional requirement if best-effort service is being used is users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within acceptable parameters.

Circuit Breakers [RFC8083] is an update to RTP [RFC3550] that defines criteria for when one is required to stop sending RTP Packet Streams. The circuit breakers is to be implemented and followed.

10. Session Description Protocol

The mapping of above defined payload format media type is mapped to fields in the Session Description Protocol (SDP) according to [RFC8866].

10.1. Mapping of payload type parameters to SDP

10.1.1. For V3C atlas components

* The media name in the "m=" line of SDP MUST be application.
* The encoding name in the "a=rtpmap" line of SDP must be v3c
* The clock rate in the "a=rtpmap" line MUST be 90000.

An example of media representation in SDP is as follows:

```
m=application 49170 RTP/AVP 98
a=rtpmap:98 v3c/90000
a=fmtp:98 v3c-unit-header=08000000; // V3C_AD
   v3c-ptl-tier-flag=1
```

10.1.2. For V3C video components

* The media name in the "m=" line of SDP MUST be video.

* The encoding name in the "a=rtpmap" line of SDP can be any video subtype, e.g. avc, hevc, vvc etc.

* The clock rate in the "a=rtpmap" line MUST be 90000.


* The OPTIONAL parameters may include any optional parameters from the respective video payload specifications.

An example of media representation corresponding to occupancy component in SDP is as follows:

```
m=video 49170 RTP/AVP 99
a=rtpmap:99 H265/90000
a=fmtp:99 sprop-max-don-diff=0;
   v3c-unit-header=10000000
```

Ilola & Kondrad          Expires 28 October 2022               [Page 30]
When v3c-unit-header or v3c-unit-type indicate V3C unit type V3C_PVD, v3c-parameter-set, v3c-atlas-data or v3c-common-atlas-data may be signaled along the video stream. When v3c-parameter-set, v3c-atlas-data or v3c-common-atlas-data are present it indicates that the provided data is static for the whole duration of the stream.

When v3c-parameter-set, v3c-atlas-data or v3c-common-atlas-data are signaled along the video stream it is expected the respective v3c-parameter-set, v3c-atlas-data or v3c-common-atlas-data remain static for the duration of the stream.

An example of media representation in SDP is as follows:

```plaintext
m=video 49170 RTP/AVP 99
a=rtpmap:99 H265/90000
a=fmtp:99 v3c-unit-header=28000000;
v3c-parameter-set=F6F0093992;
v3c-atlas-data=ABCA,5D5A,68
```

10.2. Grouping Framework

Different V3C components can be represented by their own respective RTP streams. A grouping tool, as defined in [RFC5888], may be extended to support V3C grouping.

Group attribute with V3C type is provided to allow application to identify "m" lines that belong to the same V3C bitstream. Grouping type V3C MUST be used with the group attribute. The tokens that follow are mapped to 'mid'-values of individual media lines in the SDP.

```plaintext
a=group:V3C <tokens> <v3c specific session-level parameters>
```

The V3C grouping type attribute related v3c-specific session level parameters can include the following optional information:

```plaintext
v3c-parameter-set=<value>
v3c-atlas-data=<value>
v3c-common-atlas-data=<value>
v3c-sei=<value>
```

When signaled as a session level parameter, the data is considered to be static for the duration of the stream.

The following example shows an SDP including four media lines, three describing V3C video components and one V3C atlas component. All the media lines are grouped under one V3C group which provides the V3C parameter set.
... 

V3C group attribute type can be used as follows to indicate different V3C components and associate static atlas data with them.

... 

The following example describes how every v3c video component is packed into a single stream and associated with static atlas data.

... 

Ilola & Kondrad Expires 28 October 2022 [Page 32]
The example below describes how content with two atlases can be signaled as separate streams.

...  
a=group:V3C 1 2 3 4 5 6 7 8 v3c-parameter-set=AF6F00939921878;  
v3c-common-atlas-data=AFFA,0110;  
m=video 40000 RTP/AVP 96  
a=rtpmap:96 H264/90000  
a=fmtp:96 v3c-unit-header=10000000 // occupancy, atlas 0  
a=mid:1  
m=video 40002 RTP/AVP 97  
a=rtpmap:97 H264/90000  
a=fmtp:97 v3c-unit-header=18000000 // geometry, atlas 0  
a=mid:2  
m=video 40004 RTP/AVP 98  
a=rtpmap:98 H264/90000  
a=fmtp:98 v3c-unit-header=20000000 // attribute, atlas 0  
a=mid:3  
m=video 40008 RTP/AVP 100  
a=rtpmap:100 v3c/90000  
a=fmtp:100 v3c-unit-header=08000000; // atlas 0  
a=mid:4  
m=video 40010 RTP/AVP 101  
a=rtpmap:101 H264/90000  
a=fmtp:101 v3c-unit-header=10020000 // occupancy, atlas 1  
a=mid:5  
m=video 40012 RTP/AVP 102  
a=rtpmap:102 H264/90000  
a=fmtp:102 v3c-unit-header=18020000 // geometry, atlas 1  
a=mid:6  
m=video 40014 RTP/AVP 103  
a=rtpmap:103 H264/90000  
a=fmtp:103 v3c-unit-header=20020000 // attribute, atlas 1  
a=mid:7  
m=video 40018 RTP/AVP 104  
a=rtpmap:104 v3c/90000  
a=fmtp:104 v3c-unit-header=08020000; // V3C_AD, atlas 1  
a=mid:8

10.3. Offer/Answer Considerations  
An example of offer which only sends V3C content. The following example contains video components at three different versions.
An example of answer which only receives V3C data with the selected versions.
Internet-Draft         RTP payload format for V3C             April 2022

... 
a=group:v3c 1 2 3 4 
m=video 50000 RTP/AVP 96 
a=rtpmap:96 H264/90000 
a=recvonly 
m=video 50002 RTP/AVP 97 
a=rtpmap:97 H265/90000 
a=recvonly 
m=video 50004 RTP/AVP 98 
a=rtpmap:98 H266/90000 
a=recvonly 
m=video 50006 RTP/AVP 96 
a=rtpmap:96 v3c/90000 
a=recvonly

An example offer, which allows bundling different V3C components on one stream, based on [RFC9143].

... 
a=group:BUNDLE 1 2 3 4 
a=group:v3c 1 2 3 4 v3c-parameter-set=AF6F00939921878 
m=video 40000 RTP/AVP 96 
a=rtpmap:96 H264/90000 
a=fmtp:96 v3c-unit-type=2;v3c-vps-id=0;v3c-atlas-id=0 
a=mid:1 
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid 
m=video 40002 RTP/AVP 96 
a=rtpmap:96 H264/90000 
a=fmtp:96 v3c-unit-type=3;v3c-vps-id=0;v3c-atlas-id=0; 
a=mid:2 
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid 
m=video 40004 RTP/AVP 96 
a=rtpmap:96 H264/90000 
a=fmtp:96 v3c-unit-type=4;v3c-vps-id=0;v3c-atlas-id=0; 
a=mid:3 
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid 
m=video 40006 RTP/AVP 97 
a=rtpmap:97 v3c/90000 
a=fmtp:97 v3c-unit-type=1;v3c-vps-id=0;v3c-atlas-id=0 
a=mid:4 
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid

An example answer, which accepts bundling of different V3C components.
10.4. Declarative SDP Considerations

Placeholder

11. IANA Considerations

Placeholder

12. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. However, as "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format’s responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity, and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this Security Considerations section discusses the security impacting properties of the payload format itself.
This RTP payload format and its media decoder do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing, and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Nor does the RTP payload format contain any active content.

13. References

13.1. Normative References


13.2. Informative References


Authors’ Addresses

Lauri Ilola  
Nokia Technologies  
Werinherstrasse 91  
D-81541 Munich  
Germany  
Email: lauri.ilola@nokia.com

Lukasz Konrad  
Nokia Technologies  
Werinherstrasse 91  
D-81541 Munich  
Germany  
Email: lukasz.konrad@nokia.com