Multiplexing Scheme Updates for QUIC
draft-aboba-avtcore-rfc7983bis-01.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 May 19, 2021.

Copyright Notice

Table of Contents

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 [RFC6347], Session Traversal
Utilities for NAT (STUN) [RFC5389], 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 [I-
D.ietf-quic-transport] to be multiplexed on the same port. For peer-
to-peer operation in WebRTC scenarios as described in [WEBRTC-
QUIC][WEBRTC-QUIC-TRIAL], RTP is used to transport audio and video
and QUIC is used for data exchange, SRTP [RFC3711] is keyed using
DTLS-SRTP [RFC5764] and therefore SRTP/SRTCP [RFC3550], STUN, TURN,
DTLS [RFC6347] and QUIC need to be multiplexed on the same port.

Since new versions of QUIC are allowed to change aspects of the wire
image, there is no guarantee that future versions of QUIC beyond
version 1 will adhere to the multiplexing scheme described in this
document.

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 2.5 of [RFC5766]). [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].

As noted in [I-D.aboba-avtcore-quic-multiplexing], 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.

The first octet of QUIC long header packets fall in the range 192 to
255. Since QUIC long header packets preceed 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 demultplexed 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
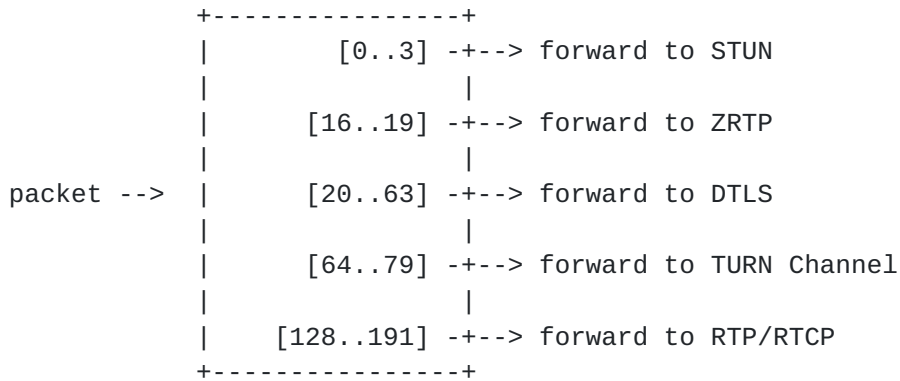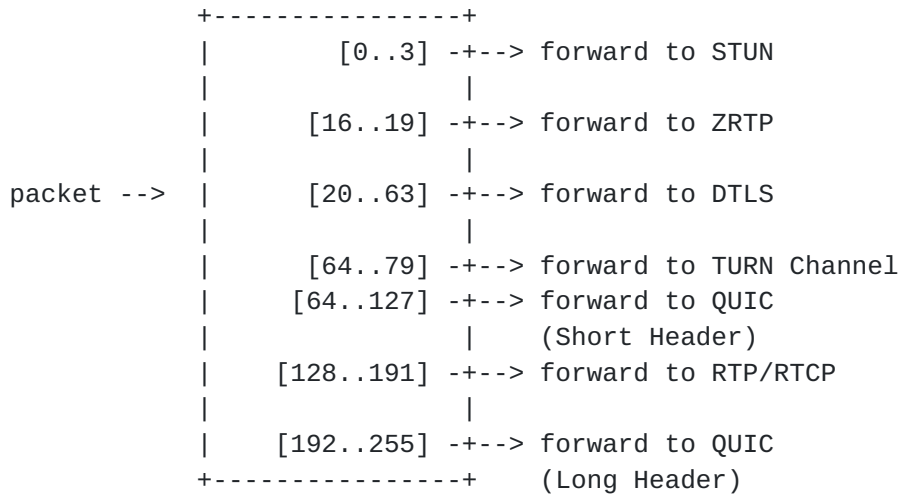dropped and an alert MAY be logged. This process is summarized in
Figure 3.

```
                    +----------------+
                    |         [0..3] -+--> forward to STUN
                    |                |
                    |        [16..19] -+--> forward to ZRTP
                    |                |
        packet -->  |        [20..63] -+--> forward to DTLS
                    |                |
                    |        [64..79] -+--> forward to TURN Channel
                    |        [64..127] -+--> forward to QUIC
                    |                |      (Short Header)
                    |       [128..191] -+--> forward to RTP/RTCP
                    |                |
                    |       [192..255] -+--> forward to QUIC
                    +----------------+      (Long Header)
```

Figure 3: The receiver's packet demultiplexing algorithm.

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.

When QUIC is used for only for data exchange, the TLS-within-QUIC
exchange [I-D.ietf-quic-tls] 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,
but if badly implemented, both transport and key derivation could be
adversely impacted.

5. IANA Considerations

   This document does not require actions by IANA.

6. References

6.1. Normative References

[I-D.ietf-quic-tls]
          Thomson, M. and S. Turner, "Using Transport Layer Security
          (TLS) to Secure QUIC", draft-ietf-quic-tls-32 (work in
          progress), October 20, 2020.

[I-D.ietf-quic-transport]
          Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed
          and Secure Transport", draft-ietf-quic-transport-32 (work
          in progress), October 20, 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
          Requirement Levels", BCP 14, RFC 2119, DOI
          10.17487/RFC2119, March 1997, <http://www.rfc-
          editor.org/info/rfc2119>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
          Jacobson, "RTP: A Transport Protocol for Real-Time
          Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July
          2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K.
          Norrman, "The Secure Real-time Transport Protocol (SRTP)",
          RFC 3711, DOI 10.17487/RFC3711, March 2004,
          <http://www.rfc-editor.org/info/rfc3711>.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing,
          "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI
          10.17487/RFC5389, October 2008, <http://www.rfc-
          editor.org/info/rfc5389>.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer
          Security (DTLS) Extension to Establish Keys for the Secure
          Real-time Transport Protocol (SRTP)", RFC 5764, DOI
          10.17487/RFC5764, May 2010, <http://www.rfc-
          editor.org/info/rfc5764>.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using
          Relays around NAT (TURN): Relay Extensions to Session
          Traversal Utilities for NAT (STUN)", RFC 5766, DOI
          10.17487/RFC5766, April 2010, <http://www.rfc-

            editor.org/info/rfc5766>.

[RFC7983]   Petit-Huguenin, M. and G. Salgueiro, "Multiplexing Scheme
             Updates for Secure Real-time Transport Protocol (SRTP)
             Extension for Datagram Transport Layer Security (DTLS)",
             RFC 7983, DOI 10.17487/RFC7983, September 2016,
             <https://www.rfc-editor.org/info/rfc7983>.

6.2. Informative References

[I-D.aboba-avtcore-quic-multiplexing]
             Aboba, B., Thatcher, P. and C. Perkins, "QUIC
             Multiplexing", draft-aboba-avtcore-quic-multiplexing-04
             (work in progress), January 28, 2020.

[RFC6189]   Zimmermann, P., Johnston, A., Ed., and J. Callas, "ZRTP:
             Media Path Key Agreement for Unicast Secure RTP", RFC 6189,
             DOI 10.17487/RFC6189, April 2011, <http://www.rfc-
             editor.org/info/rfc6189>.

[RFC6347]   Rescorla, E. and N. Modadugu, "Datagram Transport Layer
             Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347,
             January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[WEBRTC-QUIC]
             Thatcher, P. and B. Aboba, "QUIC API For Peer-to-Peer
             Connections", W3C Community Group Draft (work in progress),
             January 2020, <https://w3c.github.io/webrtc-quic>

[WEBRTC-QUIC-TRIAL]
             Hampson, S., "RTCQuicTransport Coming to an Origin Trial
             Near You (Chrome 73)", January 2019,
             <https://developers.google.com/web/updates/
             2019/01/rtcquictransport-api>

Acknowledgments

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