AVTCORE Working Group
INTERNET-DRAFT
Updates: 7983, 5764
Category: Standards Track
Expires: June 28, 2023

B. Aboba
Microsoft Corporation
G. Salgueiro
Cisco Systems
C. Perkins
University of Glasgow

29 December 2022

Multiplexing Scheme Updates for QUIC
draft-ietf-avtcore-rfc7983bis-07.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.

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 [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 be multiplexed on the same port.

The multiplexing scheme described in this document supports multiple use cases. Peer-to-peer QUIC in WebRTC scenarios, described in [P2P-QUIC] [P2P-QUIC-TRIAL], transports audio and video over SRTP, alongside QUIC, used 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/RTCP [I-D.ietf-avtcore-rtp-over-quic] STUN, TURN and QUIC need to be multiplexed on the same port.

While the scheme described in this document is compatible with QUIC version 2 [I-D.ietf-quic-v2], it is not compatible with QUIC bit greasing [RFC9287]. As a result, endpoints that wish to use multiplexing on their socket MUST NOT send the grease_quic_bit transport parameter.

1.1. 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.

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] allocates 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].

In the absence of QUIC bit greasing, the first octet of a QUIC packet (e.g. a short header packet in QUIC v1 or v2) may fall in the range 64 to 127, thereby overlapping with the allocated range for TURN channels of 64 to 79. However, in practice this overlap does not represent a problem. TURN channel packets will only be received from a TURN server to which TURN allocation and channel-binding requests have been sent. Therefore a TURN client receiving packets from the source IP address and port of a TURN server only needs to disambiguate STUN (i.e. regular TURN) packets from TURN channel packets; (S)RTP, (S)RTCP, ZRTP, DTLS or QUIC packets will not be sent from a source IP address and port that had previously responded to TURN allocation or channel-binding requests.

As a result, if the source IP address and port of a packet does not match that of a responding TURN server, a packet with a first octet of 64 to 127 can be unambiguously demultiplexed as QUIC.

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 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 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 (inclusive) or between 192 and 255 (inclusive) then the packet is QUIC. If the value is between 64 and 79 (inclusive) and the packet has a source IP address and port of a responding TURN server, then the packet is TURN channel; if the source IP address and port is not that of a responding TURN server, then the packet is QUIC.

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
                |                | (if from TURN server), else QUIC
                |     [80..127] -+--> forward to QUIC
                |                |
                |    [128..191] -+--> forward to RTP/RTCP
                |                |
                |    [192..255] -+--> forward to QUIC
                +----------------+
```

Figure 3: The receiver's packet demultiplexing algorithm.

Note: Endpoints that wish to demultiplex QUIC MUST NOT send the grease_quic_bit transport parameter, described in [RFC9287].

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 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 aversely 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

[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>.

[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>.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8489] Petit-Huguenin, M., Salgueiro, G., Rosenberg, J., Wing, D., Mahy, R. and P. Matthews, "Session Traversal Utilities for NAT (STUN)", RFC 8489, DOI 10.17487/RFC8489, February 2020, <https://www.rfc-editor.org/info/rfc8489>.

[RFC8656] Reddy, T., Johnston, A., Matthews, P. and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 8656, DOI 10.17487/RFC8656, February 2020, <https://www.rfc-editor.org/info/rfc8656>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/info/rfc9001>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/info/rfc9147>.

[RFC9287] Thomson, M., "Greasing the QUIC Bit", RFC 9287, DOI 10.17487/RFC9287, August 2022, <https://www.rfc-editor.org/info/rfc9287>.

6.2. Informative References

[I-D.ietf-avtcore-rtp-over-quic]
Ott, J. and M. Engelbart, "RTP over QUIC", draft-ietf-avtcore-rtp-over-quic (work in progress), October 24, 2022.

[I-D.ietf-quic-v2]
Duke, M., "QUIC Version 2", draft-ietf-quic-v2 (work in progress), December 15, 2022.

[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>.

[P2P-QUIC]   Thatcher, P., Aboba, B. and R. Raymond, "QUIC API For Peer-to-Peer Connections", W3C ORTC Community Group Draft (work in progress), 23 May 2021, <https://github.com/w3c/p2p-webtransport>

[P2P-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

We would like to thank Martin Thomson, Roni Even, Jonathan Lennox 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