Workgroup: QUIC
Internet-Draft: draft-duke-quic-v2-00
Published: 22 April 2021
Intended Status: Standards Track
Expires: 24 October 2021
Authors: M. Duke
F5 Networks, Inc.

QUIC Version 2

Abstract

This document specifies QUIC version 2, which is identical to QUIC version 1 except for some trivial details. Its purpose is to combat various ossification vectors and exercise the version negotiation framework. Over time, it may also serve as a vehicle for needed protocol design changes.

Discussion of this work is encouraged to happen on the QUIC IETF mailing list quic@ietf.org or on the GitHub repository which contains the draft: https://github.com/martinduke/draft-duke-quic-v2.

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 24 October 2021.

Copyright Notice

Table of Contents

1. Introduction

QUIC [QUIC-TRANSPORT] has numerous extension points, including the version number that occupies the second through fifth octets of every long header (see [I-D.ietf-quic-invariants]). If experimental versions lower in frequency, and QUIC version 1 constitutes the vast majority of QUIC traffic, there is the potential for middleboxes to ossify on the version octets always being 0x00000001.

Furthermore, version 1 Initial packets are encrypted with keys derived from a universally known salt, which allow observers to inspect the contents of these packets, which include the TLS Client Hello and Server Hello messages. Again, middleboxes may ossify on the version 1 key derivation and packet formats.

Finally [QUIC-VN] provides two mechanisms for endpoints to negotiate the QUIC version to use. The "incompatible" version negotiation method can support switching from any initial QUIC version to any other version with full generality, at the cost of an additional round-trip at the start of the connection. "Compatible" version negotiation eliminates the round-trip penalty but levies some restrictions on how much the two versions can differ semantically.

QUIC version 2 is meant to mitigate ossification concerns and exercise the version negotiation mechanisms. The only behavioral changes is that Initial packets use a different salt for key derivation. Any endpoint that supports two versions needs to implement version negotiation to protect against downgrade attacks.

This document may, over time, also serve as a vehicle for other needed changes to QUIC version 1.

[I-D.duke-quic-version-aliasing] is a more robust, but much more complicated, proposal to address these ossification problems. By design, it requires incompatible version negotiation. QUICv2 enables exercise of compatible version negotiation mechanism.

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 RFC 2119 [RFC2119].

3. Changes from QUIC Version 1

QUIC version 2 endpoints MUST implement the QUIC version 1 specification as described in [QUIC-TRANSPORT], [I-D.ietf-quic-tls], and [I-D.ietf-quic-recovery], with the following changes:

- The version field of long headers is 0x00000002.

- The salt used to derive Initial keys in Sec 5.2 of [I-D.ietf-quic-tls] changes to

```
initial_salt = 0xa707c203a59b47184a1d62ca570406ea7ae3e5d3
```

4. Version Negotiation Considerations

QUIC version 2 endpoints SHOULD also support QUIC version 1. Any QUIC endpoint that supports multiple versions MUST fully implement [QUIC-VN] to prevent version downgrade attacks.

Note that version 2 meets that document's definition of a compatible version with version 1. Therefore, v2-capable servers MUST use compatible version negotiation unless they do not support version 1.

As version 1 support is more likely than version 2 support, a client SHOULD use QUIC version 1 for its original version unless it has out-of-band knowledge that the server supports version 2.

Note that the only wire image differences between a version-1-to-2 compatible negotiation and a version 1 connection are that (1) Handshake packet headers will encode version 2, and (2) server Initial packets and client second-flight Initial packets will both encode version 2 and use keys derived from the version 2 salt.

5. Ossification Considerations

QUIC version 2 provides protection against some forms of ossification. Devices that assume that all long headers will contain encode version 1, or that the version 1 Initial key derivation

formula will remain version-invariant, will not correctly process version 2 packets.

However, many middleboxes such as firewalls focus on the first packet in a connection, which will often remain in the version 1 format due to the considerations above.

Clients interested in combating firewall ossification can initiate a connection using version 2 if they are either reasonably certain the server supports it, or are willing to suffer a round-trip penalty if they are incorrect.

6. Security Considerations

QUIC version 2 introduces no changes to the security or privacy properties of QUIC version 1.

The mandatory version negotiation mechanism guards against downgrade attacks, but downgrades have no security implications, as the version properties are identical.

7. IANA Considerations

This document requests that IANA add the following entry to the QUIC version registry:

Value: 0x00000002

Status: permanent

Specification: This Document

Change Controller: IETF

Contact: QUIC WG

8. References

8.1. Normative References

[I-D.ietf-quic-recovery] Iyengar, J. and I. Swett, "QUIC Loss Detection and Congestion Control", Work in Progress, Internet-Draft, draft-ietf-quic-recovery-34, 14 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-quic-recovery-34.txt>.

[I-D.ietf-quic-tls] Thomson, M. and S. Turner, "Using TLS to Secure QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-tls-34, 14 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-quic-tls-34.txt>.

[QUIC-TRANSPORT]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", Work in Progress, Internet-Draft, draft-ietf-quic-transport-34, 14 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-quic-transport-34.txt>.

[QUIC-VN] Schinazi, D. and E. Rescorla, "Compatible Version Negotiation for QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-version-negotiation-02, 2 November 2020, <http://www.ietf.org/internet-drafts/draft-ietf-quic-version-negotiation-02.txt>.

8.2. Informative References

[I-D.duke-quic-version-aliasing]
Duke, M., "QUIC Version Aliasing", Work in Progress, Internet-Draft, draft-duke-quic-version-aliasing-04, 30 October 2020, <http://www.ietf.org/internet-drafts/draft-duke-quic-version-aliasing-04.txt>.

[I-D.ietf-quic-invariants]
Thomson, M., "Version-Independent Properties of QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-invariants-13, 14 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-quic-invariants-13.txt>.

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

Author's Address

Martin Duke
F5 Networks, Inc.

Email: martin.h.duke@gmail.com