QUIC
Internet-Draft
Intended status: Standards Track
Expires: June 4, 2018

M. Thomson
Mozilla

December 01, 2017

Version-Independent Properties of QUIC
draft-thomson-quic-invariants-00

Abstract

This document defines the properties of the QUIC transport protocol that are expected to remain unchanged over time as new versions of the protocol are developed.

Note to Readers

Discussion of this draft takes place on the QUIC working group mailing list (quic@ietf.org), which is archived at https://mailarchive.ietf.org/arch/search/?email_list=quic [1].

Working Group information can be found at https://github.com/quicwg [2]; source code and issues list for this draft can be found at https://github.com/quicwg/base-drafts/labels/-invariants [3].

Table of Contents

1. Introduction

In addition to providing secure, multiplexed transport, QUIC [QUIC-TRANSPORT] includes the ability to negotiate a version. This allows the protocol to change over time in response to new requirements. Many characteristics of the protocol will change between versions.

This document describes the subset of QUIC that is intended to remain stable as new versions are developed and deployed.

The primary goal of this document is to ensure that it is possible deploy new versions of QUIC. By documenting the things that can't change, this document aims to preserve the ability to change any other aspect of the protocol. Thus, unless specifically described in this document, any aspect of the protocol can change between different versions.

Appendix A is a non-exhaustive list of some incorrect assumptions that might be made based on knowledge of QUIC version 1; these do not apply to every version of QUIC.

2. Conventions and Definitions

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.

3. An Extremely Abstract Description of QUIC

QUIC is a connection-oriented protocol between two endpoints. Those endpoints exchange UDP datagrams. These UDP datagrams contain QUIC packets. QUIC endpoints use QUIC packets to establish a QUIC connection, which is shared protocol state between those endpoints.

4. QUIC Packet Headers

A QUIC packet is the content of the UDP datagrams exchanged by QUIC endpoints. This document describes the contents of those datagrams.

QUIC defines two types of packet header: long and short. Long packets are identified by the most significant bit of the first octet being set; short packets have that bit cleared.

Aside from the values described here, the payload of QUIC packets is version-specific and of arbitrary length.

4.1. Long Header

Long headers take the form described in Figure 1. Bits that have version-specific semantics are marked with an X.

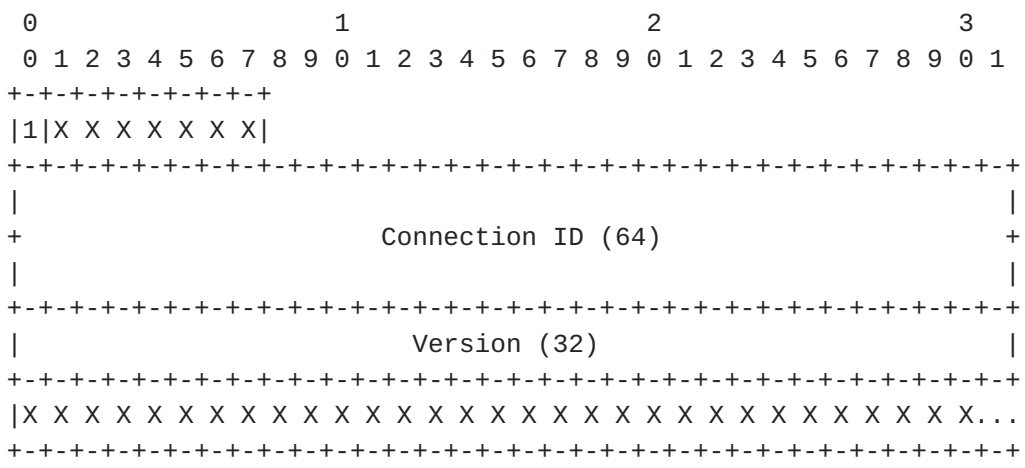

Figure 1: QUIC Long Header

A QUIC packet with a long header has the high bit of the first octet set to 1.

A QUIC packet with a long header has two fixed fields immediately following the first octet: a 64-bit Connection ID (see Section 4.3) and a 32-bit Version (see Section 4.4).

4.2. Short Header

Short headers take the form described in Figure 2. Bits that have version-specific semantics are marked with an X.

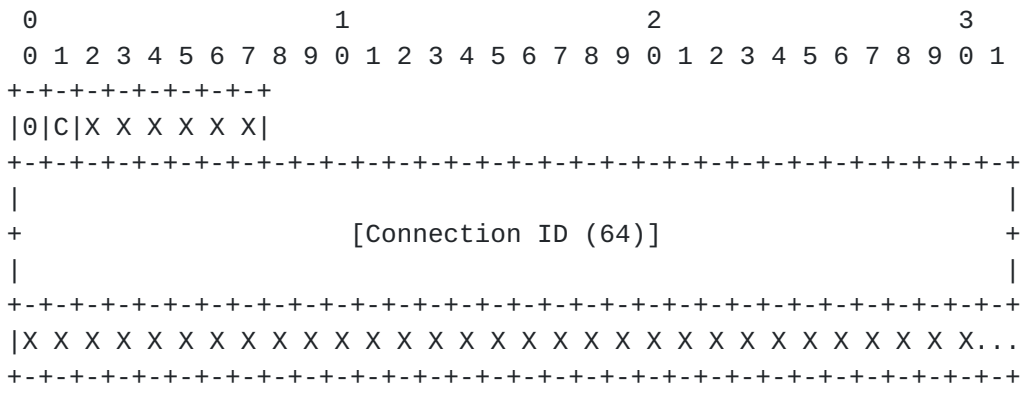

Figure 2: QUIC Short Header

A QUIC packet with a short header has the high bit of the first octet set to 0.

A QUIC packet with a short header includes an optional connection ID and no version field. The second bit of that octet (that is, 0x40) determines whether the connection ID is present. If the second bit

is cleared, a 64-bit connection ID immediately follows the first octet. If the second bit is set, the remainder of the packet has version-specific semantics.

4.3. Connection ID

A connection ID is an opaque 64-bit field.

The primary function of a connection ID is to ensure that changes in addressing at lower protocol layers (UDP, IP, and below) don't cause packets for a QUIC connection to be delivered to the wrong endpoint. The connection ID is used by endpoints and the intermediaries that support them to ensure that each QUIC packet can be delivered to the correct instance of an endpoint. At the endpoint, the connection ID is used to identify which QUIC connection the packet is intended for.

The connection ID is chosen by endpoints using version-specific methods. Packets for the same QUIC connection might use different connection ID values.

4.4. Version

QUIC versions are identified with a 32-bit integer, encoded in network byte order. Version 0 is reserved for version negotiation (see Section 5). All other version numbers are potentially valid.

5. Version Negotiation

A QUIC endpoint that receives a packet with a long header and a version it either does not understand or does not support sends a Version Negotiation packet in response. Packets with a short header do not trigger version negotiation and are always associated with an existing connection.

A Version Negotiation packet sets the high bit of the first octet, and thus it conforms with the format of a packet with a long header as defined in this document. A Version Negotiation packet is identifiable as such by the Version field, which is set to 0x00000000.

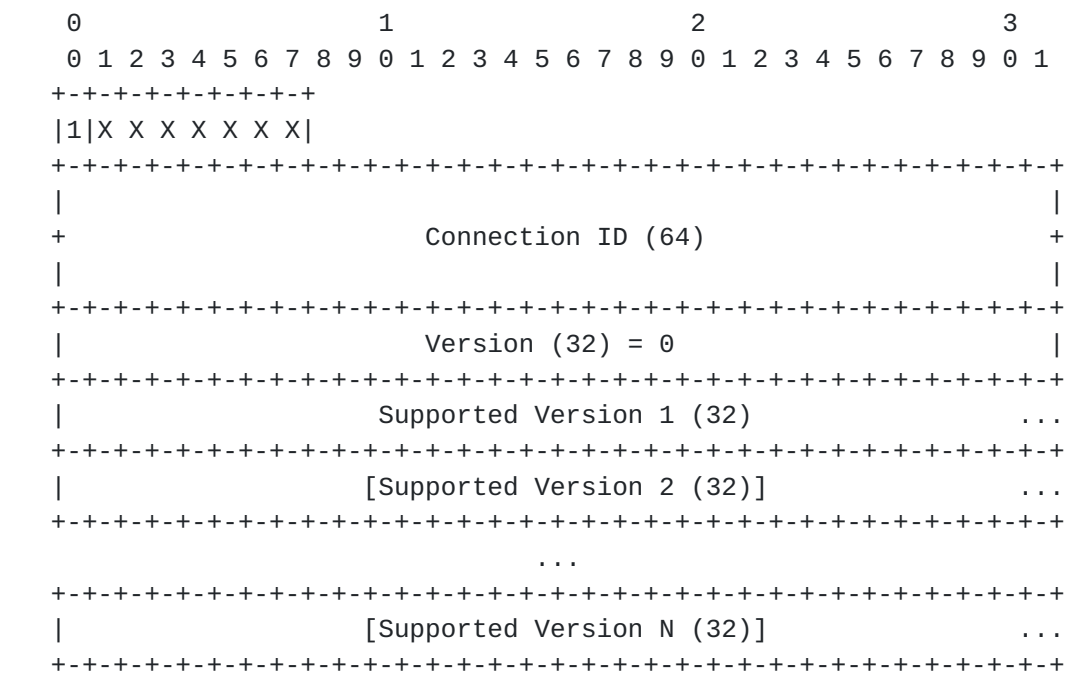

Figure 3: Version Negotiation Packet

The Version Negotiation packet contains a list of Supported Version fields, each identifying a version that the endpoint sending the packet supports. The Supported Version fields follow the Version field. A Version Negotiation packet contains no other fields. An endpoint MUST ignore a packet that contains no Supported Version fields, or a truncated Supported Version.

Version Negotiation packets do not use integrity or confidentiality protection. A specific QUIC version might authenticate the packet as part of its connection establishment process.

The Connection ID field in a Version Negotiation packet contains the Connection ID from the packet that was received. This provides some protection against injection of Version Negotiation packets by off-path attackers.

An endpoint that receives a Version Negotiation packet might change the version that it decides to use for subsequent packets. The conditions under which an endpoint changes QUIC version will depend on the version of QUIC that it chooses.

See [QUIC-TRANSPORT] for a more thorough description of how an endpoint that supports QUIC version 1 generates and consumes a Version Negotiation packet.

6. Security and Privacy Considerations

It is possible that middleboxes could use traits of a specific version of QUIC and assume that when other versions of QUIC exhibit similar traits the same underlying semantic is being expressed. There are potentially many such traits (see Appendix A). Some effort has been made to either eliminate or obscure some observable traits in QUIC version 1, but many of these remain. Other QUIC versions might make different design decisions and so exhibit different traits.

The QUIC version number does not appear in all QUIC packets, which means that reliably extracting information from a flow based on version-specific traits requires that middleboxes retain state for every connection ID they see.

The Version Negotiation packet described in this document is not integrity-protected, it only has modest protection against insertion by off-path attackers. QUIC versions MUST define a mechanism that authenticates the values it contains.

7. IANA Considerations

This document makes no request of IANA.

8. References

8.1. Normative References

[QUIC-TRANSPORT]
Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-00 (work in progress), December 2017.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

8.2. Informative References

[QUIC-TLS] Thomson, M., Ed. and S. Turner, Ed., "Using Transport Layer Security (TLS) to Secure QUIC", draft-ietf-quic-tls-00 (work in progress), December 2017.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, DOI 10.17487/RFC5116, January 2008, <https://www.rfc-editor.org/info/rfc5116>.

8.3. URIs

[1] https://mailarchive.ietf.org/arch/search/?email_list=quic

[2] https://github.com/quicwg

[3] https://github.com/quicwg/base-drafts/labels/-invariants

Appendix A. Incorrect Assumptions

There are several traits of QUIC version 1 [QUIC-TRANSPORT] that are not protected from observation, but are nonetheless considered to be changeable when a new version is deployed.

This section lists a sampling of incorrect assumptions that might be made based on knowledge of QUIC version 1. Some of these statements are not even true for QUIC version 1. This is not an exhaustive list, it is intended to be illustrative only.

The following statements are NOT guaranteed to be true for every QUIC version:

- QUIC uses TLS [QUIC-TLS] and some TLS messages are visible on the wire
- QUIC long headers are only exchanged during connection establishment
- Every flow on a given 5-tuple will include a connection establishment phase
- QUIC forbids acknowledgments of packets that only contain ACK frames, therefore the last packet before a long period of quiescence might be assumed to contain an acknowledgment
- QUIC uses an AEAD (AEAD_AES_128_GCM [RFC5116]) to protect the packets it exchanges during connection establishment
- QUIC packet numbers appear after the Version field

- QUIC packet numbers increase by one for every packet sent
- QUIC has a minimum size for the first handshake packet sent by a client
- QUIC stipulates that a client speaks first
- A QUIC Version Negotiation packet is only sent by a server
- A QUIC connection ID changes infrequently
- The same connection ID is used for packets sent by both endpoints
- A QUIC server chooses the connection ID
- QUIC endpoints change the version they speak if they are sent a Version Negotiation packet
- Only one connection at a time is established between any pair of QUIC endpoints

Author's Address

Martin Thomson
Mozilla

Email: martin.thomson@gmail.com