Manageability of the QUIC Transport Protocol
draft-ietf-quic-manageability-02

Abstract

   This document discusses manageability of the QUIC transport protocol,
   focusing on caveats impacting network operations involving QUIC
   traffic. Its intended audience is network operators, as well as
   content providers that rely on the use of QUIC-aware middleboxes,
   e.g. for load balancing.

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 January 3, 2019.

Table of Contents

1. Introduction

   QUIC [QUIC-TRANSPORT] is a new transport protocol currently under
   development in the IETF quic working group, focusing on support of
   semantics as needed for HTTP/2 [QUIC-HTTP]. Based on current
   deployment practices, QUIC is encapsulated in UDP and encrypted by
   default. The current version of QUIC integrates TLS [QUIC-TLS] to
   encrypt all payload data and most control information. Given QUIC is
   an end-to-end transport protocol, all information in the protocol
   header, even that which can be inspected, is is not meant to be

   mutable by the network, and is therefore integrity-protected to the
   extent possible.

   This document provides guidance for network operation on the
   management of QUIC traffic. This includes guidance on how to
   interpret and utilize information that is exposed by QUIC to the
   network as well as explaining requirement and assumptions that the
   QUIC protocol design takes toward the expected network treatment. It
   also discusses how common network management practices will be
   impacted by QUIC.

   Of course, network management is not a one-size-fits-all endeavour:
   practices considered necessary or even mandatory within enterprise
   networks with certain compliance requirements, for example, would be
   impermissible on other networks without those requirements. This
   document therefore does not make any specific recommendations as to
   which practices should or should not be applied; for each practice,
   it describes what is and is not possible with the QUIC transport
   protocol as defined.

   QUIC is at the moment very much a moving target. This document
   refers the state of the QUIC working group drafts as well as to
   changes under discussion, via issues and pull requests in GitHub
   current as of the time of writing.

1.1. Notational Conventions

   The words "MUST", "MUST NOT", "SHOULD", and "MAY" are used in this
   document. It's not shouting; when these words are capitalized, they
   have a special meaning as defined in [RFC2119].

2. Features of the QUIC Wire Image

   In this section, we discusses those aspects of the QUIC transport
   protocol that have an impact on the design and operation of devices
   that forward QUIC packets. Here, we are concerned primarily with
   QUIC's unencrypted wire image [WIRE-IMAGE], which we define as the
   information available in the packet header in each QUIC packet, and
   the dynamics of that information. Since QUIC is a versioned
   protocol, the wire image of the header format can also change from
   version to version. However, at least the mechanism by which a
   receiver can determine which version is used and the meaning and
   location of fields used in the version negotiation process is
   invariant [QUIC-INVARIANTS].

   This document is focused on the protocol as presently defined in
   [QUIC-TRANSPORT] and [QUIC-TLS], and will change to track those
   documents.

2.1. **QUIC Packet Header Structure**

   QUIC packets may have either a long header, or a short header. The
   first bit of the QUIC header indicates which type of header is
   present.

   The long header exposes more information. It is used during
   connection establishment, including version negotiation, server
   retry, and 0-RTT data. It contains a version number, as well as
   source and destination connection IDs for grouping packets belonging
   to the same flow. The definition and location of these fields in the
   QUIC long header are invariant for future versions of QUIC, although
   future versions of QUIC may provide additional fields in the long
   header [QUIC-INVARIANTS].

   Short headers are used after connection establishment. The only
   information they contain for inspection on the path is an optional,
   variable-length destination connection ID.

   As of draft version 13 of the QUIC transport document, the following
   information may be exposed in QUIC packet headers:

   o header type: the long header has a 7-bit packet type field
      following the Header Form bit. Header types correspond to stages
      of the handshake; see Section 4.1 of [QUIC-TRANSPORT].

   o version number: The version number is present in the long header,
      and identifies the version used for that packet. Note that during
      Version Negotiation (see Section 2.6, and Section 4.3 of
      [QUIC-TRANSPORT], the version number field has a special value
      (0x00000000) that identifies the packet as a Version Negotiation
      packet.

   o source and destination connection ID: The source and destination
      connection IDs are variable-length fields that can be used to
      identify the connection associated with a QUIC packet, for load-
      balancing and NAT rebinding purposes; see Section 4.3 and
      Section 2.4. The source connection ID corresponds to the
      destination connection ID the source would like to have on packets
      sent to it, and is only present on long packet headers. The
      destination connection ID, if present, is present on both long and
      short header packets. On long header packets, the length of the
      connection IDs is also present; on short header packets, the
      length of the destination connection ID is implicit.

   o length: the length of the remaining quic packet after the length
      field, present on long headers. This field is used to implement
      coalesced packets during the handshake (see Section 2.2).

o packet number: Every packet has an associated packet number;
   however, this packet number is encrypted, and therefore not of use
   to on-path observers. This packet number has a fixed location and
   length in long headers, and an implicit location and encrypted
   variable length in short headers.

o key phase: The Key Phase bit, present in short headers identifies
   the key used to encrypt the packet during key rotation.

2.2. Coalesced Packets

Multiple QUIC packets may be coalesced into a UDP datagram, with a
datagram carrying one or more long header packets followed by zero or
one short header packets. When packets are coalesced, the Length
fields in the long headers are used to separate QUIC packets. The
length header field is variable length and its position in the header
is also variable depending on the length of the source and
destionation connection ID. See Section 4.6 of [QUIC-TRANSPORT].

2.3. Integrity Protection of the Wire Image

As soon as the cryptographic context is established, all information
in the QUIC header, including those exposed in the packet header, is
integrity protected. Further, information that were sent and exposed
in previous packets when the cryptographic context was established
yet, e.g. for the cryptographic initial handshake itself, will be
validated later during the cryptographic handshake. Therefore,
devices on path MUST NOT change any information or bits in QUIC
packet headers, since alteration of header information will lead to a
failed integrity check at the receiver, and can even lead to
connection termination.

2.4. Connection ID and Rebinding

The connection ID in the QUIC packet headers allows routing of QUIC
packets at load balancers on other than five-tuple information,
ensuring that related flows are appropriately balanced together; and
to allow rebinding of a connection after one of the endpoint's
addresses changes - usually the client's, in the case of the HTTP
binding. Client and server negotiate connection IDs during the
handshake; typically, however, only the server will request a
connection ID for the lifetime of the connection. Connection IDs for
either endpoint may change during the lifetime of a connection, with
the new connection ID being negotiated via encrypted frames. See
Section 6.1 of [QUIC-TRANSPORT].

2.5. Packet Numbers

The packet number field is always present in the QUIC packet header;
however, it is always encrypted. The encryption key for packet
number protection on handshake packets sent before cryptographic
context establishment is specific to the QUIC version, while packet
number protection on subsequent packets uses secrets derived from the
end-to-end cryptographic context. Packet numbers are therefore not
part of the wire image that is useful to on-path observers.

2.6. Version Negotiation and Greasing

Version negotiation is not protected, given the used protection
mechanism can change with the version. However, the choices provided
in the list of version in the Version Negotiation packet will be
validated as soon as the cryptographic context has been established.
Therefore any manipulation of this list will be detected and will
cause the endpoints to terminate the connection.

Also note that the list of versions in the Version Negotiation packet
may contain reserved versions. This mechanism is used to avoid
ossification in the implementation on the selection mechanism.
Further, a client may send a Initial Client packet with a reserved
version number to trigger version negotiation. In the Version
Negotiation packet the connection ID and packet number of the Client
Initial packet are reflected to provide a proof of return-
routability. Therefore changing these information will also cause
the connection to fail.

3. Network-visible information about QUIC flows

This section addresses the different kinds of observations and
inferences that can be made about QUIC flows by a passive observer in
the network based on the wire image in Section 2. Here we assume a
bidirectional observer (one that can see packets in both directions
in the sequence in which they are carried on the wire) unless noted.

3.1. Identifying QUIC traffic

The QUIC wire image is not specifically designed to be
distinguishable from other UDP traffic.

The only application binding currently defined for QUIC is HTTP
[QUIC-HTTP]. HTTP over QUIC uses UDP port 443 by default, although
URLs referring to resources available over HTTP over QUIC may specify
alternate port numbers. Simple assumptions about whether a given
flow is using QUIC based upon a UDP port number may therefore not
hold; see also [RFC7605] section 5.

3.1.1. Identifying Negotiated Version

   An in-network observer assuming that a set of packets belongs to a
   QUIC flow can infer the version number in use by observing the
   handshake: an Initial packet with a given version from a client to
   which a server responds with an Initial packet with the same version
   implies acceptance of that version.

   Negotiated version cannot be identified for flows for which a
   handshake is not observed, such as in the case of NAT rebinding;
   however, these flows can be associated with flows for which a version
   has been identified; see Section 3.4.

   In the rest of this section, we discuss only packets belonging to
   Version 1 QUIC flows, and assume that these packets have been
   identified as such through the observation of a version negotiation.

3.1.2. Rejection of Garbage Traffic

   A related question is whether a first packet of a given flow on known
   QUIC-associated port is a valid QUIC packet, in order to support in-
   network filtering of garbage UDP packets (reflection attacks, random
   backscatter). While heuristics based on the first byte of the packet
   (packet type) could be used to separate valid from invalid first
   packet types, the deployment of such heuristics is not recommended,
   as packet types may have different meanings in future versions of the
   protocol.

3.2. Connection confirmation

   Connection establishment uses Initial, Handshake, and Retry packets
   containing a TLS handshake on Stream 0. Connection establishment can
   therefore be detected using heuristics similar to those used to
   detect TLS over TCP. A client using 0-RTT connection may also send
   data packets in 0-RTT Protected packets directly after the Initial
   packet containing the TLS Client Hello. Since these packets may be
   reordered in the network, note that 0-RTT Protected data packets may
   be seen before the Initial packet. Note that only clients send
   Initial packets, so the sides of a connection can be distinguished by
   QUIC packet type in the handshake.

3.3. Application Identification

   The cleartext TLS handshake may contain Server Name Indication (SNI)
   [RFC6066], by which the client reveals the name of the server it
   intends to connect to, in order to allow the server to present a
   certificate based on that name. It may also contain information from
   Application-Layer Protocol Negotiation (ALPN) [RFC7301], by which the

client exposes the names of application-layer protocols it supports;
an observer can deduce that one of those protocols will be used if
the connection continues.

Work is currently underway in the TLS working group to encrypt the
SNI in TLS 1.3 [TLS-ENCRYPT-SNI], reducing the information available
in the SNI to the name of a fronting service, which can generally be
identified by the IP address of the server anyway. If used with
QUIC, this would make SNI-based application identification impossible
through passive measurement.

3.4. Flow association

The QUIC Connection ID (see Section 2.4) is designed to allow an on-
path device such as a load-balancer to associate two flows as
identified by five-tuple when the address and port of one of the
endpoints changes; e.g. due to NAT rebinding or server IP address
migration. An observer keeping flow state can associate a connection
ID with a given flow, and can associate a known flow with a new flow
when when observing a packet sharing a connection ID and one endpoint
address (IP address and port) with the known flow.

The connection ID to be used for a long-running flow is chosen by the
server (see [QUIC-TRANSPORT] section 5.6) during the handshake. This
value should be treated as opaque; see Section 4.3 for caveats
regarding connection ID selection at servers.

3.5. Flow teardown

The QUIC does not expose the end of a connection; the only indication
to on-path devices that a flow has ended is that packets are no
longer observed. Stateful devices on path such as NATs and firewalls
must therefore use idle timeouts to determine when to drop state for
QUIC flows.

Changes to this behavior have been discussed in the working group,
but there is no current proposal to implement these changes: see
https://github.com/quicwg/base-drafts/issues/602.

3.6. Round-trip time measurement

Round-trip time of QUIC flows can be inferred by observation once per
flow, during the handshake, as in passive TCP measurement; this
requires parsing of the QUIC packet header and the cleartext TLS
handshake on stream 0.

In the common case, the delay between the Initial packet containing
the TLS Client Hello and the Handshake packet containing the TLS

Server Hello represents the RTT component on the path between the
observer and the server. The delay between the TLS Server Hello and
the Handshake packet containing the TLS Finished message sent by the
client represents the RTT component on the path between the observer
and the client. While the client may send 0-RTT Protected packets
after the Initial packet during 0-RTT connection re-establishment,
these can be ignored for RTT measurement purposes.

Handshake RTT can be measured by adding the client-to-observer and
observer-to-server RTT components together. This measurement
necessarily includes any transport and application layer delay at
both sides.

The spin bit experiment, detailed in [QUIC-SPIN], provides an
additional method to measure intraflow per-flow RTT. When a QUIC
flow is sending at full rate (i.e., neither application nor flow
control limited), the latency spin bit described in that document
changes value once per round-trip time (RTT). An on-path observer
can observe the time difference between edges in the spin bit signal
in a single direction to measure one sample of end-to-end RTT. Note
that this measurement, as with passive RTT measurement for TCP,
includes any transport protocol delay (e.g., delayed sending of
acknowledgements) and/or application layer delay (e.g., waiting for a
request to complete). It therefore provides devices on path a good
instantaneous estimate of the RTT as experienced by the application.
A simple linear smoothing or moving minimum filter can be applied to
the stream of RTT information to get a more stable estimate.

An on-path observer that can see traffic in both directions (from
client to server and from server to client) can also use the spin bit
to measure "upstream" and "downstream" component RTT; i.e, the
component of the end-to-end RTT attributable to the paths between the
observer and the server and the observer and the client,
respectively. It does this by measuring the delay between a spin
edge observed in the upstream direction and that observed in the
downstream direction, and vice versa.

Application-limited and flow-control-limited senders can have
application and transport layer delay, respectively, that are much
greater than network RTT. Therefore, the spin bit provides network
latency information only when the sender is neither application nor
flow control limited. When the sender is application-limited by
periodic application traffic, where that period is longer than the
RTT, measuring the spin bit provides information about the
application period, not the RTT. Simple heuristics based on the
observed data rate per flow or changes in the RTT series can be used
to reject bad RTT samples due to application or flow control
limitation.

Since the spin bit logic at each endpoint considers only samples on
packets that advance the largest packet number seen, signal
generation itself is resistant to reordering. However, reordering
can cause problems at an observer by causing spurious edge detection
and therefore low RTT estimates, if reordering occurs across a spin
bit flip in the stream. This can be probabilistically mitigated by
the observer also tracking the low-order bits of the packet number,
and rejecting edges that appear out-of-order [RFC4737].

3.7. Flow symmetry measurement

QUIC explicitly exposes which side of a connection is a client and
which side is a server during the handshake. In addition, the
symmerty of a flow (whether primarily client-to-server, primarily
server-to-client, or roughly bidirectional, as input to basic traffic
classification techniques) can be inferred through the measurement of
data rate in each direction. While QUIC traffic is protected and
ACKS may be padded, padding is not required.

4. Specific Network Management Tasks

In this section, we address specific network management and
measurement techniques and how QUIC's design impacts them.

4.1. Stateful treatment of QUIC traffic

Stateful treatment of QUIC traffic is possible through QUIC traffic
and version identification (Section 3.1) and observation of the
handshake for connection confirmation (Section 3.2). The lack of any
visible end-of-flow signal (Section 3.5) means that this state must
be purged either through timers or through least-recently-used
eviction, depending on application requirements.

4.2. Passive network performance measurement and troubleshooting

Limited RTT measurement is possible by passive observation of QUIC
traffic; see Section 3.6. No passive measurement of loss is possible
with the present wire image. Extremely limited observation of
upstream congestion may be possible via the observation of CE
markings on ECN-enabled QUIC traffic.

4.3. Server cooperation with load balancers

In the case of content distribution networking architectures
including load balancers, the connection ID provides a way for the
server to signal information about the desired treatment of a flow to
the load balancers.

Server-generated Connection IDs must not encode any information other
that that needed to route packets to the appropriate backend
server(s): typically the identity of the backend server or pool of
servers, if the data-center's load balancing system keeps "local"
state of all flows itself. Care must be exercised to ensure that the
information encoded in the Connection ID is not sufficient to
identify unique end users. Note that by encoding routing information
in the Connection ID, load balancers open up a new attack vector that
allows bad actors to direct traffic at a specific backend server or
pool. It is therefore recommended that Server-Generated Connection
ID includes a cryptographic MAC that the load balancer pool server
are able to identify and discard packets featuring an invalid MAC.

4.4. DDoS Detection and Mitigation

Current practices in detection and mitigation of Distributed Denial
of Service (DDoS) attacks generally involve passive measurement using
network flow data [RFC7011], classification of traffic into "good"
(productive) and "bad" (DoS) flows, and filtering of these bad flows
in a "scrubbing" environment. Key to successful DDoS mitigation is
efficient classification of this traffic.

Limited first-packet garbage detection as in Section 3.1.2 and
stateful tracking of QUIC traffic as in Section 4.1 above can be used
in this classification step.

Note that the use of a connection ID to support connection migration
renders 5-tuple based filtering insufficient, and requires more state
to be maintained by DDoS defense systems, and linkability resistance
in connection ID update mechanisms means that a connection ID aware
DDoS defense system must have the same information about flows as the
load balancer.

However, it is questionable if connection migrations needs to be
supported in a DDOS attack. If the connection migration is not
visible to the network that performs the DDoS detection, an active,
migrated QUIC connection may be blocked by such a system under
attack. However, a defense system might simply rely on the fast
resumption mechanism provided by QUIC.

4.5. QoS support and ECMP

[EDITOR'S NOTE: this is a bit speculative; keep?]

QUIC does not provide any additional information on requirements on
Quality of Service (QoS) provided from the network. QUIC assumes
that all packets with the same 5-tuple {dest addr, source addr,
protocol, dest port, source port} will receive similar network

treatment. That means all stream that are multiplexed over the same
QUIC connection require the same network treatment and are handled by
the same congestion controller. If differential network treatment is
desired, multiple QUIC connections to the same server might be used,
given that establishing a new connection using 0-RTT support is cheap
and fast.

QoS mechanisms in the network MAY also use the connection ID for
service differentiation, as a change of connection ID is bound to a
change of address which anyway is likely to lead to a re-route on a
different path with different network characteristics.

Given that QUIC is more tolerant of packet re-ordering than TCP (see
Section 2.5), Equal-cost multi-path routing (ECMP) does not
necessarily need to be flow based. However, 5-tuple (plus eventually
connection ID if present) matching is still beneficial for QoS given
all packets are handled by the same congestion controller.

5. IANA Considerations

This document has no actions for IANA.

6. Security Considerations

Supporting manageability of QUIC traffic inherently involves
tradeoffs with the confidentiality of QUIC's control information;
this entire document is therefore security-relevant.

Some of the properties of the QUIC header used in network management
are irrelevant to application-layer protocol operation and/or user
privacy. For example, packet number exposure (and echo, as proposed
in this document), as well as connection establishment exposure for
1-RTT establishment, make no additional information about user
traffic available to devices on path.

At the other extreme, supporting current traffic classification
methods that operate through the deep packet inspection (DPI) of
application-layer headers are directly antithetical to QUIC's goal to
provide confidentiality to its application-layer protocol(s); in
these cases, alternatives must be found.

7. Contributors

Dan Druta contributed text to Section 4.4. Igor Lubashev contributed
text to Section 4.3 on the use of the connection ID for load
balancing. Marcus Ilhar contributed text to Section 3.6 on the use
of the spin bit.

8. Acknowledgments

9. References

9.1. 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, <https://www.rfc-editor.org/info/
              rfc2119>.

9.2. Informative References

   [Ding2015]
              Ding, H. and M. Rabinovich, "TCP Stretch Acknowledgments
              and Timestamps - Findings and Impliciations for Passive
              RTT Measurement (ACM Computer Communication Review)", July
              2015, <http://www.sigcomm.org/sites/default/files/ccr/
              papers/2015/July/0000000-0000002.pdf>.

   [IPIM]   Allman, M., Beverly, R., and B. Trammell, "In-Protocol
              Internet Measurement (arXiv preprint 1612.02902)",
              December 2016, <https://arxiv.org/abs/1612.02902>.

   [QUIC-HTTP]
              Bishop, M., "Hypertext Transfer Protocol (HTTP) over
              QUIC", draft-ietf-quic-http-13 (work in progress), June
              2018.

   [QUIC-INVARIANTS]
              Thomson, M., "Version-Independent Properties of QUIC",
              draft-ietf-quic-invariants-01 (work in progress), March
              2018.

   [QUIC-SPIN]
              Trammell, B. and M. Kuehlewind, "The QUIC Latency Spin
              Bit", draft-ietf-quic-spin-exp-00 (work in progress),
              April 2018.

[QUIC-TLS]
          Thomson, M. and S. Turner, "Using Transport Layer Security
          (TLS) to Secure QUIC", draft-ietf-quic-tls-13 (work in
          progress), June 2018.

[QUIC-TRANSPORT]
          Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed
          and Secure Transport", draft-ietf-quic-transport-13 (work
          in progress), June 2018.

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov,
           S., and J. Perser, "Packet Reordering Metrics", RFC 4737,
           DOI 10.17487/RFC4737, November 2006, <https://www.rfc-
           editor.org/info/rfc4737>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS)
           Extensions: Extension Definitions", RFC 6066, DOI
           10.17487/RFC6066, January 2011, <https://www.rfc-
           editor.org/info/rfc6066>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken,
           "Specification of the IP Flow Information Export (IPFIX)
           Protocol for the Exchange of Flow Information", STD 77,
           RFC 7011, DOI 10.17487/RFC7011, September 2013,
           <https://www.rfc-editor.org/info/rfc7011>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan,
           "Transport Layer Security (TLS) Application-Layer Protocol
           Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301,
           July 2014, <https://www.rfc-editor.org/info/rfc7301>.

[RFC7605] Touch, J., "Recommendations on Using Assigned Transport
           Port Numbers", BCP 165, RFC 7605, DOI 10.17487/RFC7605,
           August 2015, <https://www.rfc-editor.org/info/rfc7605>.

[TLS-ENCRYPT-SNI]
          Huitema, C. and E. Rescorla, "Issues and Requirements for
          SNI Encryption in TLS", draft-ietf-tls-sni-encryption-03
          (work in progress), May 2018.

[WIRE-IMAGE]
          Trammell, B. and M. Kuehlewind, "The Wire Image of a
          Network Protocol", draft-trammell-wire-image-04 (work in
          progress), April 2018.

Authors' Addresses

   Mirja Kuehlewind
   ETH Zurich
   Gloriastrasse 35
   8092 Zurich
   Switzerland

   Email: mirja.kuehlewind@tik.ee.ethz.ch

   Brian Trammell
   ETH Zurich
   Gloriastrasse 35
   8092 Zurich
   Switzerland

   Email: ietf@trammell.ch