QUIC TRANSPORT LAYER PROTOCOL

Athanasios Trilivas,

Sr IP SECURITY ENGINEER

AGENDA

- Introduction $ History
- QUIC Explained
- Summary
- References

INTRODUCTION & HISTORICAL DETAILS

Why, when and how it all came together, forming a new protocol

Introduction & History

- QUIC stands for “QUIC UDP INTERNET CONNECTIONS”
- Development started on 2012 by Google and after a long trial and revisioning period it was finally released to IETF.
- Which after having completed all changes finally standardized it in 2021 via RFC 9000.
- It was the reason behind HTTP/3 development so soon after HTTP/2 was standardized in 2015.
- It is widely supported by all web servers and most browsers like Chrome, Edge, Firefox and can be enabled on Safari.

Introduction & History

- Google actually started developing because TCP was showing clear signs of “ossification”
- Which roughly translates to inability to adopt further advancements without adding to complexity or incurring penalties in protocol operation.
- Indeed TCP advancements such as MPTCP and TCP Fast Open although provided similar performance improvements to QUIC suffer from middlebox* mishandling.
- So, QUIC’s first objective was not to impose any need for immediate hardware replacement, rather make use of current resources transparently while reducing latency and providing security.

* “middleboxes” are the devices effectively transporting the data from source to destination. This of course includes routers, firewalls, load balancers as well as any kind of virtualized network service residing in the Cloud. They are known to manipulate TCP headers under various circumstances.

Introduction & History

- QUIC is not a competitor to TCP but a rather successful evolvement in a long line of possible successors that never really caught on.
- Google's initial experimentation with QUIC as well as release to the public after adopting it in its own technologies (Youtube, Chrome and the Android platform), provided fertile ground for its uneventful expansion.
- QUIC accounts at the moment for 7,9% of web traffic and more than 30% of the traffic within the Datacenters **(Fig 1).**
- Industry leaders such as Microsoft, Meta, Amazon, AKAMAI and Cloudflare have also adopted it with others joining in the near future.
- Since protocol design criteria demanded full compatibility with existing infrastructure, it intrinsically employs TLSv1.3. Therefore HTTP/2 can't be used as an overlying L7 protocol and HTTP/3 had to be introduced **(Fig 2).**

Introduction & History

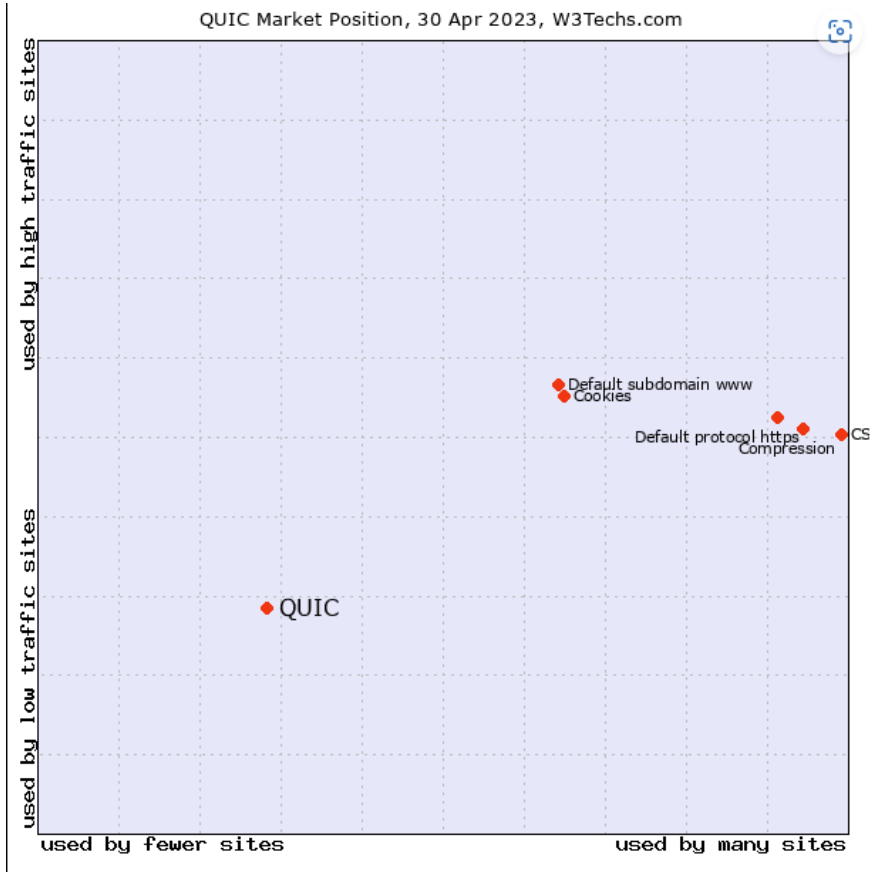

Fig. 1 QUIC web traffic share

- QUIC accounts at the moment for roughly 8% of web traffic and more than 30% of the traffic within the Datacenters.
- Fig.1 on the left represents web traffic only; Datacenter traffic metrics are for now a closely kept secret for most except Google.
- Use of TCP will continue for decades but the gap with QUIC or newer protocols will be shortened.

Introduction & History

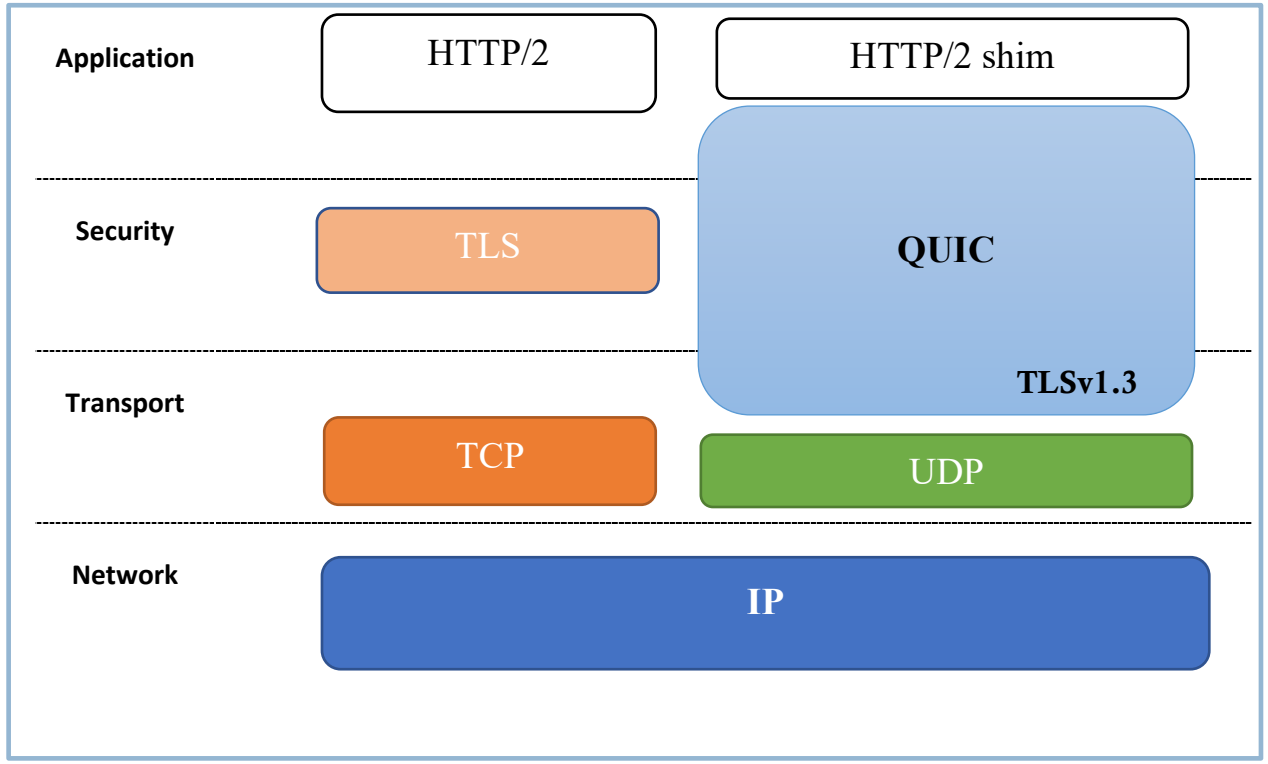

Fig. 2. QUIC over the traditional HTTPS stack

QUIC Explained

QUIC’s distinguishing characteristics

QUIC Explained

- QUIC was developed as a transport layer user protocol with UDP as the underlying transport mechanism. Using UDP allows QUIC packets to pass through intermediate devices without alteration of L4 header data.
- QUIC is cryptographic: packets are authenticated and encrypted preventing them from being altered and limiting protocol wear and tear by intermediate devices **(Fig 3)**.
- Introduces Connection ID (CID) in order allow for fast and troublesome handover amongst transmission (L1) technologies (Connection Migration).

QUIC Explained

- QUIC is reliable like TCP, and can incorporate all the best practices, and latest algorithms currently mused by TCP. However, QUIC does not implement a specific congestion control algorithm allowing endpoints to choose which algorithm to use, like CUBIC. It also mitigates loss recovery by using unique packet numbers to avoid retransmission ambiguity and by using explicit signaling in acknowledgements (ACKs) for accurate RTT measurements.
- QUIC fixes packet errors thus reducing retransmissions and delays (using FEC – Forward Error Correction). Furthermore, any mechanism of FEC can be adopted, encrypted or not.

QUIC Explained

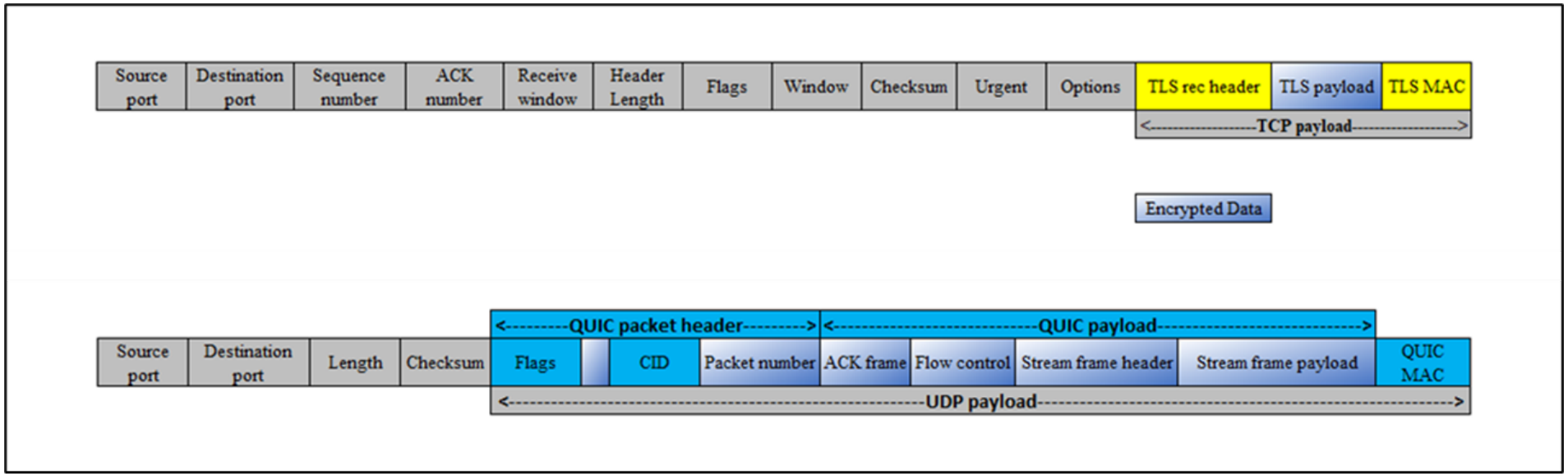

Figure 3. Unlike TCP combined with TLS, QUIC also encrypts transport layer fields

QUIC Explained

- **Minimized Handshake:** When used in combination with TLS to provide security, TCP wastes time setting up the connection **(Fig. 4)**. QUIC completes both the three-way and the cryptographic handshake in one RTT thus minimizing connection setup time.
  This feature is called "*0-RTT connection setup*"

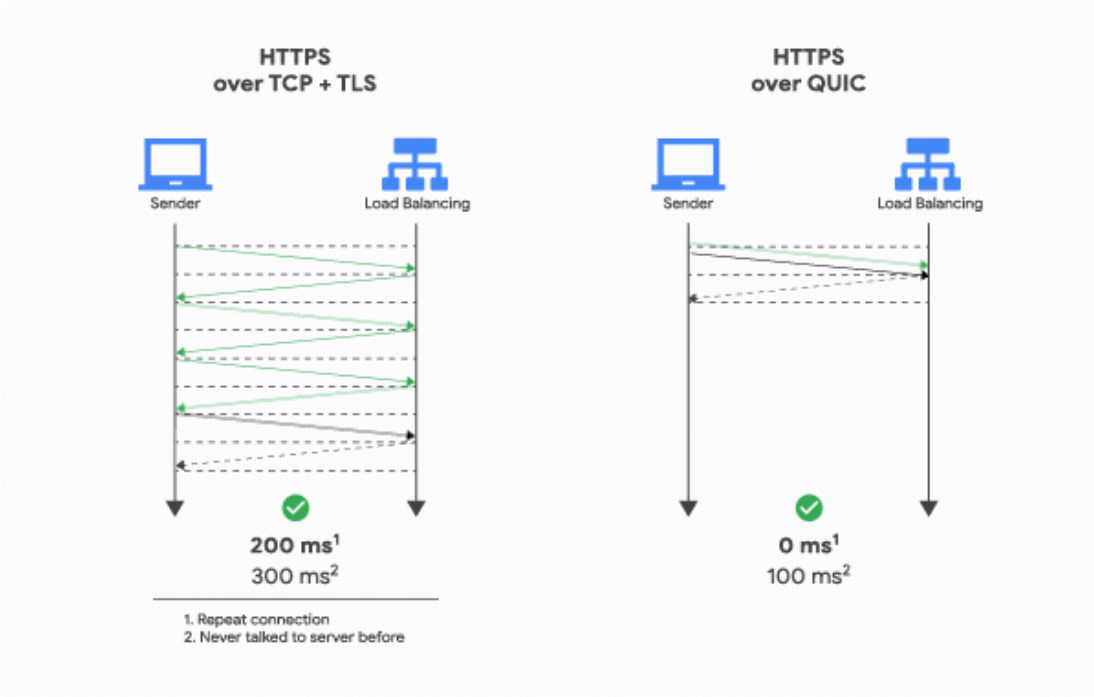

Fig 4. TCP and TLS versus QUIC connection set-up
Source: Google

QUIC Explained

- QUIC eliminates head-of-queue* blocking by using lightweight abstraction over the data structure—streams—that are multiplexed over a single connection so that the loss of a packet only affects the stream carrying the lost packet. In Fig. 5 on the right QUIC streams can be considered distinct.

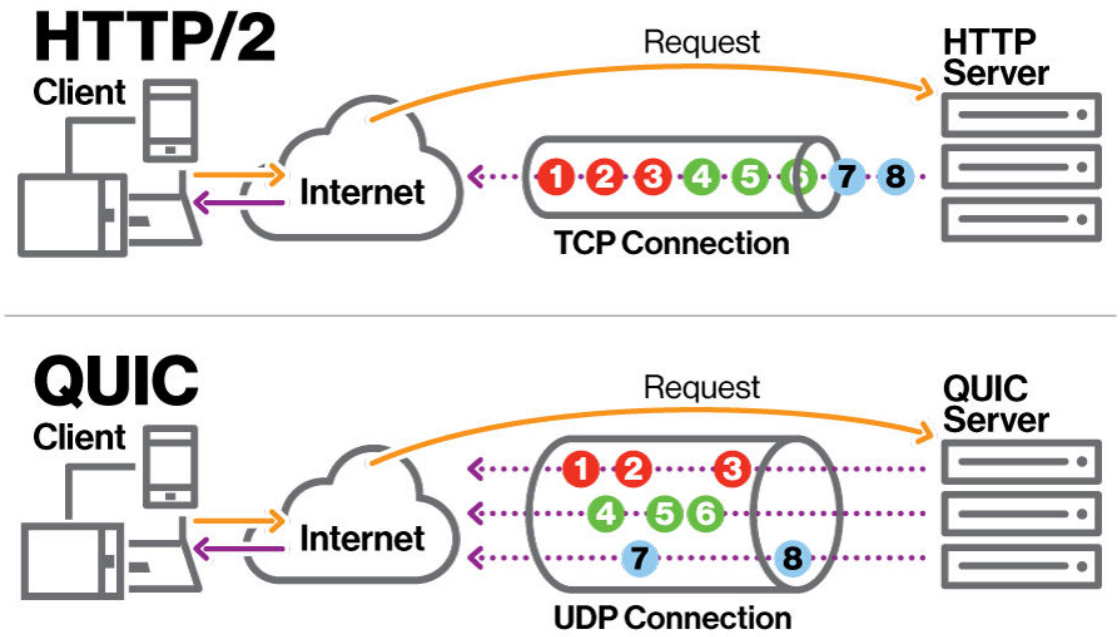

Fig 5. TCP (HTTP/2) versus QUIC (HTTP/3)

Source: Devopedia

* head-of-queue blocking takes place due to an HTTP/2 functionality where byte streams of different data sources are multiplexed. The loss of one packet however in the transmission leads to all byte streams being delayed until a successful retransmission occurs.

QUIC Explained

- **Connection migration** is a mechanism actually in use today by Android and Apple smartphones. For example, QUIC allows for a client to switch from the cellular network to a close-by Wi-Fi network and vice versa no matter the fact that the IP address changes.
- TCP on the other hand will drop the connection if the host device changes its IP address. Either TCP or the supported application will force connection re-establishment leading to great latencies introduced and a waste of bandwidth by resending data.
- QUIC mitigates this by using the CID (Connection Identifier) which is a number assigned to the 4-tuple of source and destination IP addresses and ports (Fig. 6)
- Since using a CID raises privacy issues, the CID can actually be changed to avoid tracking. The CID is encrypted with the payload and in case it changes, original and subsequent CID are kept by the TELCO.

QUIC Explained

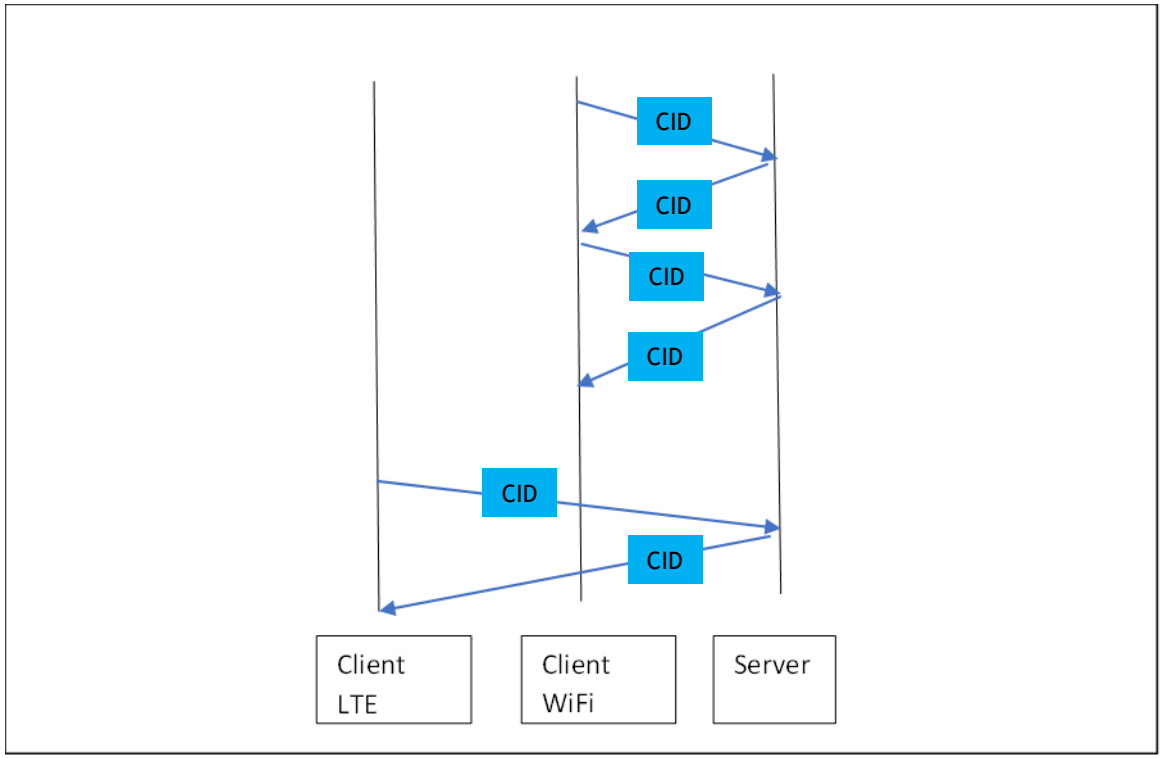

Figure 5. The use of CID numbering by QUIC to preserve connection migration; Client switches seamlessly to LTE (4G) from WiFi while moving outside WiFi's coverage area.

Conclusion

Summarization of advantages and drawbacks

Conclusion

In short, QUIC comes with the following advantages:

- Speed: QUIC reduces the number of RTTs that are required to establish a connection between a client and a server by combining both the cryptographic and transport handshakes.
- Security: QUIC is fully encrypted transport, except for the UDP header. This encryption prevents modification from middleboxes.
- Multiplexing: Applications commonly multiplex units of data within TCP's single byte-stream abstraction. To avoid head-of-line blocking because of TCP's sequential delivery, QUIC supports multiple streams within a connection.

Conclusion

- Connection migration: QUIC's connection ID enables connections to persist no matter the changes to the client's IP address and port. Such changes can be caused for example, switching from WiFi to cellular as the user moves outside the premises.
- Congestion control & Error Control:

Conclusion

A couple of disadvantages are plaguing QUIC however:

- Security (again): QUIC blocks visibility to packet payloads which is a nightmare for security specialists; Indeed currently there is no reverse proxy that can decrypt network traffic to determine whether it is carrying malware to the end device or it is leaking company data towards the Internet. In extreme cases where security is of importance, QUIC should be disabled by blocking port 443/udp.
- It remains still an ARQ protocol (Automatic Repeat Request) even though a more efficient one; That translates to time delays for individual streams at least. Newer under development protocols will be using **Erasure Codes.** This technique uses linear algebra in order to intermingle data in packets so that if one packet is lost, the information can be recovered without retransmission. What is more, this is performed without enlarging the transported data. QUIC employs a Forward Error Correction mechanism but retransmissions are still necessary.

References

References

- RFC 9000 - QUIC: A UDP-Based Multiplexed and Secure Transport (ietf.org)
- RFC 9001 - Using TLS to Secure QUIC (ietf.org)
- RFC 9002 - QUIC Loss Detection and Congestion Control (ietf.org)
- RFC 9114 - HTTP/3 (ietf.org)
- Applicability of the QUIC Transport Protocol (quicwg.org)
- Get a head start with QUIC (cloudflare.com)
- QUIC, a multiplexed transport over UDP (chromium.org)
- Usage Statistics of QUIC for Websites, April 2023 (w3techs.com)
- RFC 9170 - Long-Term Viability of Protocol Extension Mechanisms (ietf.org)
- slides-99-nwcrg-08-swett-quic-fec-00 (ietf.org)