Quic Protocol IETF's TCP - SSL - HTTP Replacement

Features 3x latency improvement, rapid mobile reconnects, 10x firewall session reduction

RFC 9000 Internet Engineering Task Force (IETF) has readied an update to Internet's protocols delivering a scorching 3x performance with rapid mobile reconnect features. A weekend experiment by Google employee Jim Roskind advanced over 12 years through the Internet's standards process. Google, YouTube, Gmail, Facebook, Microsoft and others have deployed it on their web servers. All major O/S and browsers have enabled QUIC. When QUIC is disabled by firewalls, it falls back to use the serial discrete TCP, SSL & HTTP connection methods and versions.

Latency is the problem – not even money can change the speed of light. TCP/IP's stalwart TCP Transmission Control Protocol's sliding window of bytes can overcome effects of latency, but not eliminate its affect on the three-way sys-syn-ack handshake, and that of SSL and HTTP connection latency until QUIC combined them all into one latency reducing handshake features termed 0-rtt.

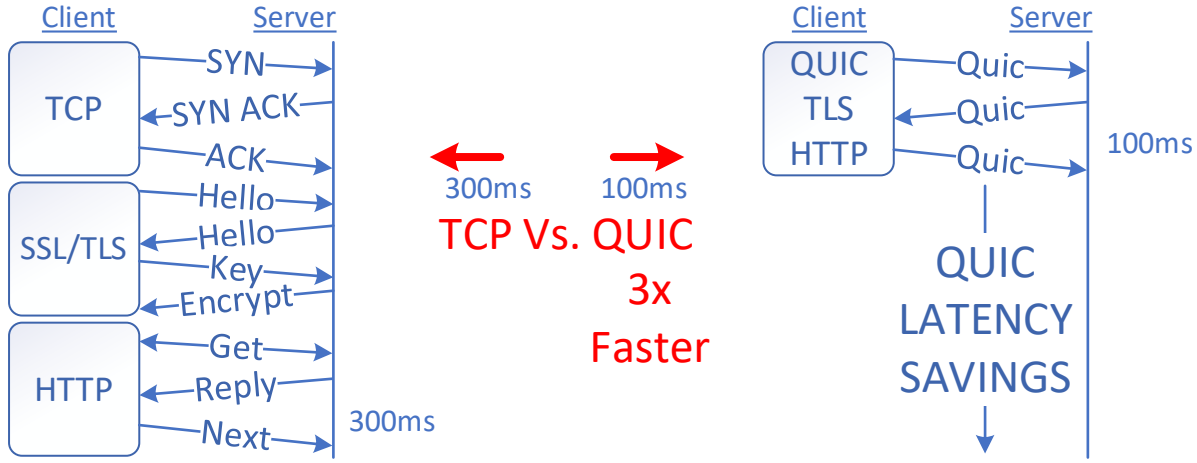


Diagram TCP vs. QUIC (c) bill@alderson.us, reuse granted with attribution

TCP vs QUIC 0-RTT Single Connection Setup

Network devices across global networks take significant time to set up connections. Figure 1 shows a client setting up a connection to a server. The client on the left initiates a TCP three-way handshake to a server in the middle. QUIC does in one volley what TCP / SSL and HTTP Hypertext Transfer Protocol require in many volleys.

Single Object One Video Load

Measurement	TCP	QUIC	QUIC RAW Advantage	Mixed Results TCP HTTP/2	Difference
Sessions	1	1	0		0% Same Sessions
Packets	1985	2123	-138		-7% TCP Fewer Packets
Time span, seconds	40.332	40.241	0.091		0% QUIC Marginally Better Time
Average pps	49	53	4		7% QUIC Better Average pps
Average packet size	1089	1192	103		9% QUIC Larger packet size
Bytes	2161738	2530541	-368803		-17% TCP Fewer Bytes
Average bits/s	428000	503000	75000		18% QUIC Better Throughput bits/s

Full Page Load With Multiple Objects

Measurement	TCP	QUIC	QUIC RAW Advantage	Advantage QUIC HTTP/3	Difference
Sessions	13	1	12		92% Less Sessions
Packets	679	577	102		15% QUIC Fewer Packets
Time span, seconds	2.371	1.317	1.054		44% QUIC Significantly Better Time, seconds
Average pps	286	438	152		53% QUIC Better Average pps
Average packet size	899	1156	257		29% QUIC Larger packet size
Bytes	610673	667225	-56552		-9% TCP Fewer Bytes
Average bits/s	2060000	4053000	1993000		97% QUIC Better Throughput bits/s

QUIC vs TCP Object Page Load Benchmarks (c) bill@alderson.us, reuse granted with attribution

Exhibits display results of two QUIC vs. TCP tests, at top a single YouTube video load. QUIC wins by 90ms, losing on total bytes and packets to TCP using fewer of both. On bottom, QUIC uses one session to TCP's 13 winning handily in every area except total bytes. QUIC doubles effective bandwidth speed and cuts time. The bottom full page was small and had few objects, a larger page with more objects would be far greater than 3x QUIC. Later in the detailed analysis we will discuss these tests further.

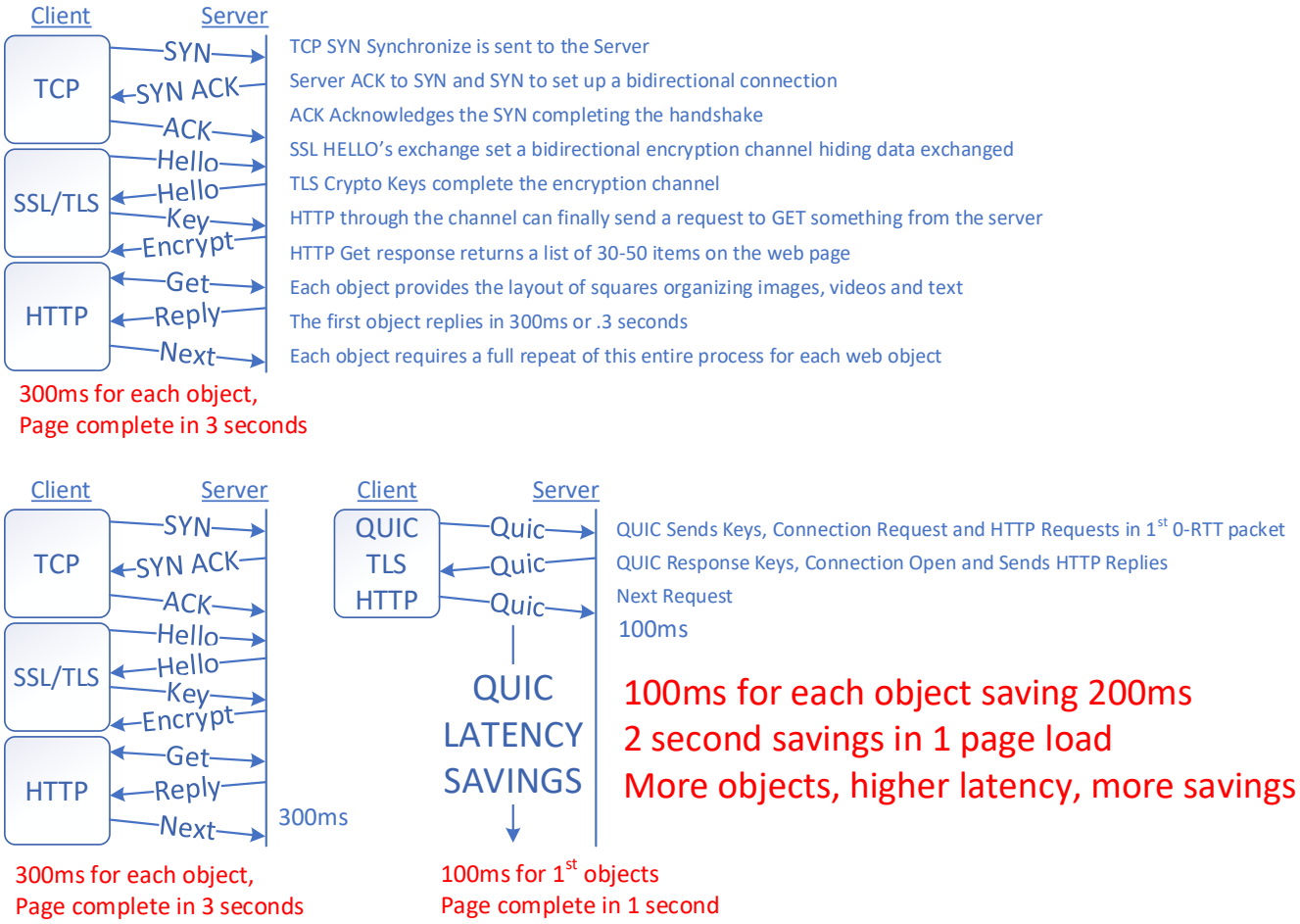
Zero, the number, continues its challenging history, having its origin from India around 600AD lost, returning with Arabic numbers in the twelfth century – rejected as heretical, nevertheless being foundational to Algebra and Calculus mathematically calculating the universe. Zero finds a new application for QUIC describing Zero 0-RTT Round Trip Time latency. It takes a minute to understand that HTTP issues GET Requests for webpages. Usually, before a GET can be spawned, HTTP must await TCP and SSL/TLS to complete their handshakes. QUIC initiates a session, sends initial crypto keys and multiple HTTP requests in the very first packet. For HTTP that means 0-RTT, gaining meaningful use of the very first packet, instead of many time-consuming cross network volleys.

QUIC, the new IETF protocol on the right contrasts conventional TCP/IP Operation by having only one volley carrying all three protocol functions: TCP / TLS / HTTP. (SSL and TLS are often considered either name). IETF calls this 0-RTT as HTTP can send the first GET with zero setup latency.

Early client-server protocols suffer from single request – reply in each packet. SQL Structured Query Language, HTTP and file access protocols like Unix NFS Network Files System and Microsoft SMB/CIFS Server Message Block / Common Internet File System are no exception. Bottlenecks are painfully obvious motivating change. An established protocol evolution changes experiences significant resistance – and just like the number zero's rejection we adapt and accept the disruption to receive the benefit. Billions of people continue adapting to still-new network technology. Benefit remains the catalyst for change, speed and performance, a powerful motivator. Detailed analysis with theoretical explanation provides smart people with the ability to endure and invest in change. I offer definitive factual analysis, with experienced knowledge to accelerate acceptance of an informed, secure path to QUIC benefits.

To illustrate another need for change, HTTP suffers from a single Request – Reply Command in each transaction request requiring a packet for each request. In addition to the benefits of 0-RTT setup latency for HTTP transactions, HTTP/3 provides additive optimizations combining more HTTP Commands into a single request. Put 0-RTT and HTTP/3 together and hit a performance home run. Look at Figure 2 step by step providing theoretical results, later packet by packet analysis will further enhance understanding with practical evidence.

TCP suffers from the three-way handshake, SSL/TLS by its required multi-volley set up and HTTP suffers from single request-reply transactions as seen in Exhibits.



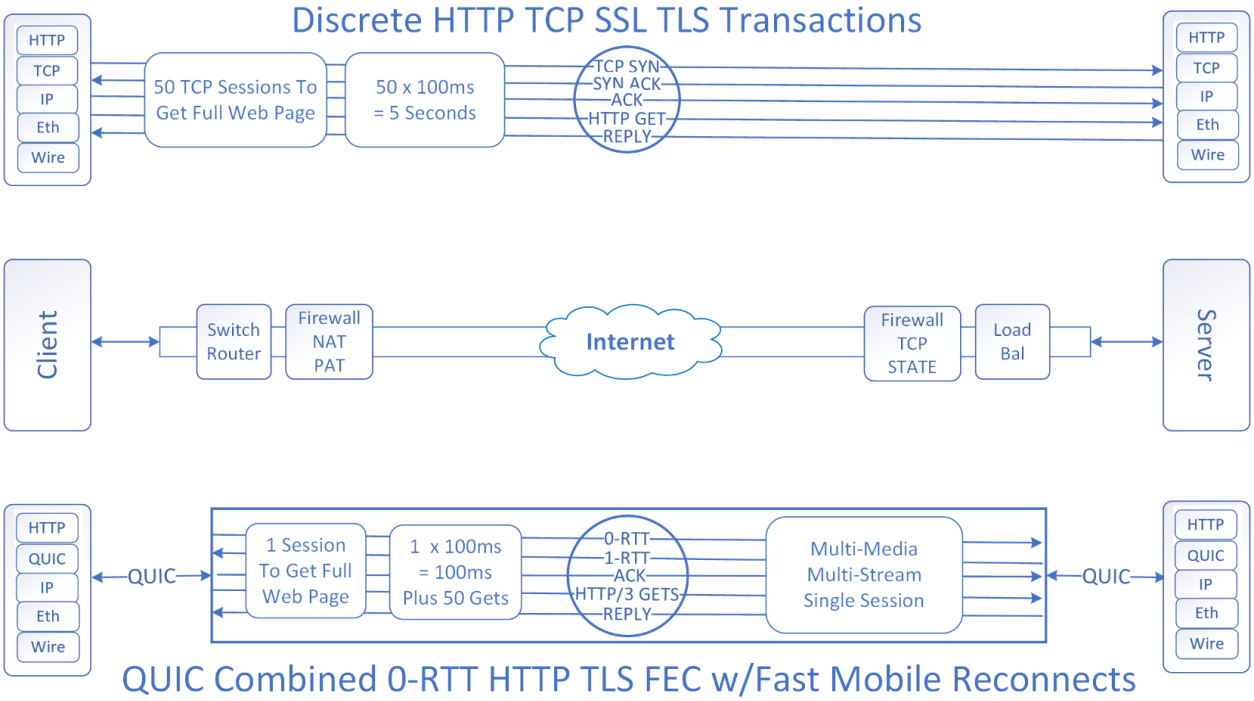
QUIC Transaction Analysis Theory Diagram TCP vs. QUIC (c) bill@alderson.us, reuse granted with attribution

QUIC solves the volley problem overcoming latency through multiple network command consolidations.

Exhibits examine OSI layer functions in order of operation – not layer hierarchy. TCP, a layer 4 OSI Transport reliable protocol opens a connection applying OSI Session layer functions to create the session. Next OSI Presentation layer function SSL/TLS provides end to end encryption and finally HTTP OSI Application layer commands and instructs data transaction actions.

Exhibits again show the correlating functions TCP, SSL / TLS / HTTP for comparison of QUIC function performance. QUIC undeniably contains multiple OSI layer functions. QUIC rides atop UDP an OSI

Transport unreliable simple protocol providing port multiplexing exposed to the network so firewalls and peers can understand what protocol is in operation.



Comparing TCP to QUIC with Middleboxes (c) bill@alderson.us, reuse granted with attribution

Exhibits offer a look at the components between the client and server transactions. Important to understand is that both TCP and IP Network layer expose their entire headers to network middleboxes as they traverse the network. Particularly important are the control bits of session setup that allow the middleboxes to know the session state. Session state is used by firewalls to know when a session is starting, in mid exchange or ending the session. It's the state information that allows the firewall to begin the aging of a session from allowed access. The most important aspects are who is initiating a session, and if it is from the trusted inside of a firewall or from the untrusted outside of the firewall. Criminals are sending TCP connection requests with a SYN to see if an address will answer on just about every TCP port. Think of a burglar checking every door on a car or home to see if it is unlocked. Firewalls know the door is not being attempted from the trusted inside location and denies access. A firewall knows when the initiator of a session is inside and by default lets devices inside create connections to the outside world. Once an initiator spawns a session to an outsider, the response packet is allowed back into the trusted area. Firewalls will not let the session live forever but if the inside and outside devices communicate regularly it continues to allow bidirectional access. Firewalls watch the content of TCP flags to know when sessions end or close.

Why is quick not always safe? QUIC packets blind firewall state information found in TCP. State information allows middlebox devices to know when a session is started when it's in the midstream. And when it's ending firewalls, middleboxes look at that state information maintaining knowledge of the start and end of sessions aging them out when appropriate. Accordingly. When blind to state information firewalls must maintain a connection that may be a long persistent connection across a

protocol that does not send state maintaining keep alive packets. There is no standard time for QUIC session keep-alive packets at an interval to keep the session open until finished, and there is no way to know when a QUIC session is finished. For instance, MS-SQL sends TCP keep alive packets every 30 seconds to keep middleboxes open in both directions. Keep alive packets for a short-lived session creates chatter that may not be required. Not having a keep alive may shut the session off in one direction or the other denying service or causing fallback to TCP. Each application requires a few qualifications and settings to ensure reliability, resilience, and security.

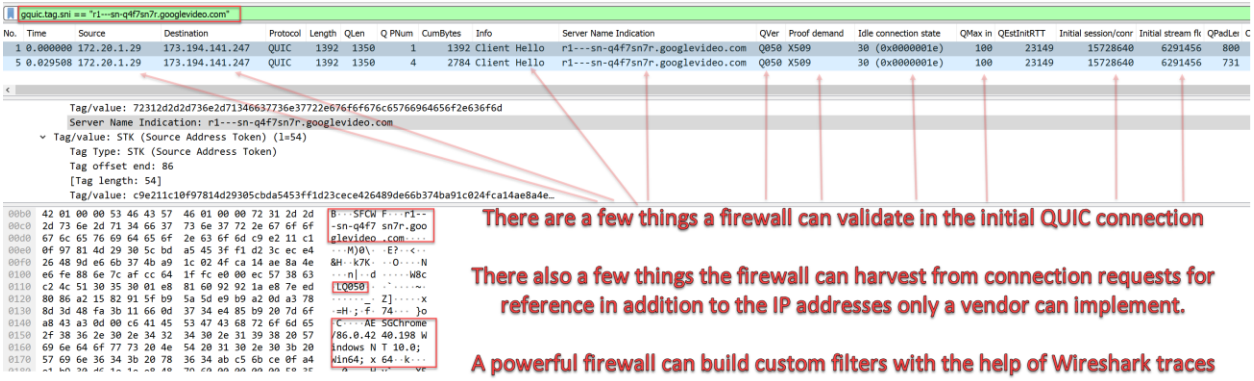
There are many different reasons QUIC protocol is not safe. We lose the ability for the firewall to maintain state and validation information because much of that information is encrypted.

The trouble with UDP is that there are no control bits, no Syn/Syn/Ack or session reset flags giving the Firewall state information to improve the security posture of sessions. At any time, a firewall can list the open connections and the status of the session. UDP only shows open or not. That holds true for Load Balancers and other gateways, devices, and VPN's. With UDP only, the firewall becomes blind.

That is why every firewall vendor recommends UDP port 443 used by QUIC is denied. You may easily web search for your firewall model instructions to deny port 443. Port 443 is used by other applications other than QUIC. More advanced firewalls can inspect protocol headers to validate that it indeed is carrying some validating element inside headers and or data fields.

Most QUIC packets expose few details to middleboxes as it encrypts everything after the initial packet setup, including the QUIC headers.

Firewalls use TCP handshake and SSL session state information for essential security checks. After QUIC encryption setup, the firewall cannot evaluate session status. Inability to inspect handshakes and control functions as allowed with TCP/SSL reduces security. Higher end firewalls with deep packet hardware-based inspection can evaluate the initial QUIC session setup – high end firewall features are costly. Merely updating software on most firewalls will likely have performance issues using software and central CPU processor resources opposed to hardware filter arrays.



Security Packet Filters to Validate QUIC (c) bill@alderson.us, reuse granted with attribution

Exhibits provides several protocol fields used in Initial Long header QUIC packets and fewer items in a short QUIC header in the remainder packets that are exposed to middleboxes as firewalls and load balancers that can be used to validate a properly formed QUIC packet. Firewalls need to investigate the QUIC headers to vet the existence of valid QUIC headers. In addition, firewalls can harvest some fields

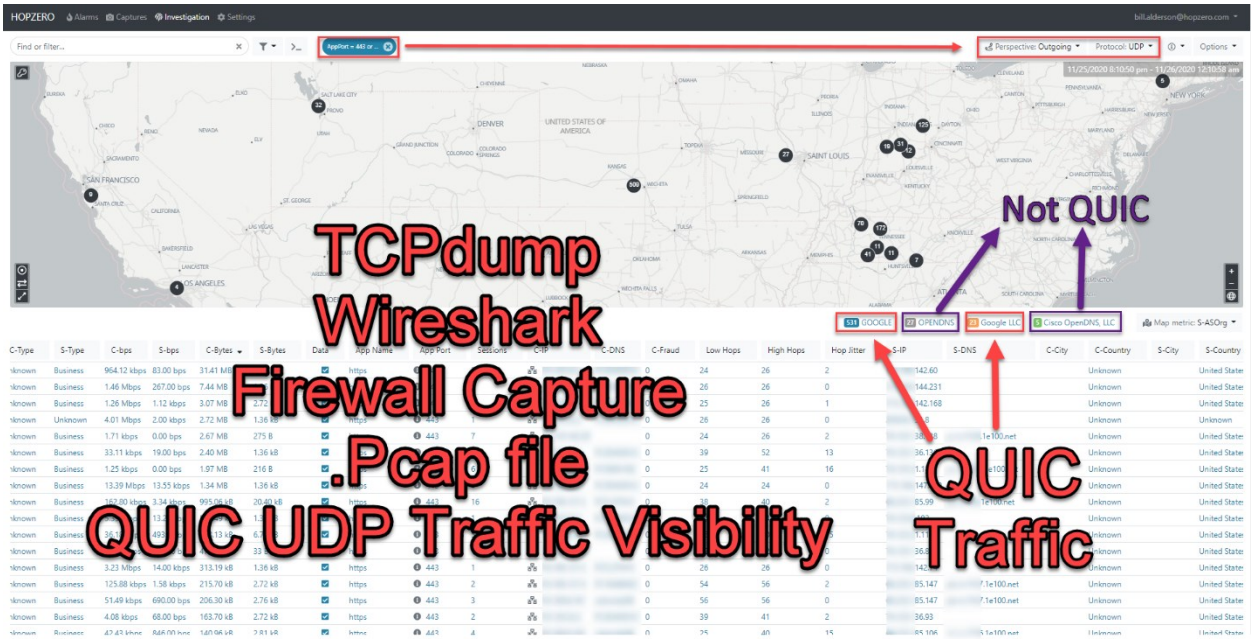
that describe the session to add context to the session naming. An example would be to capture the URL information, the QUIC version or the session ID. The remainder of the QUIC session packets are encrypted, but there are a few things to validate in each packet covering long and short QUIC packets so validating the session initialization packets to allow the session mapping that vetting those few items and the UDP socket (IP Address plus Destination / Source UDP ports). Long and Short QUIC packets are either a 1 or 0 at the same protocol header bit, not a byte offset, a bit which is a "bit" harder to discriminate. There is another IETF QUIC standard on things that do not change that can be used to determine the best way for QUIC packets to be vetted by a firewall to improve security.

When will QUIC be safe? When firewall hardware and software is updated, caution - only updating firewall software can be a dangerous assumption. Firewalls have hardware and software components. The hardware allows switching and decisions to be made very rapidly in hardware-based processor arrays. Using software only may impact the CPU adversely causing other security issues. If a firewall can be configured to set complex binary filters an enterprise might be able to set some deep inspection filters to improve QUIC security earlier than vendors support QUIC, careful as iteration may be required as standards are released. When a hardware-based firewall is built, it uses software and uses those hardware decision arrays very effectively.

How to identify QUIC utilization? On a computer examine the browser client config. Browser specific settings change often, see SecurityInstitute.com/QUIC to find updated instructions. Running Wireshark on your machine while browsing to anything Google and setting the protocol name "quic" in the filter pane will show only QUIC packets. If they show up, you are using QUIC.

An enterprise can examine firewall configuration and logs looking at UDP Port 443 incoming or outgoing traffic.

Here is an example of a TCPdump or Wireshark trace run through a tool that visualizes data travel by TCP or UDP traffic. Figure 6 displays where UDP port 443 Traffic is traveling into or out of an enterprise network, by IP, by country, how much data with deep security analysis on each external device.



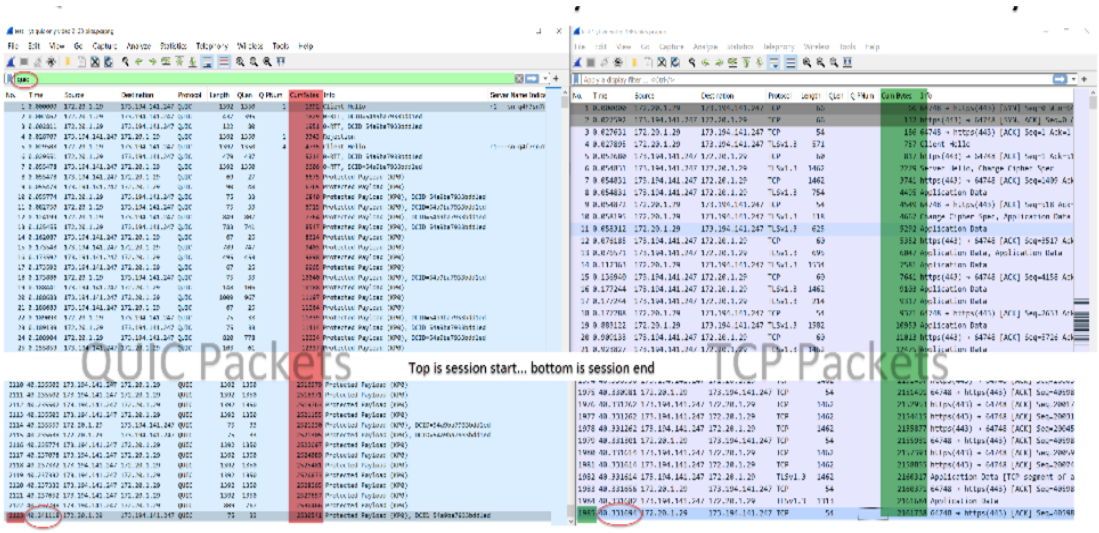
QUIC UDP Traffic Visibility from PCAP Capture File (c) bill@alderson.us, reuse granted with attribution

How would you stop quick, fast? Disable UDP port 443 or 80 for UDP functions of the firewall. Careful to analyze to ensure the traffic is QUIC and not an SSL UDP VPN as they often use the same ports. A thorough analysis should be performed before turning the protocol off unilaterally.

Figure 6 displays a comparative performance analysis of startup and end of QUIC and TCP packet sessions retrieving the exact same page contents of the same URLs. Provided here and in the graphic.

QUIC when enabled will load using QUIC on UDP port 443 unless blocked. If UDP port 443 is blocked QUIC will automatically fallback to discrete TCP – SSL – HTTP methods and current versions to retrieve the page. For each test QUIC was specifically enabled and disabled in the browser. Wireshark was used to capture the packets in both cases. TCP has the feature of keeping a session open well after it is finished with the session transfer. The last unaffected Ack and TCP session end were removed exporting only the relevant packets involved in the data transfer to compare with QUIC.

QUIC is on the left providing a view of early session packets and on the right are TCP packets. The tests were not done simultaneously, but each one independently a few minutes apart. Many tests were run, every test varied in both QUIC and TCP by a few packets due to slight Internet imperfections that are immaterial to the results. Every test was materially the same. Circled or highlighted are a few things to consider. This test uses only one session for either protocol. Surprising was the amount of total data sent was higher using QUIC. In a subsequent analysis these session tests will be shown decrypted to see what happened in more detail and gain perspective on what happens inside both TCP / SSL /TLS and



Single Object One Video Load

Measurement	TCP	QUIC	QUIC RAW Advantage	Mixed Results TCP HTTP/2	Difference
Sessions	1	1	0		0% Same Sessions
Packets	1985	2123	-138		-7% TCP Fewer Packets
Time span, seconds	40.332	40.241	0.091		0% QUIC Marginally Better Time
Average pps	49	53	4		7% QUIC Better Average pps
Average packet size	1089	1192	103		9% QUIC Larger packet size
Bytes	2161738	2530541	-368803		-17% TCP Fewer Bytes
Average bits/s	428000	503000	75000		18% QUIC Better Throughput bits/s

QUIC Vs. TCP One Video Packet Analysis (c) bill@alderson.us, reuse granted with attribution

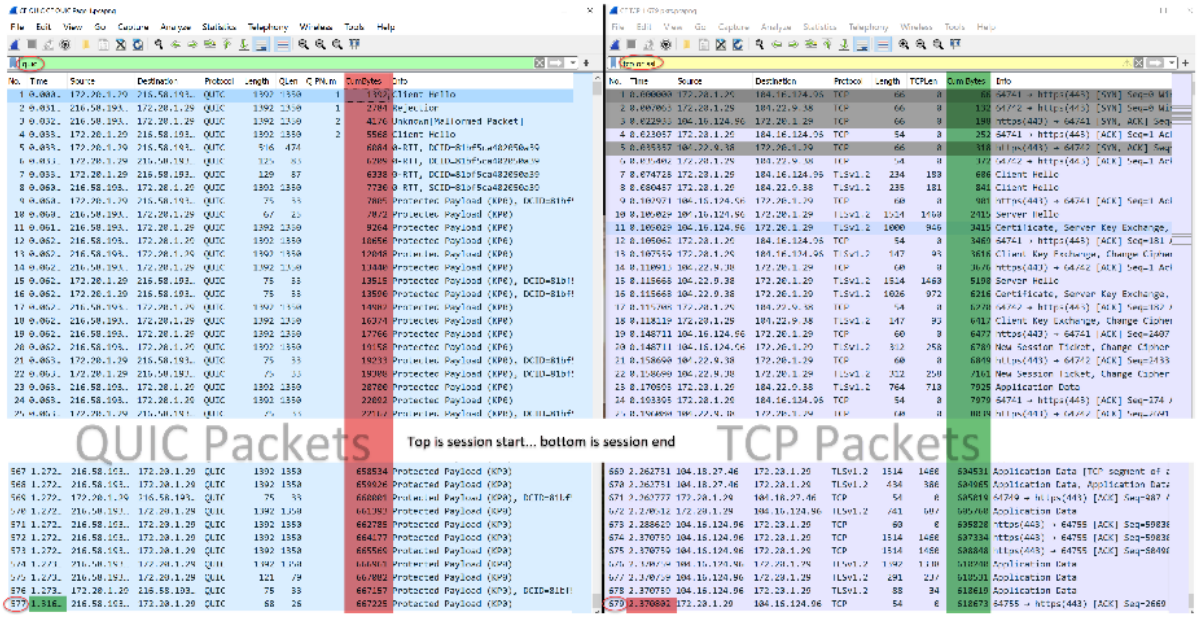
QUIC TLS encrypted tunnels through which HTTP commands are hidden.

Cumulative bytes in both sessions confirm the total bytes required for each session.

Exhibits display a comparative analysis of startup and end of QUIC and TCP packet sessions retrieving the exact same page contents of the same URLs. This test setup and assumptions are the same as previous tests. Results are quite different as QUIC outperforms TCP handily with only one exception total bytes.

Here QUIC combines all protocols into one UDP session containing the TCP equivalent of 13 sessions into one QUIC session. The only category QUIC did not win handily was total bytes.

QUIC vs. TCP Full Page Packet by Packet Performance Analysis



Full Page Load With Multiple Objects

Measurement	TCP	QUIC	QUIC RAW Advantage	Advantage QUIC HTTP/3	Difference
Sessions	13	1	12	92%	Less Sessions
Packets	679	577	102	15%	QUIC Fewer Packets
Time span, seconds	2.371	1.317	1.054	44%	QUIC Significantly Better Time, seconds
Average pps	286	438	152	53%	QUIC Better Average pps
Average packet size	899	1156	257	29%	QUIC Larger packet size
Bytes	610673	667225	-56552	-9%	TCP Fewer Bytes
Average bits/s	2060000	4053000	1993000	97%	QUIC Better Throughput bits/s

Benchmark Page https://cloudflare-quic.com

QUIC Vs. TCP Full Multi-Object Page Analysis (c) bill@alderson.us, reuse granted with attribution

QUIC (pronounced "quick") has been considered both an abbreviation and acronym standing for Quick UDP Internet Connections. In recent Internet-Drafts IETF insists QUIC a name for the protocol and not an abbreviation or acronym as previously thought – not sure why, but it could be due to trademark or copyright legal issues, perhaps due to its Google roots making it more palatable to others.

Roskind at Google implemented and deployed QUIC in 2012, describing to IETF Internet Engineering Task Force in 2013, more formally becoming an industry collaborated standard in 2016. Here is a link to the latest release Nov 2, 2020 https://datatracker.ietf.org/doc/draft-ietf-quic-applicability/

IETF Internet standards follow a process with steps and names. An Internet Draft (ID) may become a Proposed Standard, Draft Standard and on to an Internet Standard. Often referred to as RFCs Request for Comments, some are selectively or experimentally loosely implemented in products. Fundamental RFC's are written to define the administrative procedures of promulgating standards in the form of procedures.

Bill Alderson, https://Cogent.Management Advisory & Response

Bill started examining packets with a Halcyon serial data scope in 1980 as a network communications engineer at Lockheed Missiles & Space Company, Sunnyvale, California. Later joining Network General Corporation (Sniffer) as Secure Systems Manager. At PMG NetAnalyst Bill started On-The-Wire Newsletter on protocol analysis. Bill created the Certified NetAnalyst Network Security Forensics Training and Certification Program in 1995, certifying more than 3,500 Network Security Forensic Professionals from 27 countries. June 2020, November 2022 Bill received utility patents for a Method and System for Limiting the Range of Data Transmission Across IP Networks to reduce attack surface from global to local, creating a network distance perimeter for safer communications.

Bill responded to the Pentagon immediately after 9/11 to lead the communications recovery effort, solved high visibility US Stock Market denial of service attacks using protocol analysis and asked by Joint Chiefs to deploy with US Troops to Iraq / Afghanistan solving intelligence and biometric applications key to detaining insurgents.

Bill hosts the https://Disaster.Stream Podcast highlighting lessons-learned from high visibility, high stakes incident responders.