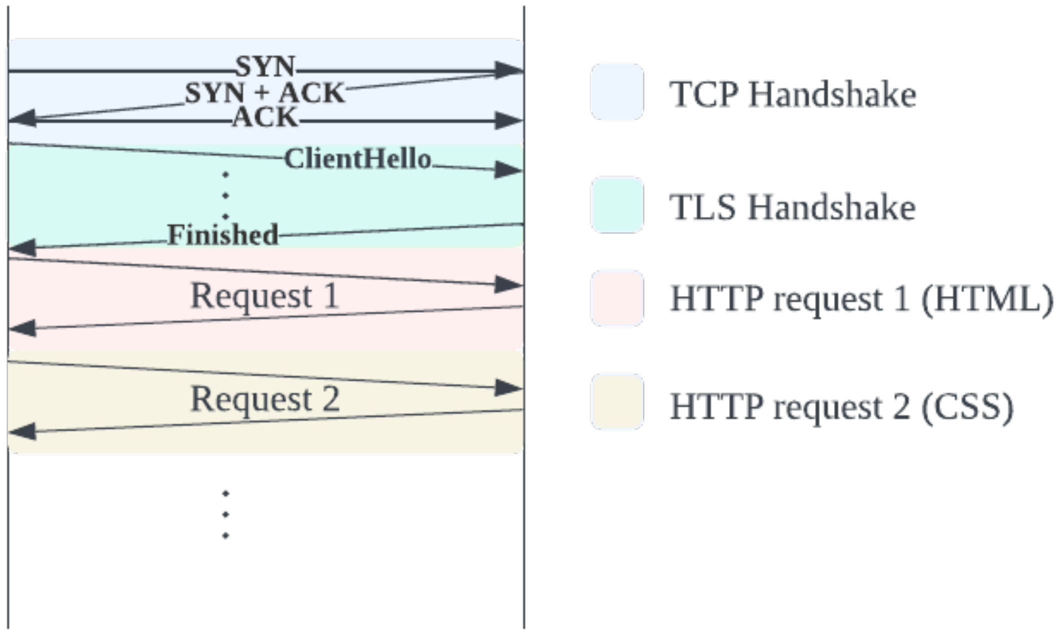
HTTP/3 – Reading

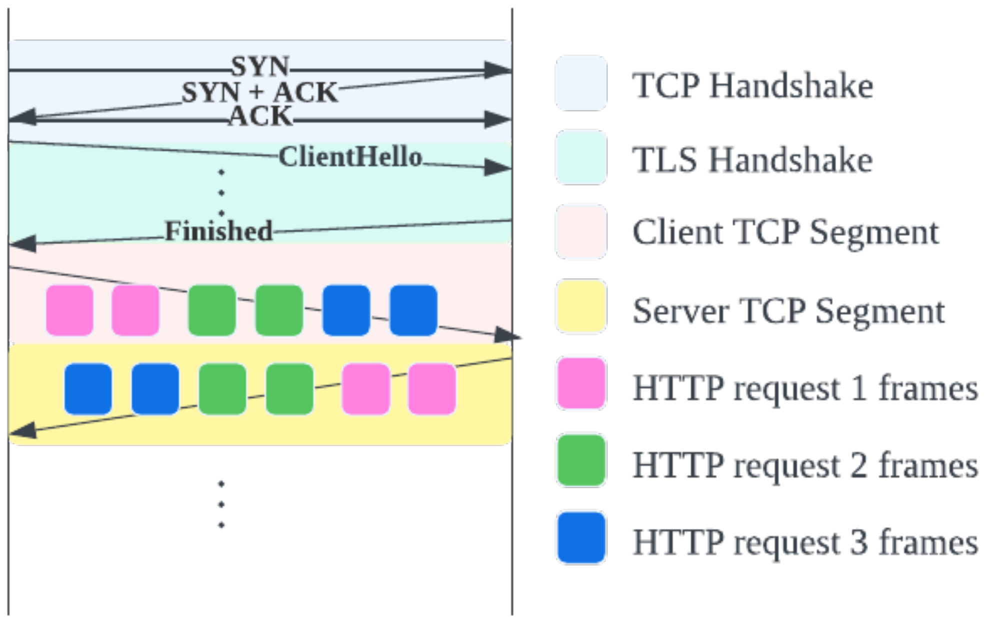
Hypertext Transfer Protocol(HTTP) is the basis for the world wide web. It is a set of rules used by clients like web browsers and web servers to exchange data over the internet. From scrolling through instagram feeds to making online transactions in the bank, we use HTTP everyday. As the significance and use of HTTP has grown tremendously over the years, HTTP has also been evolving to become faster, efficient and secure.
We now have three major versions of HTTP — 1, 2, and the latest being 3. In this article, we will take a closer look at the improvements made to HTTP/3, most importantly the change in transport layer protocol from TCP to QUIC which makes it more faster and secure.

A brief history of HTTP/1 and HTTP/2
The older versions of HTTP use TCP protocol in the transport layer. In HTTP/1, clients could only make one request at a time in a TCP connection, and there is a limit to the number of concurrent TCP connections a browser can establish per domain. This limit is 6 for browsers, such as Google Chrome and Mozilla Firefox. If a website had to load more than 6 resources, it may not be done concurrently.



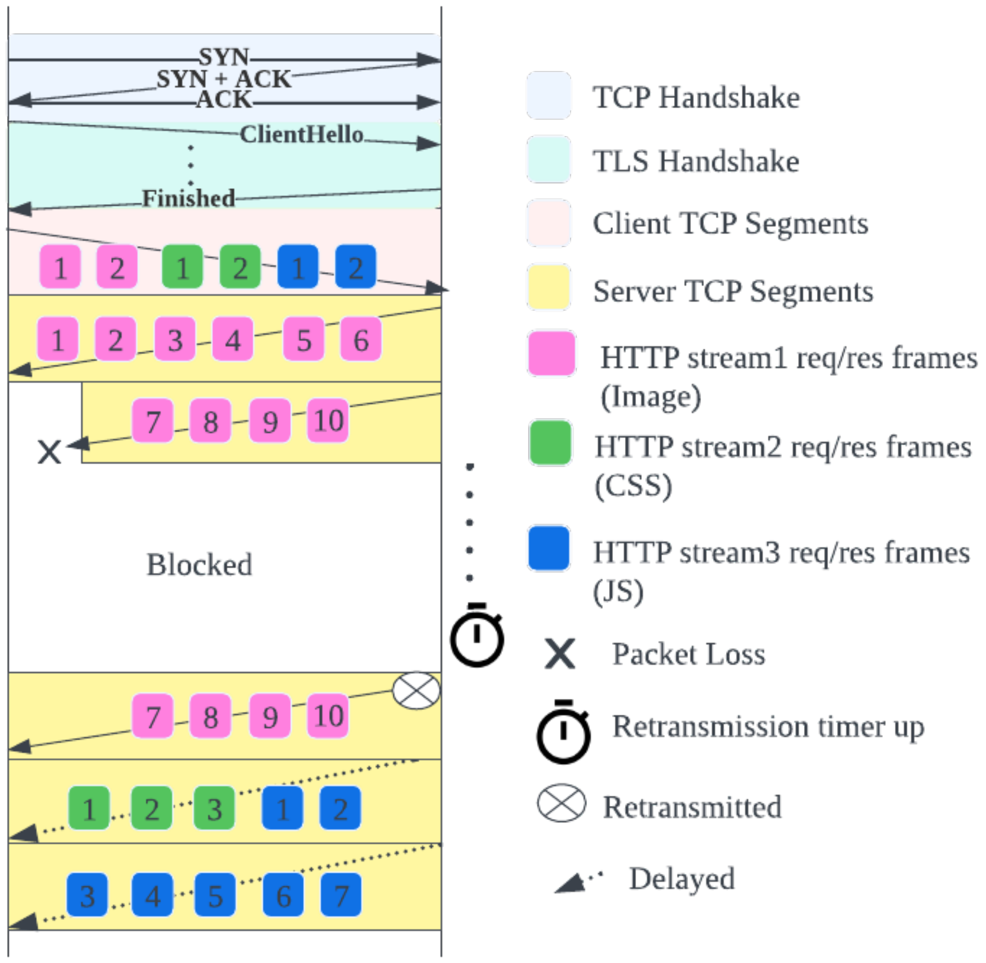
(HTTP/1 request flow)

Establishing a TCP connection involves some overhead, if HTTPS is used, which is most often the case, each connection has to spend some more time securing it through the TLS protocol. All this slowed down the load time of websites, as modern websites need to load several resources.



(HTTP/2 request flow)

HTTP/2 addressed this problem by offering multiplexed streams in a TCP connection. Each request is sent in a stream allowing clients to make multiple parallel requests in a single TCP connection. This was a major performance boost in HTTP/2 that made websites load faster.

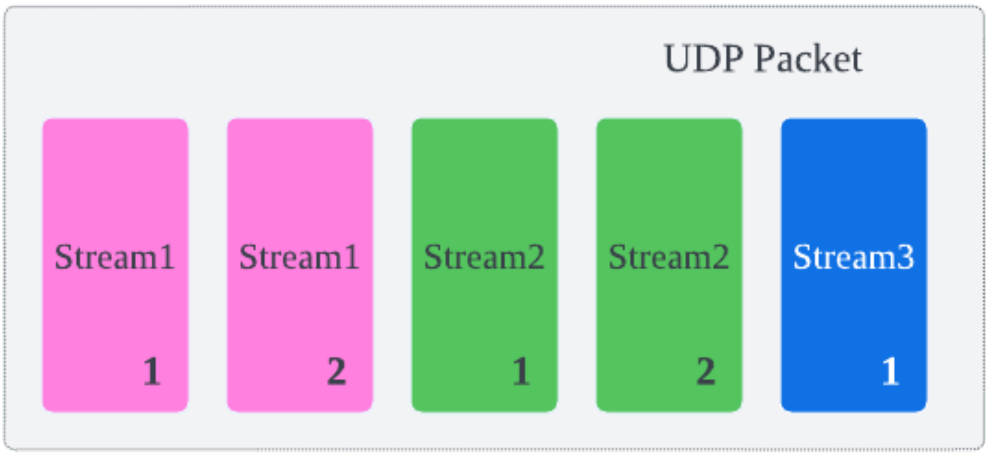


(HTTP/2 packet loss)

It still has another problem called head-of-line blocking(HOL). Streams are known to HTTP but not to TCP. Individual frames of different streams are numbered and transmitted as ordered segments in TCP. If a segment gets lost in transit, TCP will resend that after a timeout, blocking all other segments in the connection. So if a website had to load three resources(an image, a CSS file, and a JS file) in a single TCP connection, a packet loss for one of the resource requests will block the other two requests.

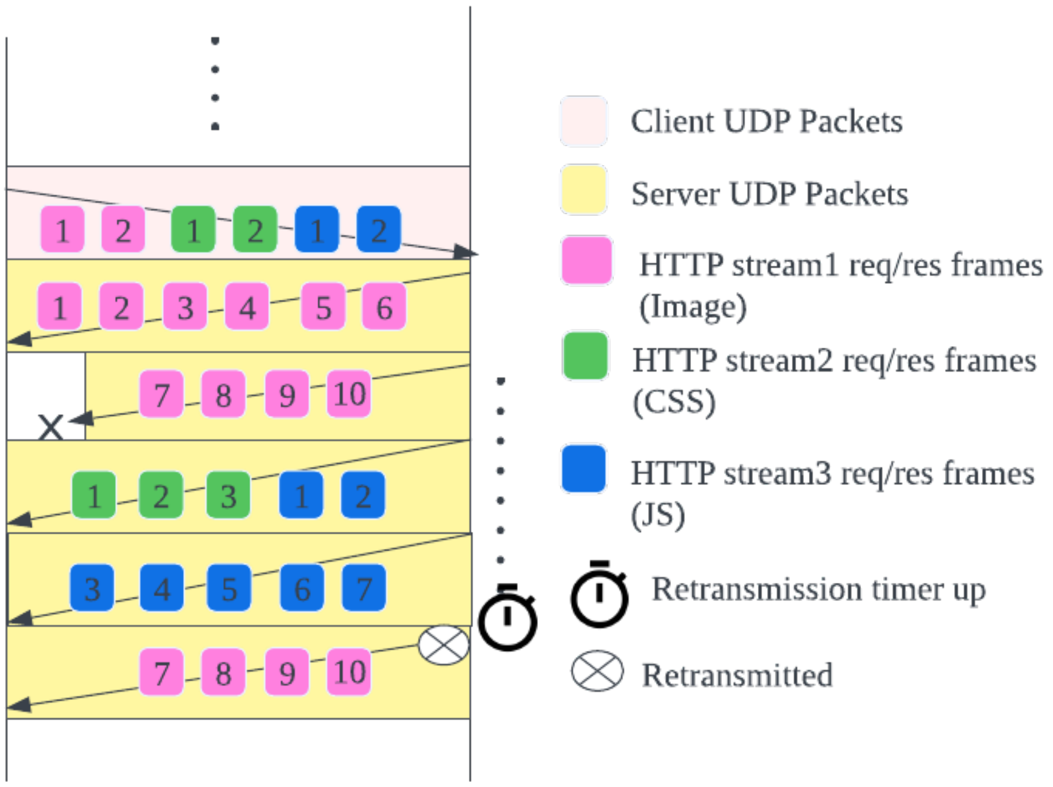
HTTP/3 and QUIC

Quick UDP Internet Connections(QUIC), pronounced as quick, is the transport layer protocol used by HTTP/3.



(QUIC UDP packet)

HTTP/3 also offers multiplexed streams like HTTP/2 using QUIC in the transport layer. Data in a QUIC stream is split into smaller chunks called frames and transmitted in UDP packets. Frames inside a UDP packet will contain the sequence number and ID of a stream, so the receiver can reorder them if it is received out of order.

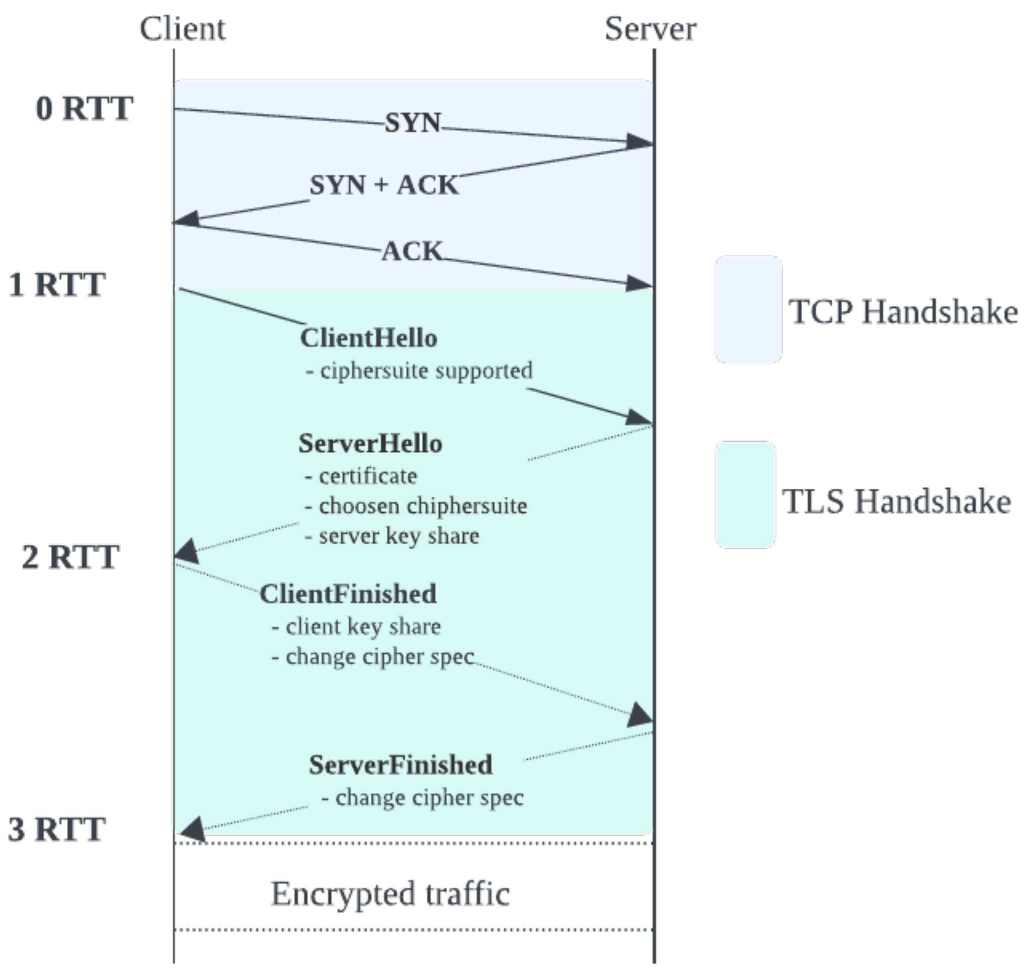

Client UDP Packets
Server UDP Packets
HTTP stream1 req/res frames (Image)
HTTP stream2 req/res frames (CSS)
HTTP stream3 req/res frames (JS)
Retransmission timer up
Retransmitted

(HTTP/3 QUIC packet loss)

No Head Of Line Blocking
If a packet gets lost, QUIC will re-transmit it after a timeout just like TCP, but it won't affect other streams as shown in the diagram above.

Faster connection establishment

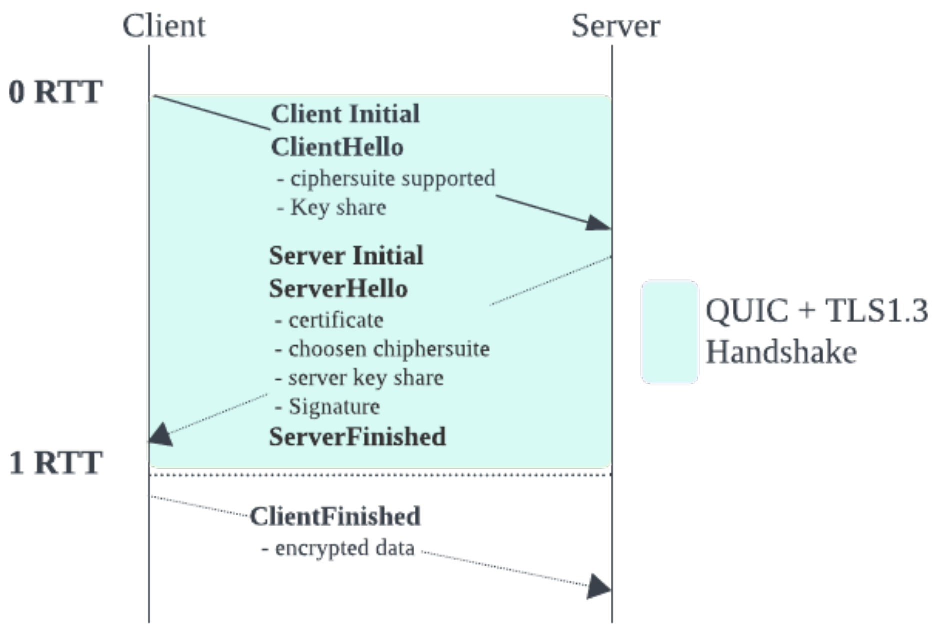
1 RTT Connection Establishment
HTTP/3 and QUIC offer 1 Round Trip Time(1-RTT) connection establishment and 0-RTT connection resumption, while HTTP/2 requires 3 RTT for a secure connection establishment.



(HTTP/2 3-RTT secure connection establishment)

As shown above, in HTTP/2, the first 3 packet exchanges are for TCP connection establishment, and the last 4 packet exchanges are for securing the connection using TLS 1.2. The third packet for TCP ACK to the server is combined with the TLS ClientHello message. So in total a secure connection establishment in HTTP/2 with TLS1.2 requires 6 packet exchanges or 3-RTT.

TLS is embedded into QUIC making the connections secure by default.



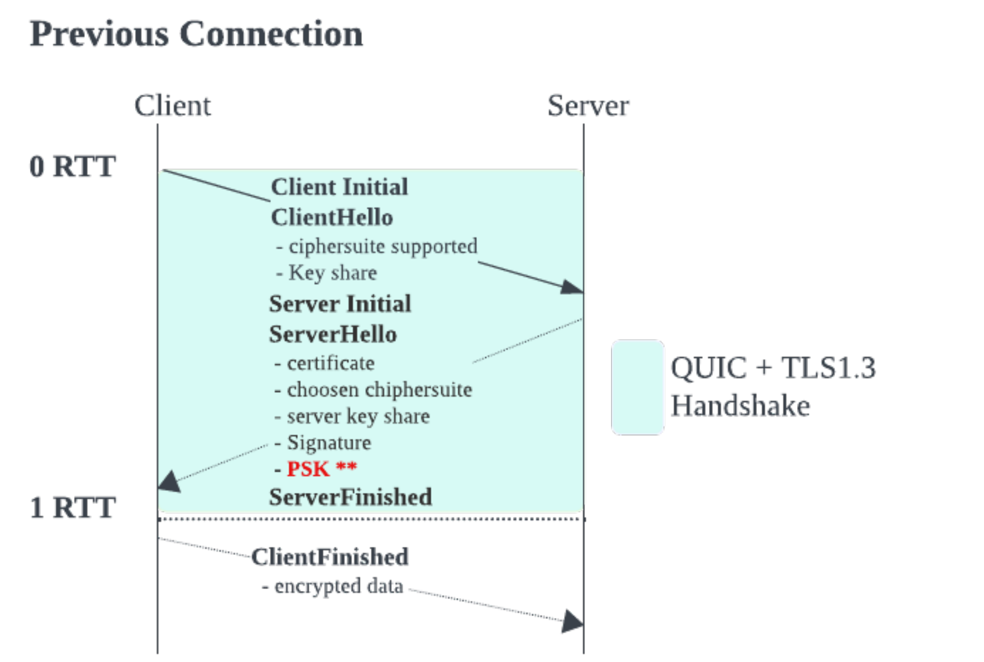
(HTTP/3 1-RTT secure connection establishment)

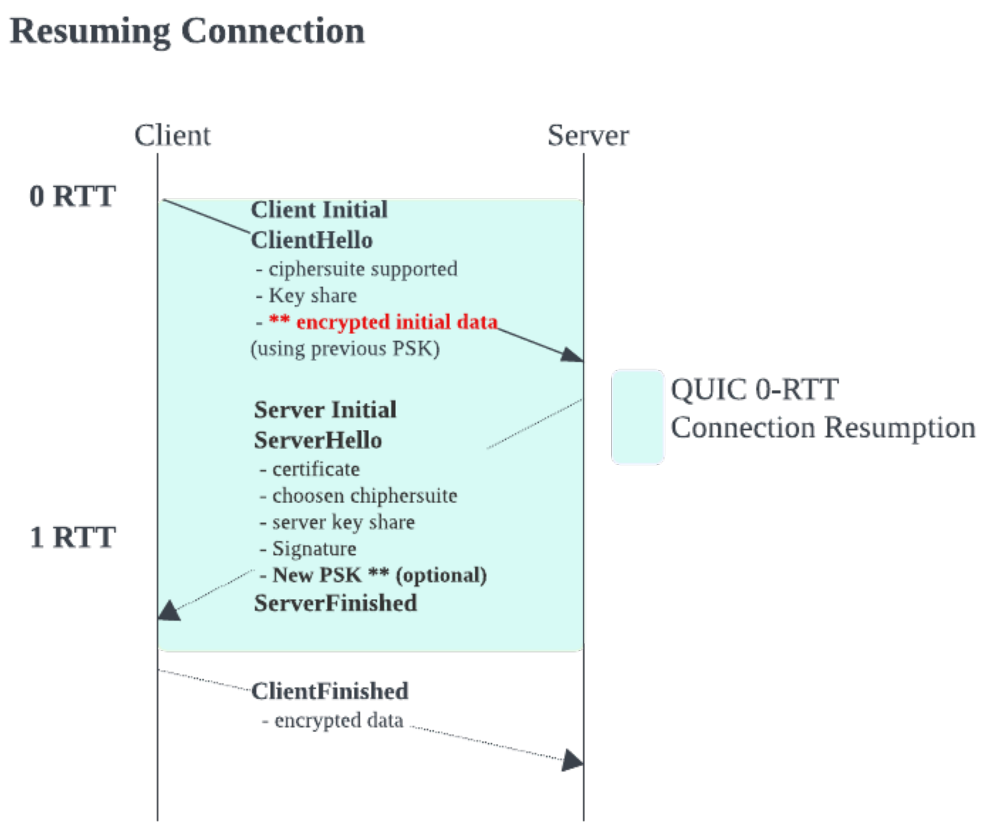
Unlike HTTP/2 where connection establishment, and securing the connection(HTTPS) are two separate sequential steps, HTTP/3 combines them into one single step. By using UDP instead of TCP, it eliminates the 3 way handshake, and by using TLS1.3, it reduces TLS handshakes from 2-RTT to 1-RTT.

Enhanced Security
TLS1.3 provides greater security by deprecating several weak cipher suite options available in the previous version and standardising on only 2 different key exchange options — ECDHE with X25519 or P-256. In TLS1.2 client and server use 1 RTT to just negotiate and agree on the key sharing algorithm, after that they both share their public parts and arrive at a common shared secret. The shared secret is then used for encrypting the connection. But In TLS 1.3 there is only one key sharing option — ECDHE, servers will typically support both the parameters X25519 and P-256 of ECDHE, so the client will select one of them and send its public part along with the initial message. In an unlikely situation where the server doesn't support the selected ECDHE parameter, it will request the client to retry with the other parameter, in such a rare case connection establishment takes 2 RTT. You can learn more about the key exchanges and improvements in TLS 1.3 from this article.

0-RTT Connection Resumption
TLS1.3 allows encryption using a Pre-Shared-Key(PSK). After a successful connection establishment, client and server can share a key and store it for a later use. The key exchange can also be done out of band. For future connections to the same server, clients can then choose to encrypt and send some initial data using the pre-shared-secret in the first packet to the server. Only the initial data from client to the server is

encrypted with the PSK, it will follow the 1-RTT handshake described above to arrive at a new secret for the rest of the connection to provide forward secrecy.

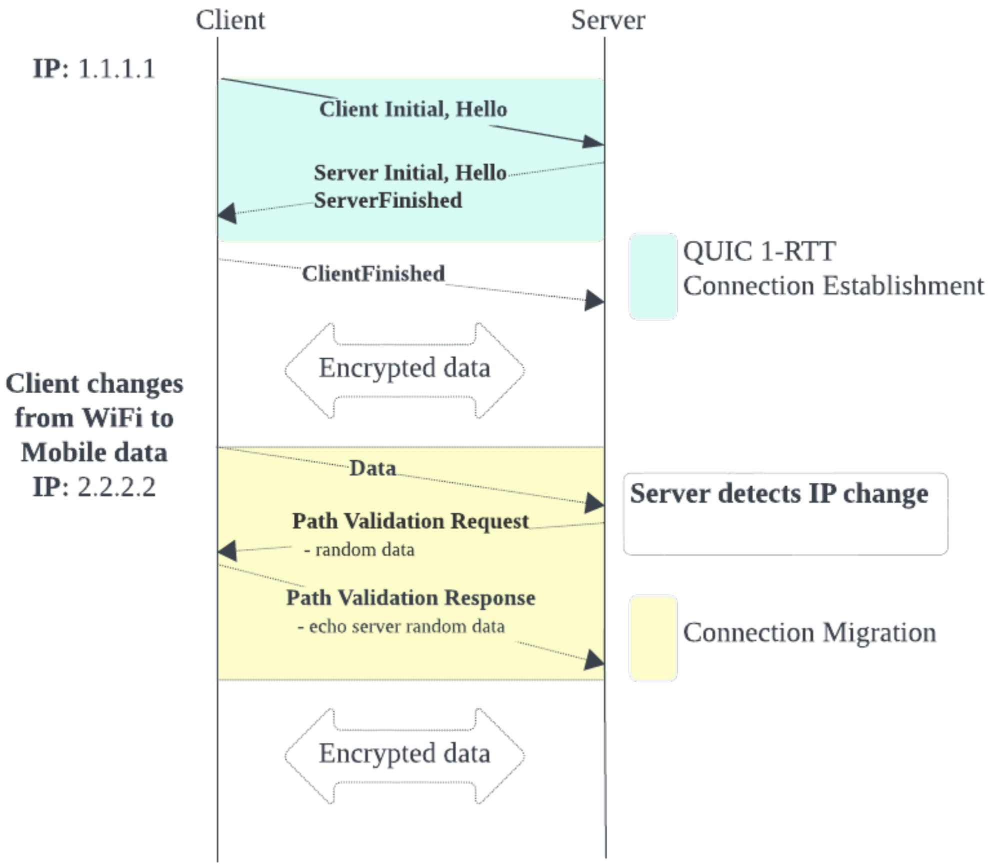




(HTTP/3 connection resumption)

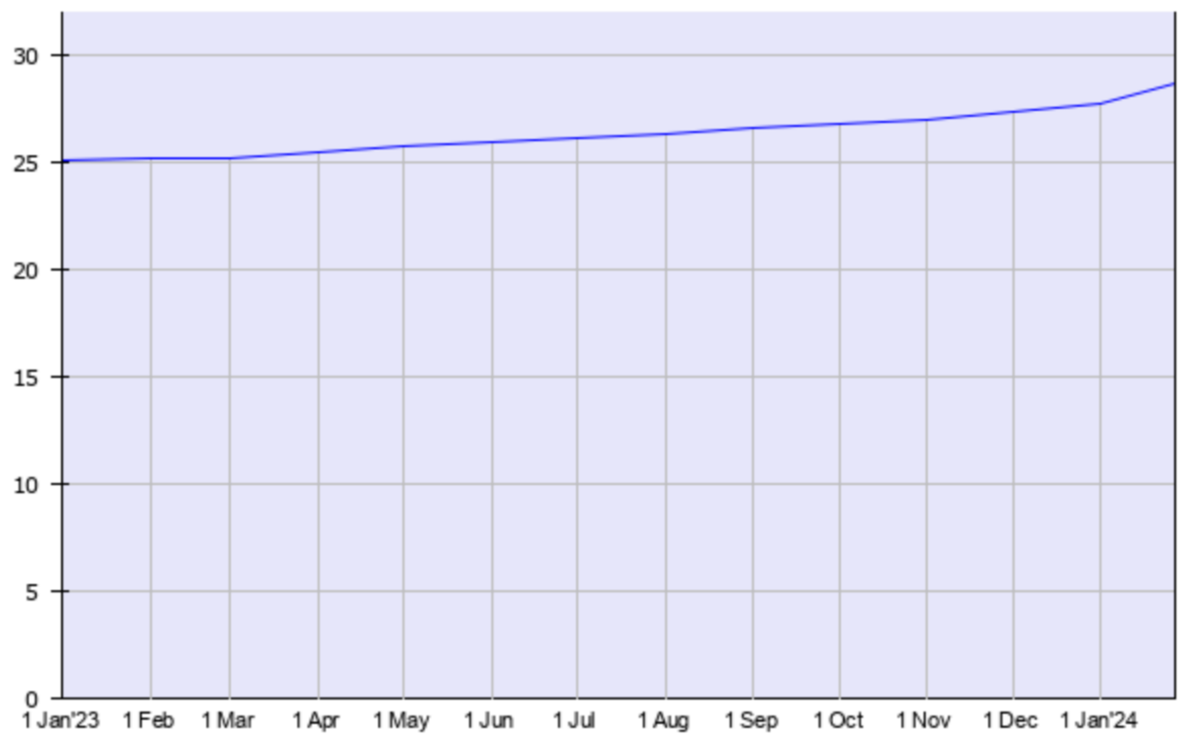
This greatly improves the performance when reconnecting to the same server, but has a downside to it; An attacker can intercept the packet and replay it at any time. Imagine the initial client data sent is for a banking application to register a transaction of 100$, the debiter will lose money if the packets are replayed. So the recommendation for servers is to accept the initial PSK encrypted data only for non state changing requests, in REST terms, only for GET requests.

Connection Migration
QUIC can persist a connection across network changes, unlike TCP which requires the same endpoint(IP and port) throughout the connection. The endpoint can change when a NAT binding of a client changes, or when a user switches between WiFi and mobile data on their phone. QUIC allows migration of connections from one endpoint to another except during connection establishment phase where the same endpoint is required for security.
To verify the new endpoint is owned by the client, and source IP is not spoofed, the server will do a path validation. Path validation is done by sending some arbitrary data to the client and verifying the client can decrypt and return the same. No man in the middle can see that arbitrary data because it is encrypted with the shared session secret obtained during connection establishment.



(HTTP/3 connection migration)

The connection ID will change after a successful connection migration. This is to make sure no one other than the client and server can associate a connection across different endpoints.



Usage of HTTP/3 for websites, 27 Jan 2024, W3Techs.com

More than 28% of websites use HTTP/3 today. There's no doubt this percent will increase in the future, but it's not going to be an easy walk. HTTP/2 migration from HTTP/1 was relatively easier since the underlying transport protocol TCP was not changed. HTTP/3 adoption goes beyond the client and server supporting it. Different network components such as load balancers and firewalls in the request path over the internet need to support QUIC. This, in my opinion, is the key challenge for HTTP/3 migration.

Source: https://medium.com/@gtamilarasan/a-closer-look-at-http3-quic-1b2b97a5cd50