QUIC Reading summary

VINAYAKA GADAG (vgadag@iu.edu)

QUIC is a transport layer network protocol, built on top of UDP to improve the performance of HTTPS traffic. It replaces most of the HTTPS stack including TLS. TCP connections usually take one round-trip delay and TLS adds two to this delay. To reduce this, we need to limit the number of connections by client and server. QUIC sends the packets that are authenticated and encrypted from the initial packet to reduce the number of RTT handshakes between the client and server. This approach reduces the handshake latency for most of the connections. The RTT handshake is reduced by at least one and supports the 0-RTT by using known server credentials on repeat connections. QUIC is designed with multiple features considering the disadvantages of other protocols such as TCP.

The motivation behind implementing the QUIC is simple. Latency in the current situation where web servers are consumed by billions of users on daily basis is more than we can estimate. All the web traffic now is moving towards the secure connection, and this adds more latency time in accessing the web servers. Current transport mechanisms use the TLS/TCP ecosystem, and their limitations are the big hurdle in improving the performance and reducing the latency time. Middlebox adds extra overhead by interfering with packet transfer when the insecure header is passed on and this adds to the current latency time. In this situation, we need faster implementation and deployment to the internet protocol to reduce the latency. But it is difficult and time-consuming using TCP since it is implemented at the OS kernel level and would require OS upgrades which takes a lot of time, and we need rigorous testing and performance testing of the entire OS. QUIC is built on the user-space which avoids dependency on other vendors like OS, middleboxes, etc. Which allows moving deployments to applications and makes the development, testing, and deployment easy.

QUIC is designed in a way that it provides its own stream for each request/response to avoid overlapping between different requests/responses. It provides the advantage of loss recovery by using the unique packet number to ensure that a lost packet would only affect the current stream and not the other streams or the entire connection and solves the retransmission ambiguity. It allows connections to transmit over IP address by using connection ID instead of modifying the IP/port. QUIC relies on combined cryptographic and transport handshakes which allow secure connection and the client caches the information once the connection is established and reduces the RTT on subsequent connection requests once the server is known. This improves the performance of the protocol. QUIC uses two types of flow-control mechanisms, connection-level, and stream-level to improve the head-of-line blocking among streams by limiting the buffer that the single-stream can consume.

The performance of the QUIC protocol has many limitations and TCP performs better in these conditions. Higher CPU utilization of server - for example, cryptography. Other conditions when QUIC's performance is subpar compared to TCP is operating in high bandwidths, low-delay, low-loss networks, pre-warmed connections- for example, YouTube. Continued research is being made

on improving the CPU utilization and expanding the implementation to include general-purpose multipath which will improve the performance. QUIC is proved to be a good replacement to traditional network settings using TCP because of the advantages such as connection establishment latency, low latency, improved congestion control, connection migration, strong encryption, limited dependency on middleboxes.