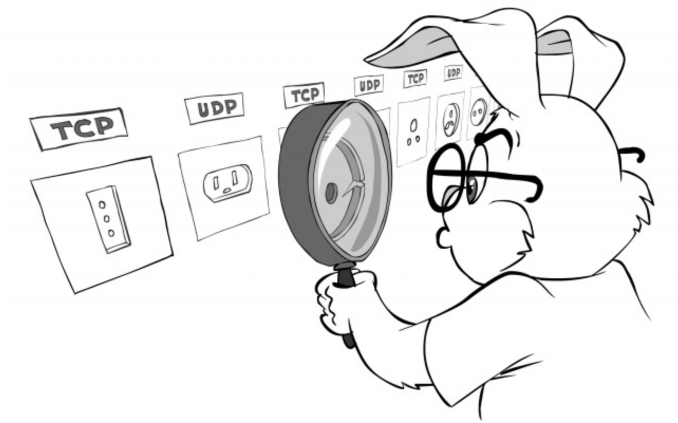
Outline

- Network Security Threads
- Securing end-to-end connections
  - Transport Layer Security (TLS)
- Security at Transport Layer
  - TCP, UDP, QUIC
- Security Network Layer
  - IP/ICMP, IPSec, VPNs, IPv6 Security



Transport Layer Attacks and Mitigation

1. TCP SYN Flood (DoS/DDoS Attack)
   - attackers exploit the TCP three-way handshake by sending a large number of SYN requests to a target server without completing the handshake, consuming server resources and causing DoS

- Mitigation:
  - SYN cookies: the server encodes connection information* in the sequence # field of SYN-ACK TCP header; if the client completes the handshaking (w/ ACK) and the cookie received is verified, resources are allocated for the connection
    - currently implemented in most recent OSs
    - e.g., setting the feature in Linux: `echo 1 > /proc/sys/net/ipv4/tcp_syncookies`
  - Backlog, Rate limiting: limits the backlog queue size (pending conn.) or the # of SYN requests a server accepts from a single source

* a cryptographic hash of the connection information, such as IP addresses, port numbers, and a secret value

Transport Layer Attacks and Mitigation

2. TCP Reset Attack
    - attackers inject forged TCP RST flag (reset) packets to forcefully terminate a TCP session between two parties, if they can guess or sniff the current sequence #s

- Mitigation:
    - use sequence number randomization to make sequence numbers more unpredictable
    - implement TCP/IP stack hardening on the server side, which ignores RST packets that do not match the sequence numbers of the active session
    - TCP RST limiting
    - firewalls and intrusion prevention systems
    - note: TLS protection does not impair a malicious TCP connection reset

Transport Layer Attacks and Mitigation

3. TCP Session Hijacking
   - an attacker intercepts and takes over an active TCP session by guessing sequence numbers and injecting malicious data into the session
     - altering TCP's sequence numbers my confuse the TCP state machine, causing desynchronization between the client and server, which may result in connection stalls, unnecessary retransmissions, or dropped segments
     - injecting malicious data will be detected and never reach the app, if TLS in use

- Mitigation:
  - use TLS to encrypt and authenticate session traffic, which prevents an attacker from modifying or injecting data into an active session
  - Initial Sequence Number (ISN) randomization
  - ensure enhanced session management, namely, through:

Transport Layer Attacks and Mitigation

3. TCP Session Hijacking (cont.)

- TCP-AO – cryptographically authenticates TCP segments using MAC over TCP header for every packet; protects the TCP state machine
  - TCP Authentication Option, not widely deployed (RFC5925)
- Connection Token Validation – adds an extra layer of session-specific validation at the transport or application layer
  - e.g., token negotiated during handshaking, not part of standard TCP
- IPSec (Transport Mode) – provides encryption and integrity protection at the network layer, securing TCP sessions
- Middlebox Support – firewalls, IDS/IPS can monitor and protect the TCP session state

Transport Layer Attacks and Mitigation

1. UDP Flood (DoS/DDoS Attack)
   - attackers flood a target with large volumes of UDP packets, overwhelming the network or server, leading to resource exhaustion

- Mitigation:
  - Rate limiting – implement rate limits on incoming UDP traffic
  - DDoS protection services – filter out malicious traffic using
    - cloud-based (e.g. Cloudflare, Akamai Prolexic, AWS Shield)
    - network-based DDoS mitigation solutions (e.g., Verizon, AT&T,…)
    - open source (sw, tools): iptables/NetFilter; DDoS Deflate; Snort; Suricata

Transport Layer Attacks and Mitigation

2. UDP Amplification
   - attackers exploit UDP-based services (such as DNS or NTP) that respond to small queries with large responses
   - by sending forged requests with the target's IP address, the attacker amplifies the attack traffic, overwhelming the target with answers to queries it didn’t ask (“reflected attack”)

- Mitigation:
  - configure UDP services (e.g., DNS, NTP) to prevent amplification, using DNS Response Rate Limiting (DNS RRL) and ensuring that NTP servers are patched
    - RRL detects patterns in arriving queries, and when it finds a pattern that suggests abuse, it can reduce the rate at which the replies are sent (from BIND 9.10 on)
  - complement with egress filtering to prevent traffic from leaving your network with a spoofed source address (attacker inside the network domain)

Transport Layer Attacks and Mitigation

3. UDP and TCP Port Scanning

- Mitigation:
  - use a firewall to block unnecessary or unused ports
  - enable IDS/IPS to detect and block suspicious port scanning activity
- Slow Port Scanning – spread over hours and days; hard to detect
  - time-distributed probing (eludes rate-limiting security approaches)
  - low-traffic volume (looks like legitimate traffic)

  difficulties:
  - detection is resource intensive (long-term traffic analysis and continuous mon.)
  - even harder if slow scanning rate is adjusted dynamically
  - risk of false positives

Transport Layer - QUIC

QUIC - UDP-Based Multiplexed and Secure Transport

- secure general-purpose transport protocol, no native
- introduced by Google (2012), now a IETF standard RFC 9000 (2021)
- motivation: overcome TCP limitations (single stream, HOL blocking, unsecure and slow handshaking)
- runs over UDP for flexibility and user space control (instead of kernel)
- core features: secure handshaking, flow control, congestion control, error control, reliable delivery, 0-RTT, stream-based (ordered sequence of bytes), multiple streams per connection (multiplexing)
- integrated as a Chromium component, supported by major browsers

Transport Layer - QUIC

- HTTP/3 built on top of IETF QUIC

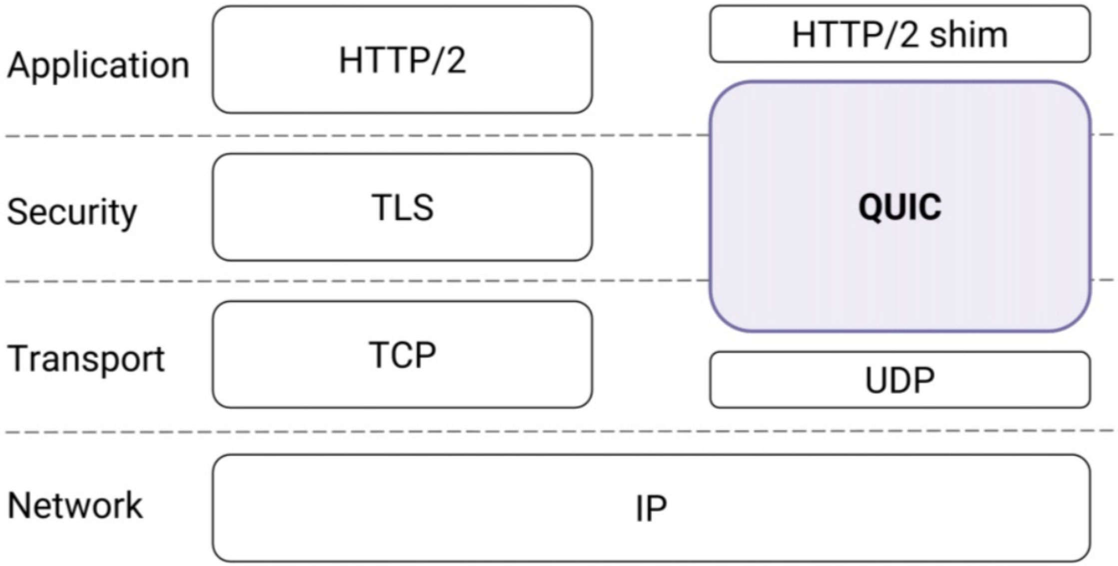
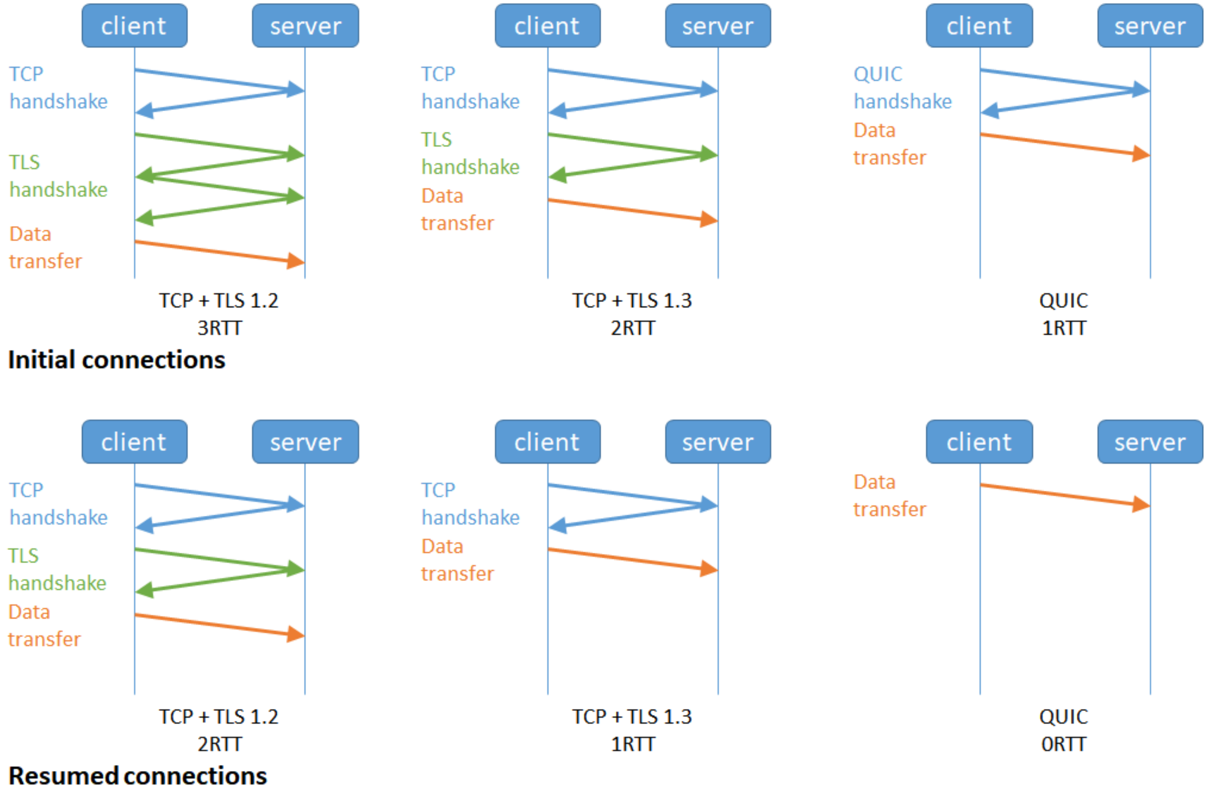


Figure 1: QUIC in the traditional HTTPS stack.

Transport Layer - QUIC

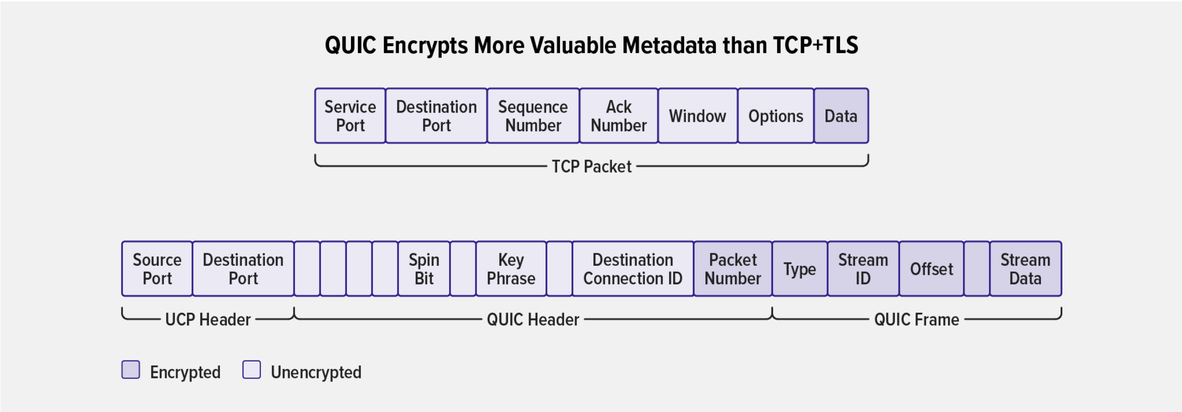
- handshaking – initial latency further reduced



Transport Layer - QUIC

QUIC - UDP-Based Multiplexed and Secure Transport

- each endpoint generates connection IDs (decoupling semantics from 5-tuple)
- stateful, connection-oriented protocol
- end-to-end transport protocol info is hidden from the network
- QUIC is not a replacement for TCP, app may use TCP if QUIC finds a fatal error
- better response time, security and privacy in detriment of interoperability



Transport Layer - QUIC

Security Features

- Built-in encryption
  - integrates TLS 1.3 natively, making encryption mandatory for all connections
  - forward secrecy and 0-RTT encryption: on session/connections resumption
- Reduced attack surface
  - (almost) no cleartext handshaking
  - encryption of metadata: unlike TCP, where metadata (seq #s, flags, etc.) is exposed, QUIC encrypts its headers, offering greater privacy
- Protecting against connection migration
  - connection migration: ability to maintain a session in presence of network changes (e.g., switching from Wi-Fi to mobile data), using Connection IDs
  - stateless reset: "silent" mechanism to detect and handle connection failures, e.g. if server state is lost

Transport Layer - QUIC

Security Features (more…)

- Resistance to replay attacks
  - although QUIC supports 0-RTT (more vulnerable to replay attacks that 1-RTT) for faster session resumption, it restricts the type of data (e.g., only idempotent requests) that can be sent during the 0-RTT phase
- Obfuscation of traffic characteristics
  - encrypted packet numbers
  - minimal exposure of connection metadata impairs traffic analysis and MitM
- Improved privacy
  - reduced latency and fingerprinting: 0-RTT and reduced amount of unencrypted handshake information (harder for external entities to fingerprint users)
  - connection ID randomization and renegotiation (`NEW_CONNECTION_ID` frames)

Transport Layer - QUIC

Security Features (more…)

- DoS protection
  - stateless server resumption: support for stateless server resumption via token sent during the handshake, reducing server-side memory for connection state
  - client-validated handshake: prevention of amplification attacks during the handshake by requiring clients to validate their IP addresses before the server commits significant resources to the connection
- Resistance to network surveillance and censorship
  - encrypts headers and control information, making it harder to block or censor
  - obfuscation of protocol versions, impairing protocol fingerprinting

Transport Layer Attacks and Mitigation

QUIC

1. Amplification attacks

- an attacker may send small initial packets with spoofed source addresses (claiming to be the victim) to a QUIC server, which then responds with larger handshake packets
  - although QUIC includes anti-spoofing mechanisms like stateless resets and address validation, improperly configured servers or excessive retries can make this attack feasible

- Mitigation:
  - implementing robust address validation mechanisms (e.g., QUIC retry packets), stateless resets, rate limiting

Transport Layer Attacks and Mitigation

QUIC

2. Reflection attacks

- like amplification attacks, reflection attacks involve sending requests to a server with a spoofed source IP address (that of a victim). The server then reflects its response back to the victim, overwhelming them with traffic
  - although QUIC includes client address validation before sending larger responses, servers that improperly validate source addresses or allow excessive retries could be vulnerable
- Mitigation:
  - QUIC servers should ensure proper client address validation, rate limit large responses, and implement packetretry tokens during the handshake to prevent reflection

Transport Layer Attacks and Mitigation

QUIC

3. Connection hijacking

- the use of encryption and Connection IDs make it harder to hijack sessions compared to TCP, however, an attacker could still attempt to intercept Connection IDs and guess seq# (very unlikely) and inject malicious packets into an active session

- Mitigation:
  - very unlikely to occur; impossible to spoof traffic without encryption keys

Transport Layer Attacks and Mitigation

QUIC

4. Packet reordering and delay attacks

- attackers may reorder packets or introduce artificial delays to degrade the performance of a QUIC connection (timeout and retries)

- Mitigation:
  - QUIC is resilient to packet reordering, but detecting abnormal levels of reordering or delays, on stable networks, could help identify potential attacks

5. Side-channel attacks

- attackers could observe patterns in encrypted traffic, such as packet sizes, timing, or metadata, to infer information about the connection

- Mitigation:
  - padding QUIC packets for obfuscating traffic patterns, randomizing Connection IDs, use time-constant processing on specific QUIC packet fields

Transport Layer Attacks and Mitigation

QUIC

6. Downgrade attacks
    - QUIC is designed to enforces TLS 1.3; there’s a possibility of attempting a downgrade to an insecure version of TLS or a less secure cipher

- Mitigation:
    - QUIC is resistant to downgrade attacks; proper configuration of cipher suites and protocols on both client and server sides can further prevent downgrades

7. Resource exhaustion attacks
    - an attacker could initiate many uncomplete handshake requests, causing the server to allocate resources without establishing actual connections (resource exhaustion)

- Mitigation:
    - stateless retries and client address validation prevent servers from being overwhelmed
    - servers limit resource allocation for incomplete handshakes

Transport Layer Attacks and Mitigation

QUIC

8. Replay attacks (0-RTT)
    - an attacker could capture and replay a client’s 0-RTT data to the server, potentially causing the server to process the same request multiple times
- Mitigation:
    - sensitive operations (e.g., transactions) should not be processed in 0-RTT mode. QUIC servers store session tickets to detect and prevent replayed data

9. Traffic correlation attacks
    - an attacker monitoring network traffic might correlate connections based on Connection ID patterns, packet sizes, and timing, even if the content is encrypted, which may lead to deanonymization
- Mitigation:
    - Connection ID rotation and traffic obfuscation techniques reduces risks

Transport Layer Attacks and Mitigation

QUIC

Real concern?

- Amplification and reflection attacks: yes, if misconfigured or improperly deployed
- Resource exhaustion (DoS) attacks: yes, especially for high-traffic services
- Replay attacks (in 0-RTT): low-moderate, especially for critical services or financial transactions
- Side-channel and traffic correlation attacks: low, especially for privacy-sensitive applications.
- Implementation bugs and misconfigurations: yes

- In summary:
  - if QUIC is properly configured and regularly maintained, the risk of attacks is very low compared to older protocols like TCP with TLS 1.2

Transport Layer Attacks and Mitigation

QUIC

Additional info:

- QUIC IETF WG
  - https://datatracker.ietf.org/wg/quic/about/
- Current implementations (2025):
  - https://github.com/quicwg/quicwg.github.io/blob/main/implementations.md
- Further reading:
  - Adam Langley, et al. “*The QUIC Transport Protocol: Design and Internet-Scale Deployment*”. In ACM SIGCOMM '17. ACM, New York, NY, USA, 183–196. DOI: https://doi.org/10.1145/3098822.3098842
  - IETF RFC 9000, May 2021, https://datatracker.ietf.org/doc/html/rfc9000