Testing QUIC with packetdrill

Vidhi Goel, Rui Paulo, Christoph Paasch
Apple Inc

SIGCOMM EPIQ
August 14 2020
Is QUIC ready to ship?

- Unit testing
- Inter-op testing between ~20 implementations
- Performance sanity of HTTP/3 vs HTTP/2
- Initial deployment for experimentation
- Is the industry ready?
Transport protocols are complex

QUIC is even harder

- Header and packet protection
- Authenticated handshake
- Multiple streams within a connection
- Large set of transport parameters and frames
- Built-in mobility; and more …
Testing Methods

- Interoperability / Performance testing
- Protocol fuzzing
- Failure testing
- Conformance testing
- Longevity / Stress testing
- Reproducible integration testing
Reproducible & precise Integration testing

Transport protocol stack

- Time $T$
- Time $(T + X)$
- Packet (pkt)
- Acknowledgment (ack)
packetdrill

• Scripting tool developed by Google

• Specify a set of events with timestamps
  • system calls, packets, shell commands, python script

• Write precise, reproducible and automated scripts

• Easy integration of new protocol options
packetdrill for QUIC

- QUIC library
  - quic_connect()
- Socket API
- UDP / IP
- Local Interface
- TLS library
  - read Initial
  - inject Initial
Example script for QUIC handshake

+0 quic_create (…, IPPROTO_QUIC) = 3
+0 quic_connect (3, …, ...) = 0

+0 quic (initial, dcid=0x1, pn=0 […])
+0 quic (handshake, dcid=0x02, pn=0 […])
+0 quic (handshake, dcid=0x02, pn=1 […])
+0 quic (application, dcid=0x1, pn=0)
Integrating QUIC into packetdrill

- QUIC packet grammar
- TLS handshake
- Packet parsing and verification
- Packet injection
QUIC packet grammar

**packet**

`packet_prefix QUIC (q_header): q_frame_list`

**q_header**

`q_packet_type, header_field1=<value> [...]`

**q_frame_list**

`q_frame [; q_frame[...]]`

**q_frame**

`q_frame_type [field1=<value> [...]]`
QUIC packet examples

// Client Initial packet
+0 > quic (initial, dcid=0x1, scid=0x2, pn=0):
    CRYPTO[offset=0, length=512];
    PADDING[length=640]

// Injected server initial, transport params are specified in CRYPTO frame
+0.1 < quic (initial, dcid=0x2, scid=0x1, pn=0):
    CRYPTO[offset=0, length=122, initial_max_stream_data_bidi_remote=5000];
    ACK[largest=0, delay=10, range_count=0, range0=0]
TLS handshake
Packet parsing and verification

- QUIC library
- Packet Protection
- Packet Parser
- Verifier

Local Interface

- PADDING
- CRYPTO
- dcid, scid, packet number

- QUIC library packet
- Script packet

+0 > quic (initial ... )
Packet injection

- QUIC library
- Packet Protection
- Packet mapping
- Wait for specified time

Local Interface

QUIC library packet
Script packet
QUIC test scripts

- Over 50 scripts and growing
- Scripts for handshake, flow control, streams, loss recovery, congestion control, PMTU discovery…
- Continuous integration and automation testing
- Use during development, regression testing & troubleshooting
Adopting a second QUIC library

- Send / receive abstraction
- Socket API
- UDP / IP
- Local Interface

- quiche_connect()
- packetdrill
- TLS library

Timer:
- read Initial
- send Initial

QUICHE library

Socket API

UDP / IP

Local Interface
Experience with QUICHE

- Easy to integrate, less than 300 lines of source code
- Reuse same test scripts for a different library
- Found issues and worked with Cloudflare to fix them
Challenges

- CPU time for TLS handshake may be variable
  - Variance introduces instability in test results
  - Use tolerance and time intervals

- **Script MUST start with QUIC handshake**
  - QUIC handshake is lengthy to write - can create inconsistencies
  - Include a handshake template

- Multiple draft versions
  - Continue to add support for newer draft
  - Specify ALPN through QUIC library API to set client version
Conclusion

• Packetdrill provides us an opportunity to test the complex protocol state machines.

• Reuse code & scripts for any QUIC library

• Testing QUIC with packetdrill will help us achieve higher quality for our QUIC implementations
Thank You!

Any questions?