QUIC
A New Internet Transport
Presenter: Jana Iyengar
<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov 2013</td>
<td>Early design and experience (TSVAREA)</td>
</tr>
<tr>
<td>Mar 2015</td>
<td>QUIC handshake (SAAG)</td>
</tr>
<tr>
<td>Mar 2015 onwards</td>
<td>Replacing QUIC's handshake with TLS1.3</td>
</tr>
<tr>
<td>July 2015</td>
<td>BarBoF, experimental results</td>
</tr>
<tr>
<td>Nov 2015</td>
<td>Cubic bug in QUIC, TCP (TCPM)</td>
</tr>
<tr>
<td>July 2016</td>
<td>BoF</td>
</tr>
</tbody>
</table>
The QUIC Experiment

HTTP/2
TLS
TCP
HTTP over QUIC
QUIC Experiment (so far)
UDP
IP
The IETF Proposal

HTTP/2
TLS
TCP
HTTP over QUIC
QUIC
TLS 1.3
TCP-like congestion control, loss recovery
UDP
IP
Standardized QUIC

HTTP/2

TLS

TCP

QUIC

Crypto handshake

TCP-like congestion control, loss recovery

UDP

IP

Application
QUIC Design Aspirations

- Deployability and evolvability
QUIC Design Aspirations

- Deployability and evolvability
- Low latency connection establishment
QUIC Design Aspirations

- Deployability and evolvability
- Low latency connection establishment
- Multistreaming and per-stream flow control
QUIC Design Aspirations

- Deployability and evolvability
- Low latency connection establishment
- Multistreaming and per-stream flow control
- Better loss recovery and flexible congestion control
QUIC Design Aspirations

- Deployability and evolvability
- Low latency connection establishment
- Multistreaming and per-stream flow control
- Better loss recovery and flexible congestion control
- Resilience to NAT-rebinding
QUIC Design Aspirations

- Deployability and evolvability
- Low latency connection establishment
- Multistreaming and per-stream flow control
- Better loss recovery and flexible congestion control
- Resilience to NAT-rebinding
- Multipath for resilience and load sharing
Deployability and Evolvability

Uses UDP as the substrate
enables deployment through various middleboxes
userspace implementation enables rapid deployment
Deployability and Evolvability

Uses UDP as the substrate
   enables deployment through various middleboxes
userspace implementation enables rapid deployment

Version negotiation
   enables protocol wire format evolution
Deployability and Evolvability

Uses UDP as the substrate
   enables deployment through various middleboxes
userspace implementation enables rapid deployment

Version negotiation
   enables protocol wire format evolution

Fully authenticated and mostly encrypted headers
   avoids network ossification
QUIC Streams

Multiplexed streams within a transport connection multiple streams avoids HoL blocking shared congestion control and loss recovery two levels of flow control: stream and connection
Congestion Control & Loss Recovery

QUIC builds on decades of experience with TCP
Congestion Control & Loss Recovery

QUIC builds on decades of experience with TCP

Incorporates TCP best practices
  TCP-like congestion control (NewReno, Cubic)
Congestion Control & Loss Recovery

QUIC builds on decades of experience with TCP

Incorporates TCP best practices

TCP-like congestion control (NewReno, Cubic)

FACK, TLP, F-RTO, Early Retransmit, …

(also, time-based loss detection)
QUIC builds on decades of experience with TCP

Incorporates TCP best practices
- TCP-like congestion control (NewReno, Cubic)
  - FACK, TLP, F-RTO, Early Retransmit, …
  - (also, time-based loss detection)

Richer signaling than TCP
Richer Signaling Than TCP

Retransmitted packets consume new sequence number
no retransmission ambiguity
prevents loss of retransmission from causing RTO
Richer Signaling Than TCP

Retransmitted packets consume new sequence number
no retransmission ambiguity
prevents loss of retransmission from causing RTO

More verbose ACK
TCP supports up to 3 SACK ranges
QUIC supports up to 256 ACK ranges
explicit packet receive times
enables ACK decimation
QUIC Implementations

Chromium (open source)
https://cs.chromium.org/chromium/src/net/quic/

quic-go (open source implementation in Go)
https://github.com/lucas-clemente/quic-go

Christian Huitema's implementation
## Debugging Tools: Wireshark

![Wireshark Screenshot](image)

### Filter: QUIC

<table>
<thead>
<tr>
<th>No.</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>985</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>3182875774876983667, Seq: 1</td>
</tr>
<tr>
<td>986</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>3182875774876983667, Seq: 2</td>
</tr>
<tr>
<td>989</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>3182875774876983667, Seq: 2</td>
</tr>
<tr>
<td>990</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>79</td>
<td>3182875774876983667, Seq: 3</td>
</tr>
<tr>
<td>991</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>3182875774876983667, Seq: 4</td>
</tr>
<tr>
<td>992</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>350</td>
<td>3182875774876983667, Seq: 5</td>
</tr>
<tr>
<td>993</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>85</td>
<td>3182875774876983667, Seq: 3</td>
</tr>
<tr>
<td>994</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>353</td>
<td>3182875774876983667, Seq: 4</td>
</tr>
<tr>
<td>995</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1412</td>
<td>2735399108252988334, Seq: 1</td>
</tr>
<tr>
<td>997</td>
<td>216.58.216.238</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>2069061289831796684, Seq: 1</td>
</tr>
<tr>
<td>998</td>
<td>173.194.46.76</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>16164325528471686122, Seq: 1</td>
</tr>
<tr>
<td>999</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>91765324381928584, Seq: 1</td>
</tr>
<tr>
<td>1000</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>91765324381928584, Seq: 2</td>
</tr>
<tr>
<td>1001</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>91765324381928584, Seq: 3</td>
</tr>
<tr>
<td>1002</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>91765324381928584, Seq: 4</td>
</tr>
<tr>
<td>1003</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>91765324381928584, Seq: 5</td>
</tr>
<tr>
<td>1004</td>
<td>173.194.46.73</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>91765324381928584, Seq: 6</td>
</tr>
<tr>
<td>1005</td>
<td>216.58.216.238</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>2069061289831796684, Seq: 1</td>
</tr>
<tr>
<td>1006</td>
<td>173.194.46.76</td>
<td>10.1.1.14</td>
<td>QUIC</td>
<td>1392</td>
<td>16164325528471686122, Seq: 1</td>
</tr>
</tbody>
</table>

**Frame 981:** 1392 bytes on wire (11136 bits), 1392 bytes captured (11136 bits) on interface 0 (outbound)
- **Internet Protocol Version 4, Src: 10.1.10.14 (10.1.10.14), Dst: 173.194.46.73 (173.194.46.73)**
- **User Datagram Protocol, Src Port: 51863 (51863), Dst Port: 80 (80)**
- **QUIC (Quick UDP Internet Connections)**
- **Public Flags: 0x00**
- **CID: 3182875774876983667**
- **Version: 002**
- **Sequence: 1**
- **Payload: 9f@da5bbae0672d965b22dc01a0e100443484c4f130000...**
Debugging Tools: Chrome

chrome://net-internals