# QUIC

## Redefining Internet Transport

**Presenter:** Jana Iyengar

Google

# QUIC

## Re[illegible] Internet Transport

Reinventing?

**Presenter:**

Jana Iyengar

Google

# QUIC

## Red g Internet Transport

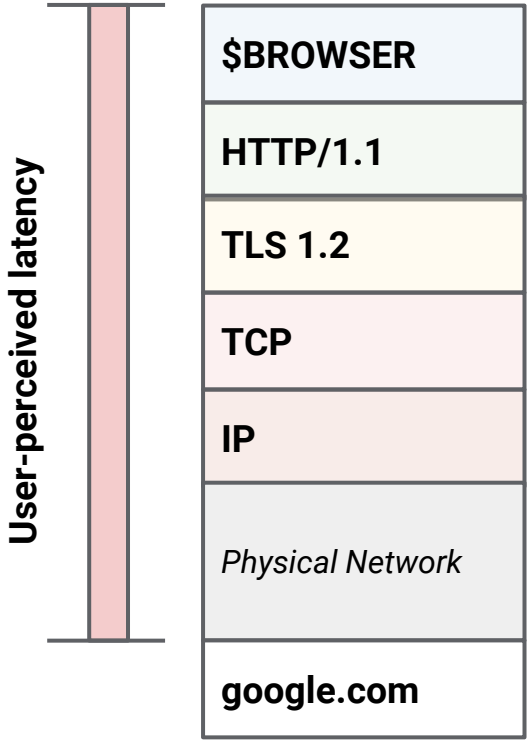
Doing

Right!

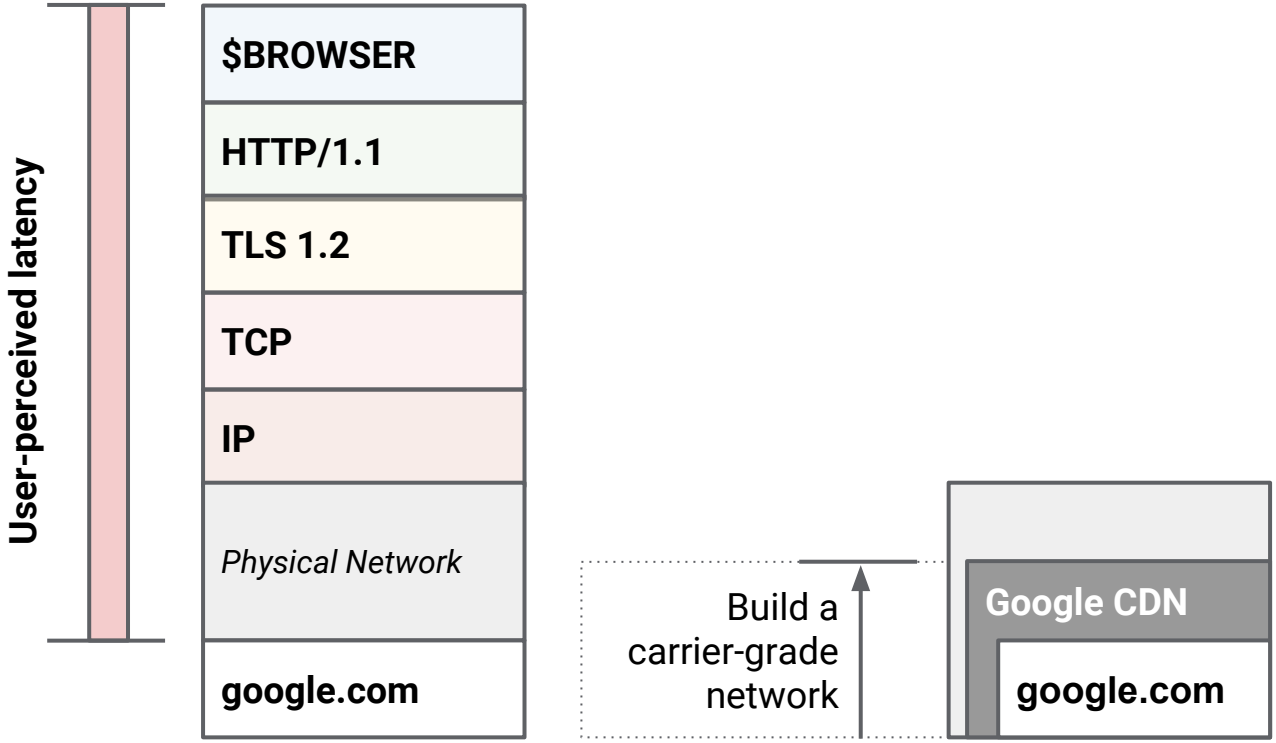
**Presenter:**

Jana Iyengar

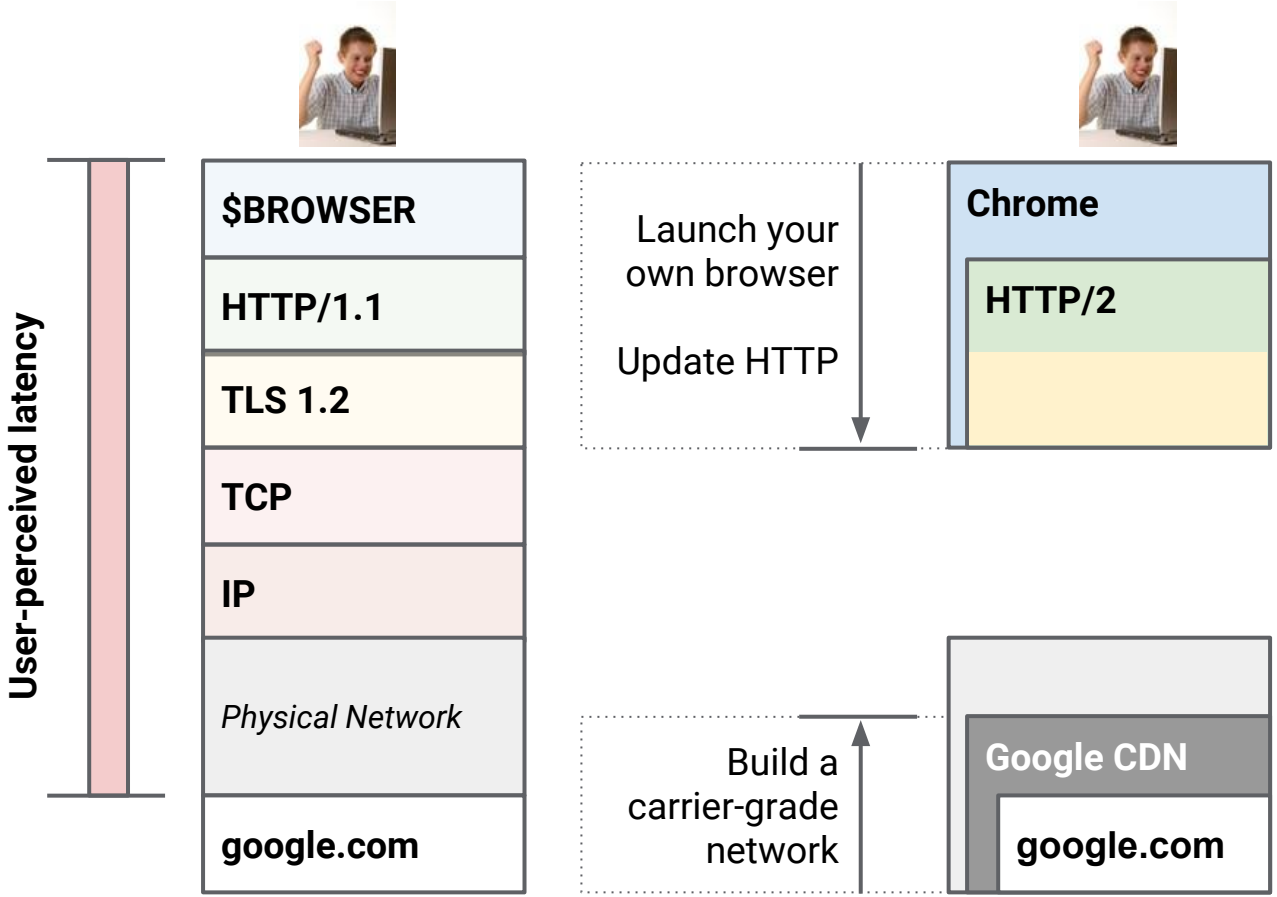
# How do you make the web faster?



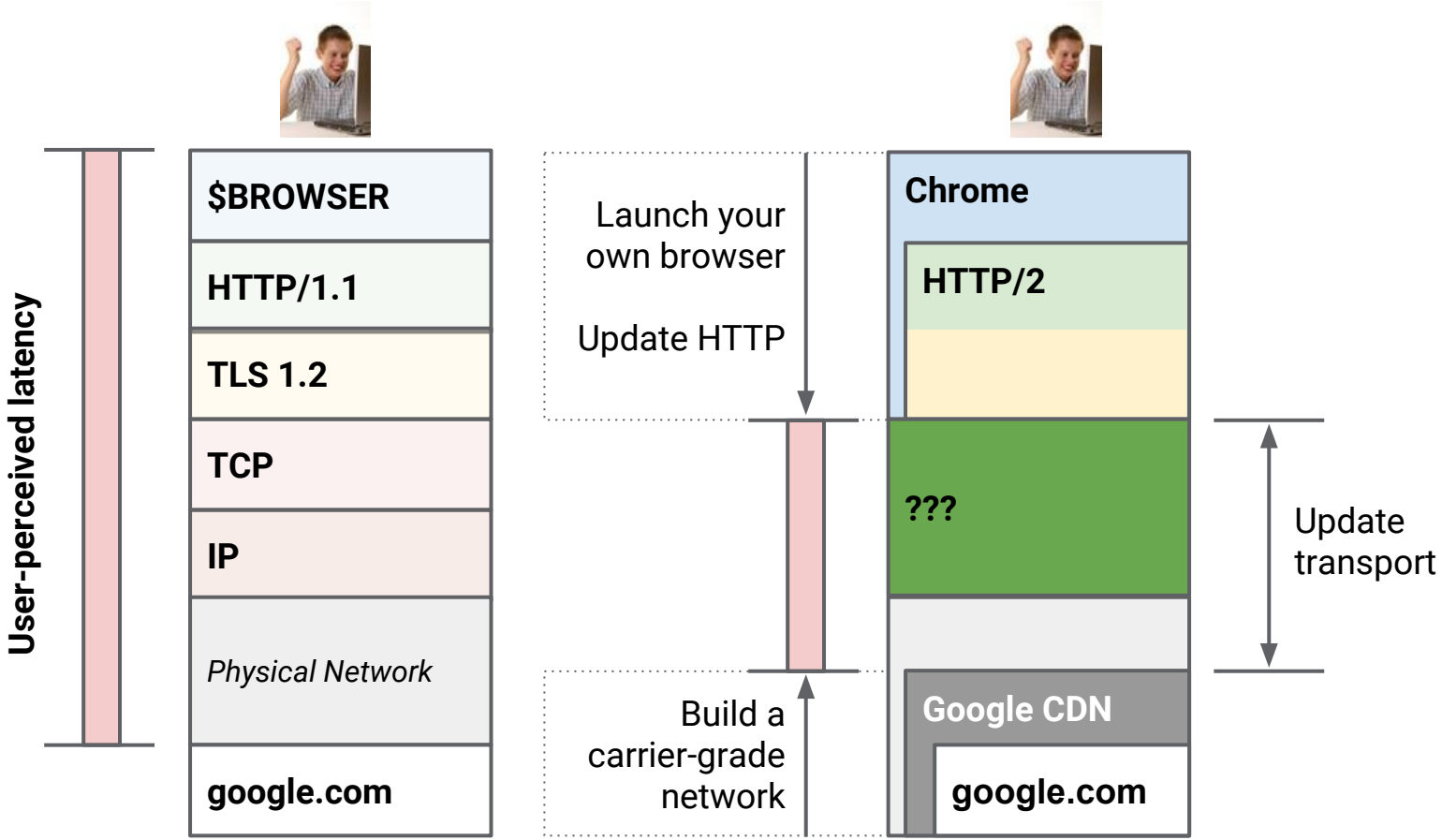
# How do you make the web faster?



# How do you make the web faster?



# How do you make the web faster?



# What is QUIC?

# QUIC

## **Q**uick **U**DP **I**nternet **C**onnections

- A reliable, multiplexed transport over UDP
- Always encrypted
- Reduces latency
- Runs in user-space
- Open sourced in Chromium

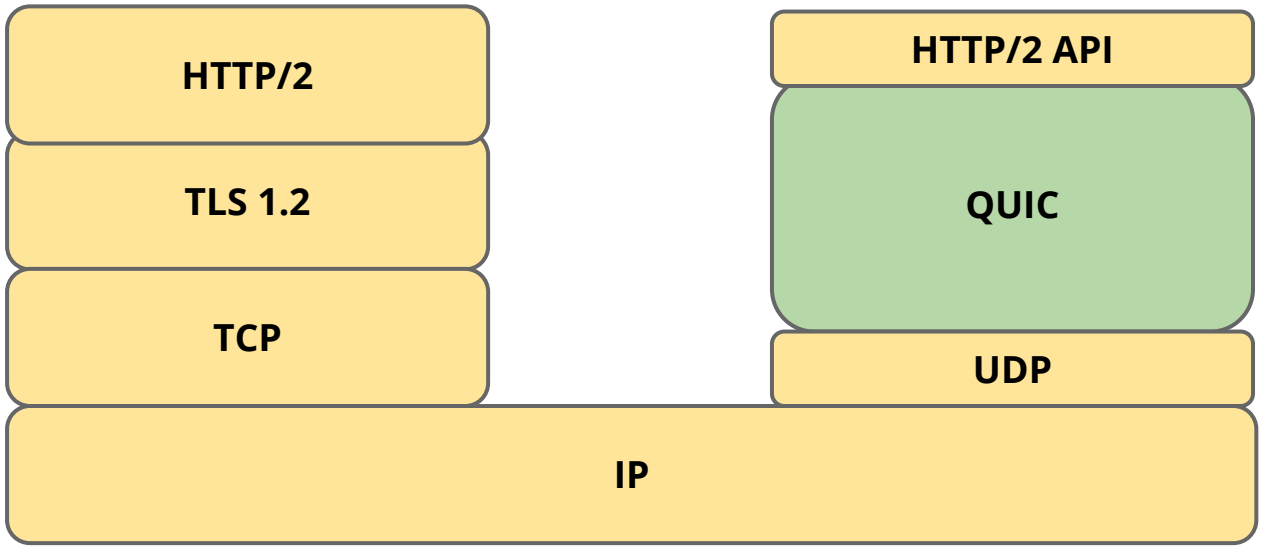
# What is QUIC?

**New transport designed to reduce web latency**

- TCP + TLS + SPDY over UDP
- Faster connection establishment than TLS/TCP
  - 0-RTT usually, 1-RTT sometimes
- Deals better with packet loss than TCP
- Has Stream-level and Connection-level Flow Control
- FEC recovery
- Multipath

*except for HTTP/2 headers, which should be fixed as well.

# Where does it fit?



# Always encrypted

**Comparable to TLS**
Perfect forward secrecy, with more efficient handshake

**IP spoofing protection**
Signed proof of address

**Inspired TLS 1.3's 0-RTT handshake**
Plan to adopt TLS 1.3 when complete

**more crypto details...**

# Connection establishment

**Connection identified by Connection ID**

- As opposed to common 5-tuple
- 64 bits
- Chosen randomly by the client
- Enables connection mobility across IP, port

# 0-RTT connection establishment

TCP

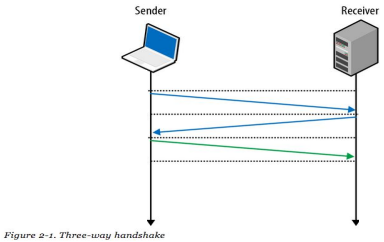
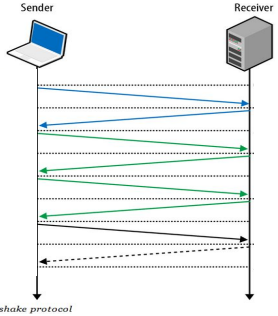



*Figure 2-1. Three-way handshake*

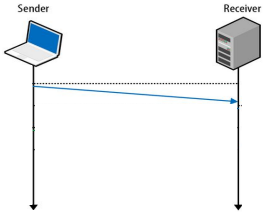
TCP + TLS




*2 4-2. TLS handshake protocol*

QUIC

(equivalent to TCP + TLS)



# First-ever connection - 1 RTT

**No cached information available**

**First CHLO is inchoate (empty)**

- Simply includes version and server name

**Server responds with REJ**

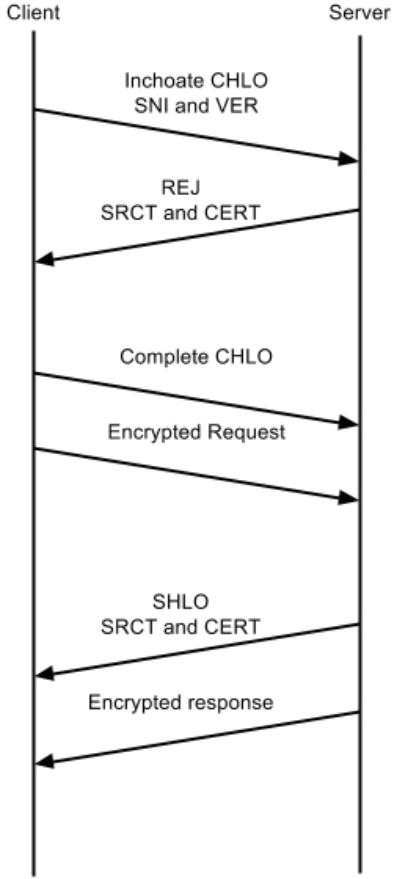
- Includes server config, certs, etc
- Allows client to make forward progress

**Second CHLO is complete**

- Followed by initially encrypted request data

**Server responds with SHLO**

- Followed immediately by forward-secure encrypted response data



# Subsequent connections - 0 RTT

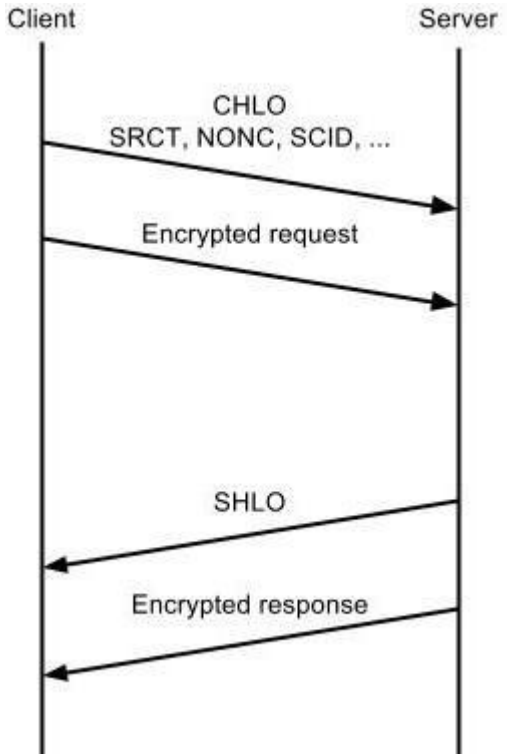
**First CHLO is complete**

Based on information from previous connection

Followed by initially encrypted data.

**Server responds with SHLO**

Followed immediately by forward-secure encrypted data



# Congestion control & reliability

**QUIC builds on decades of experience with TCP**

**Incorporates TCP best practices**

TCP Cubic - fair with TCP

FACK, TLP, F-RTO, Early Retransmit...

**More flexibility going forward**

Improved congestion feedback, control over acking

**Better signaling than TCP**

# Better signaling than TCP

**Retransmitted packets consume new sequence number**

- No retransmission ambiguity
- Prevents loss of retransmission from causing RTO

**More verbose ACK**

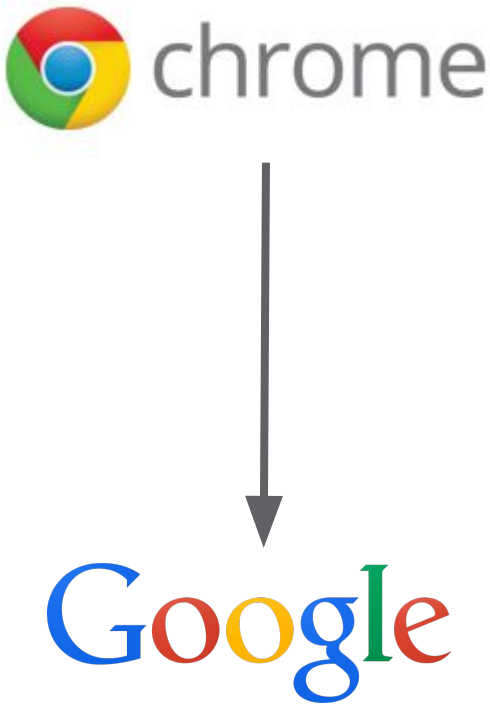
- TCP supports up to 3 SACK ranges
- QUIC supports up to 256 NACK ranges
- Per-packet receive times, even with delayed ACKs

**ACK packets consume a sequence number**

# Effective

## How quick is QUIC?

# Measuring performance




**Controlled Experiments**

**Client Side**
Latency, Bandwidth, Quality of Experience, Errors

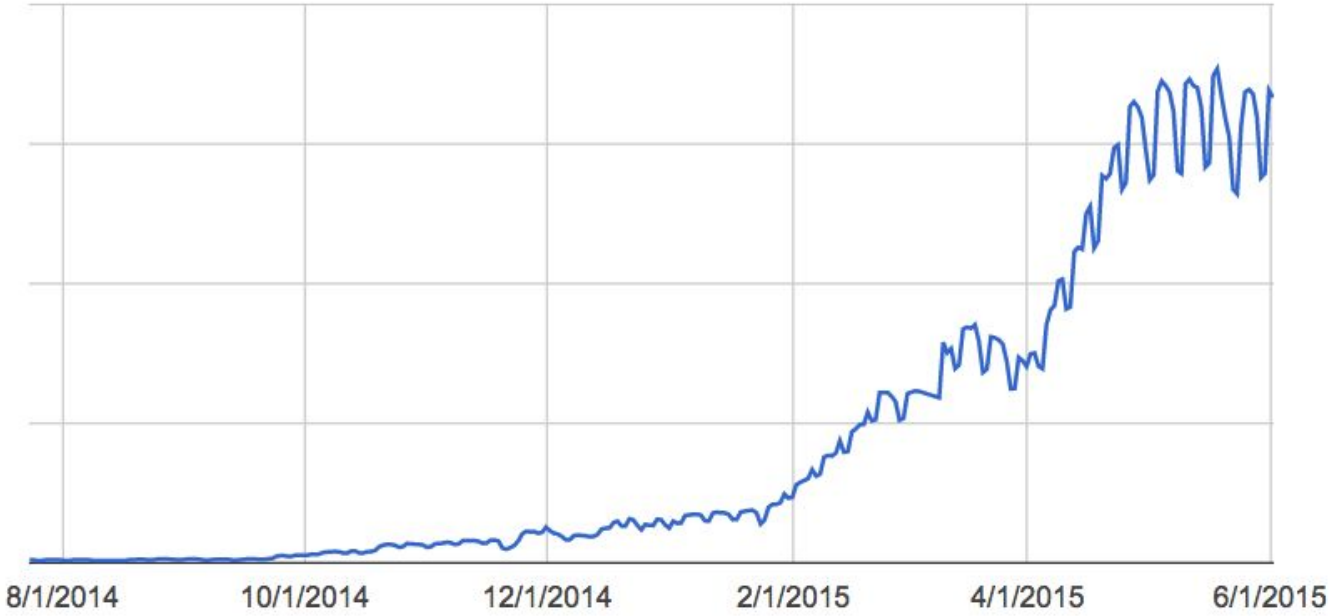
**Server Side**
Latency, Bandwidth, QUIC Success Rate

**Fine Grained Analysis**
By ASN, Server, OS, Version

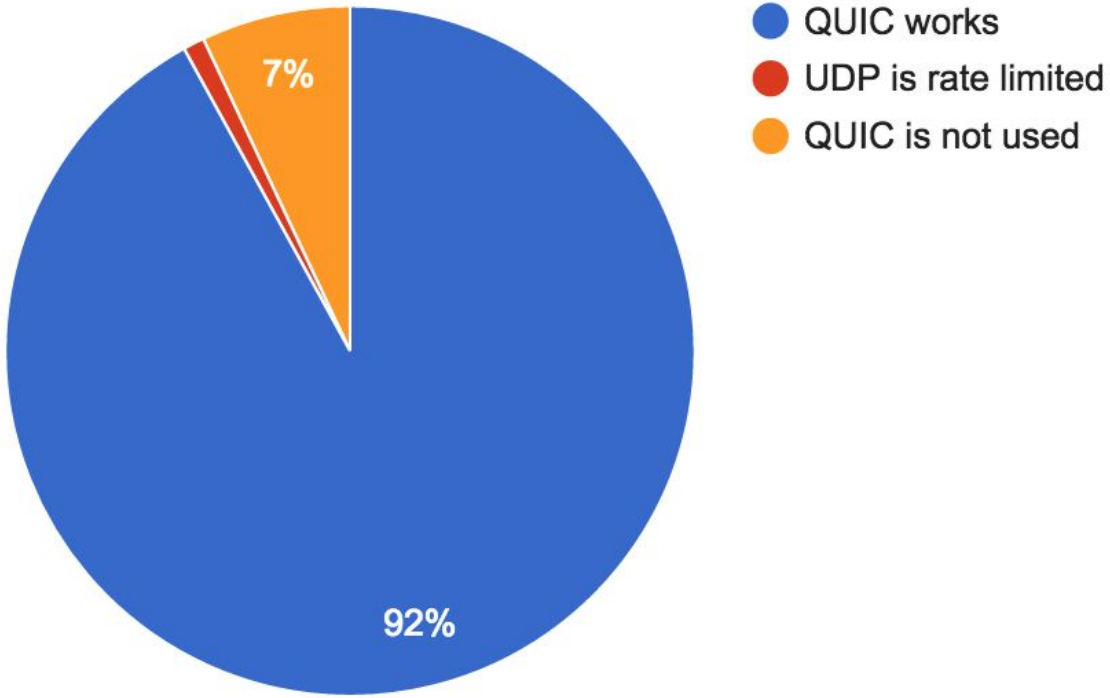
# Deployment timeline

**Tested at scale, with millions of users**

- Chrome Canary: June, 2013
- Chrome Stable: April, 2014
- Ramped up for Google traffic in 2015



# QUIC: Does it work?




QUIC handshakes fail when RTTs are greater than 2.5 seconds or when UDP is blocked

# Performance on Google properties

**Faster page loading times**

- 5% faster on average
- 1 second faster for web search at 99th-percentile

**Improved YouTube Quality of Experience**

- 30% fewer rebuffers (video pauses)

Recent Blog Post

# Where are the gains from?

**0-RTT**

- Over 50% of the latency improvement (at median and 95th-percentile)

**Improved loss recovery**

- Over 10x fewer timeout based retransmissions improve tail latency and YouTube video rebuffer rates

**Other, smaller benefits**

- e.g. head of line blocking, more efficient framing

# Safe

## What we're doing to protect users and networks

# Client-side protection

**What if UDP is blocked?**

- Chrome seamlessly falls back to HTTP/TCP

**What if the path MTU is too small?**

- QUIC handshake fails, Chrome falls back to TCP

**What if a client doesn’t want to use QUIC?**

- Chrome flag / administrative policy to disable QUIC

# When client-side protection is not enough...

**As a last resort, Google disables QUIC to specific ASNs**

- This is used as a fallback to protocol features

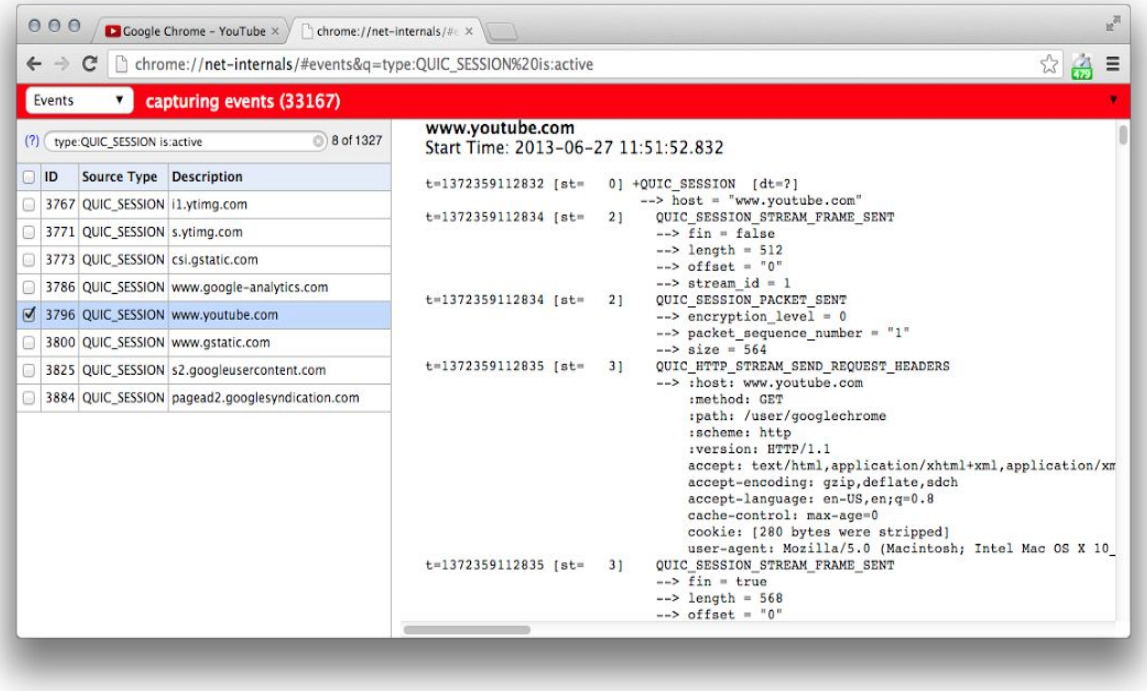
**Why do we disable QUIC delivery?**

- Degraded quality of experience measured
- Indications of UDP rate limiting at peak times of day
- End user reports (via chromium.org)

# Debugging Tools: Chrome

**chrome://net-internals**

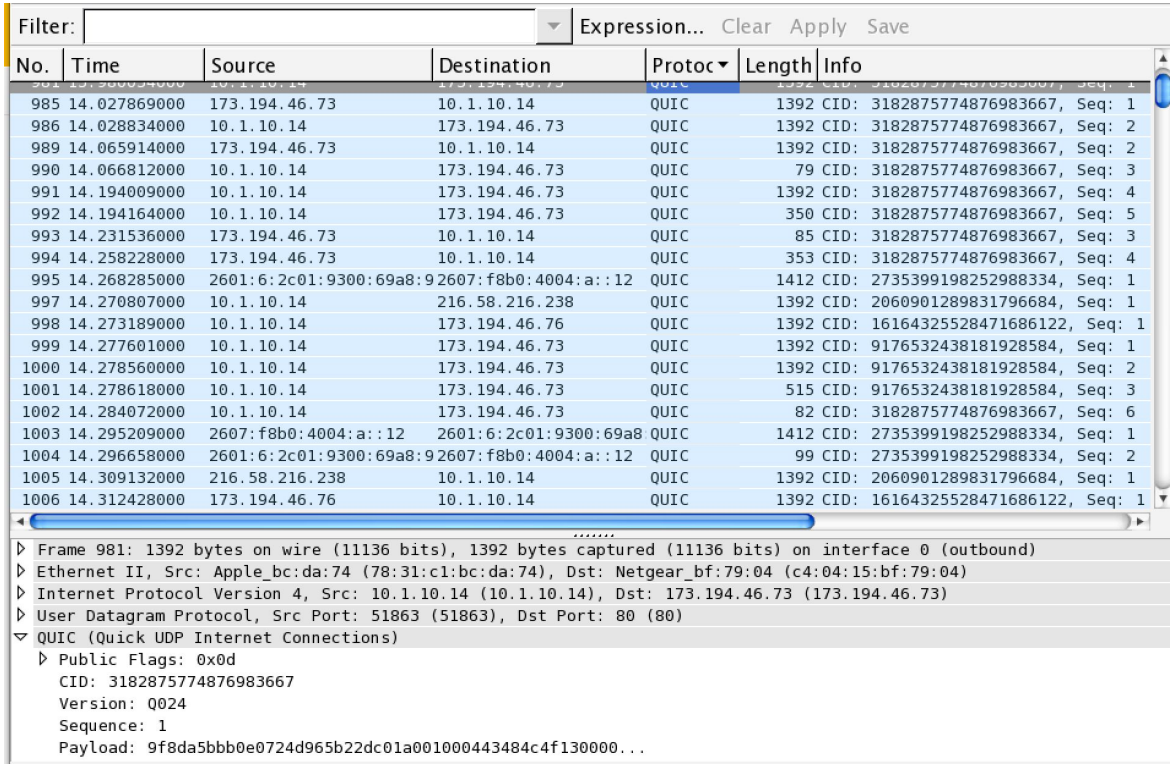
- Active QUIC sessions
- Captures all events
- Important for filing Chromium bugs



# Debugging Tools: Wireshark

## Parses

- Protocol: QUIC
- CID: Connection ID
- Seq: Sequence number
- Version: ie: Q024
- Public flags: 1 byte
- Payload: Encrypted



# What's Next?

# Future Improvements

- Forward Error Correction
- Connection Mobility
- Multipath
- More congestion control experiments

# Open source implementations

**Servers**

- Open source test server included in Chromium
- Working with other server vendors

**Clients**

- Open source Chromium client library for desktop and mobile
- Google Chrome and some Google Android apps
- Working with other browsers

# QUIC at the IETF

| | |
|---|---|
| **Nov 2013** | Initially Presented |
| **Mar 2015** | QUIC Crypto |
| **July 2015** | BarBoF Now! |
| **Ongoing** | Subsuming QUIC's 0-RTT handshake in TLS1.3 |

# Review: QUIC Summary

- Reliable, multiplexed transport
- Runs over UDP
- Always encrypted
- Lower latency connection establishment
- Optional FEC
- Rapidly evolving user-space implementation
- Open source

# QUIC

**Source**: QUIC in Chromium
**Page:** www.chromium.org/quic
**Public Mailing list:** proto-quic@chromium.org
**IETF draft**: draft-tsvwg-quic-protocol-01