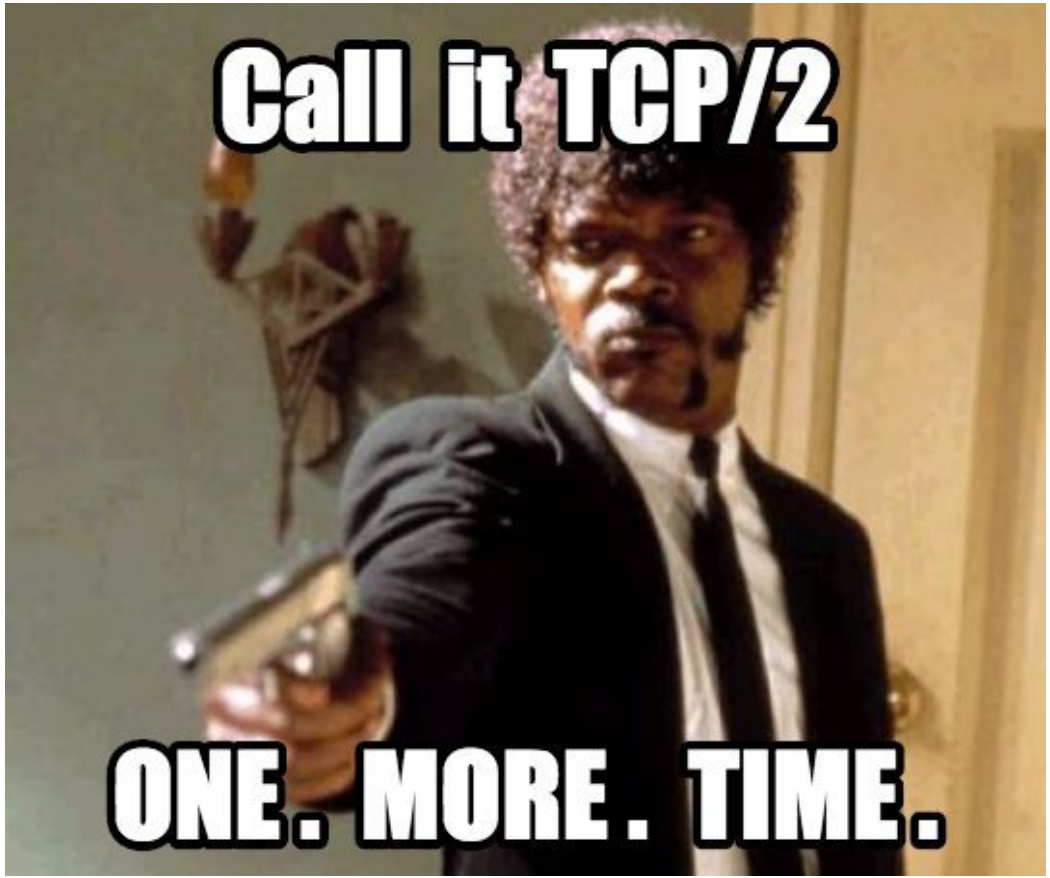
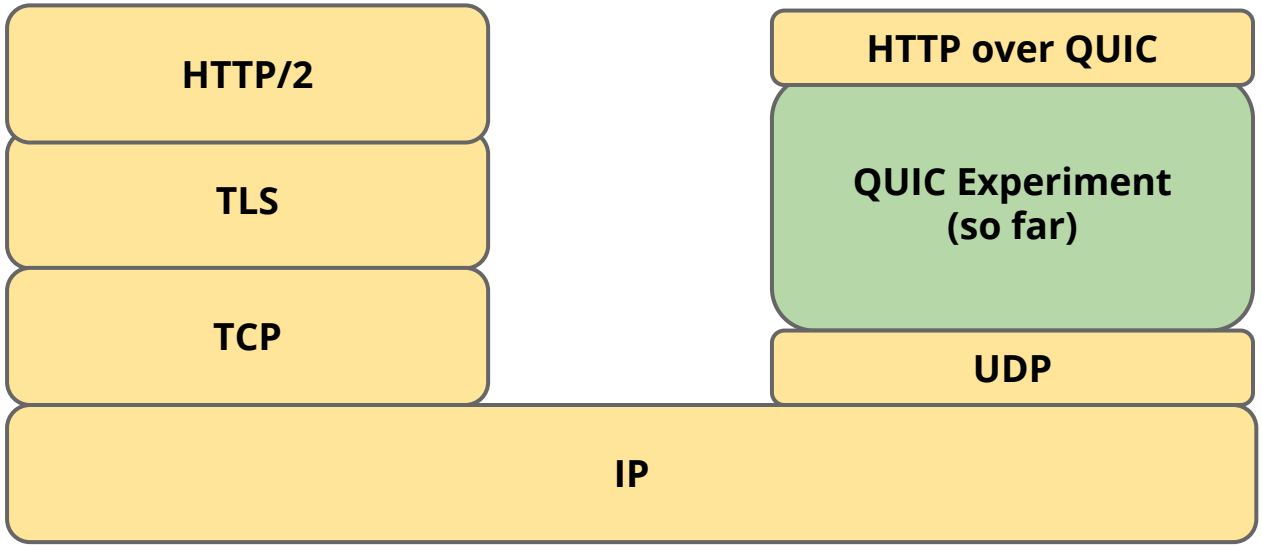
QUIC

A New Internet Transport

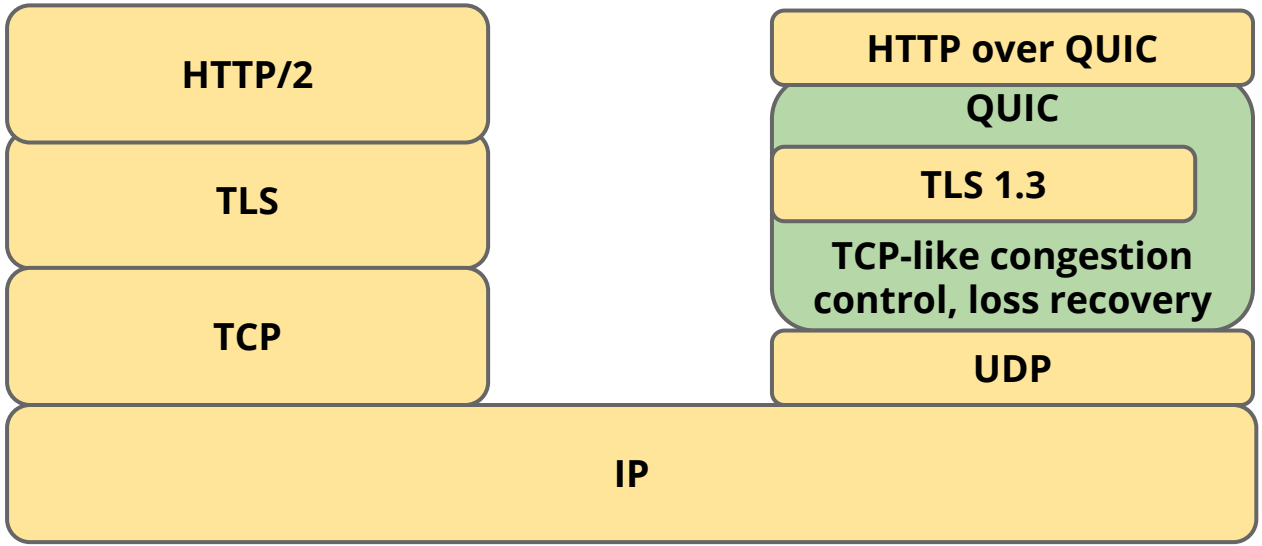
Presenter: Jana Iyengar


Call it TCP/2
ONE. MORE. TIME.

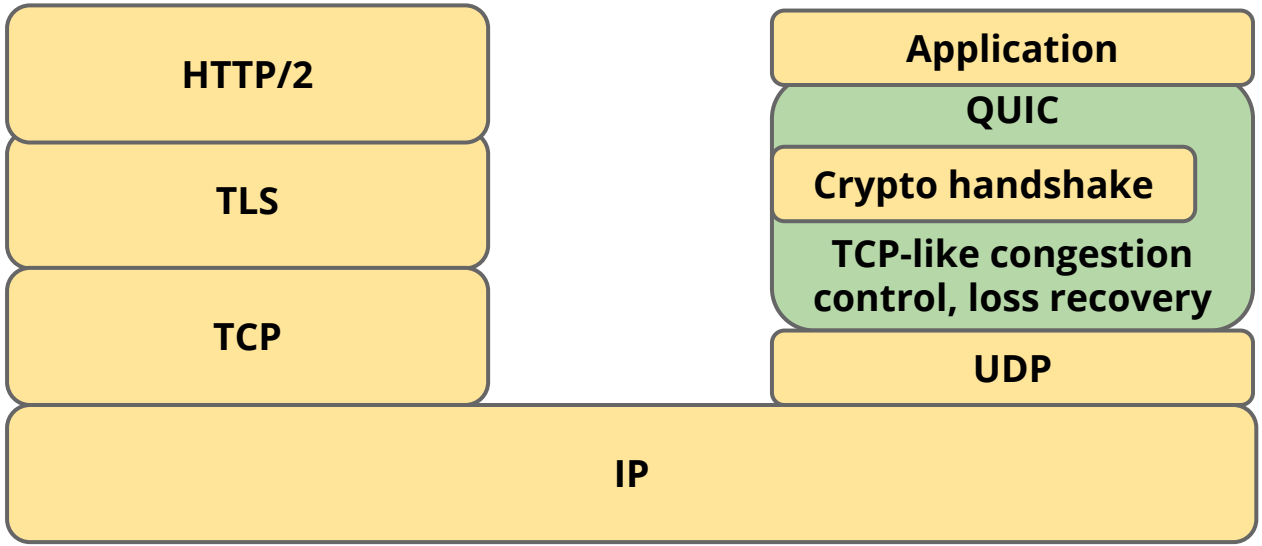
The QUIC Experiment



The IETF Proposal



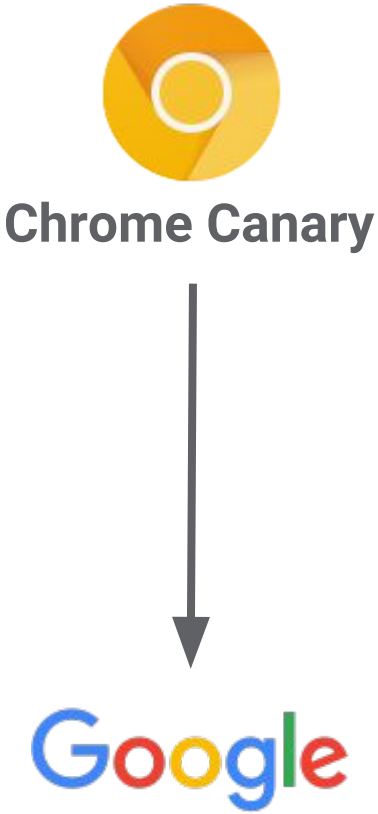
Standardized QUIC



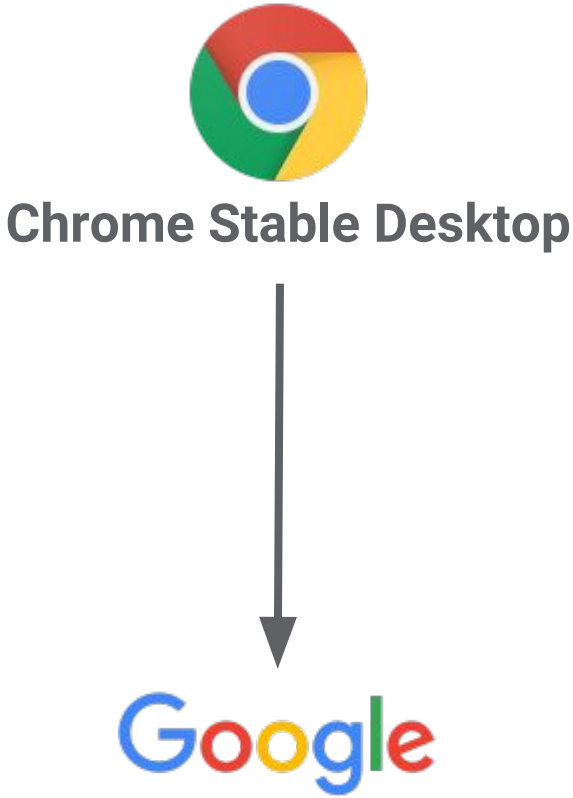
QUIC Design Aspirations

- 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

Deployment timeline: June, 2013

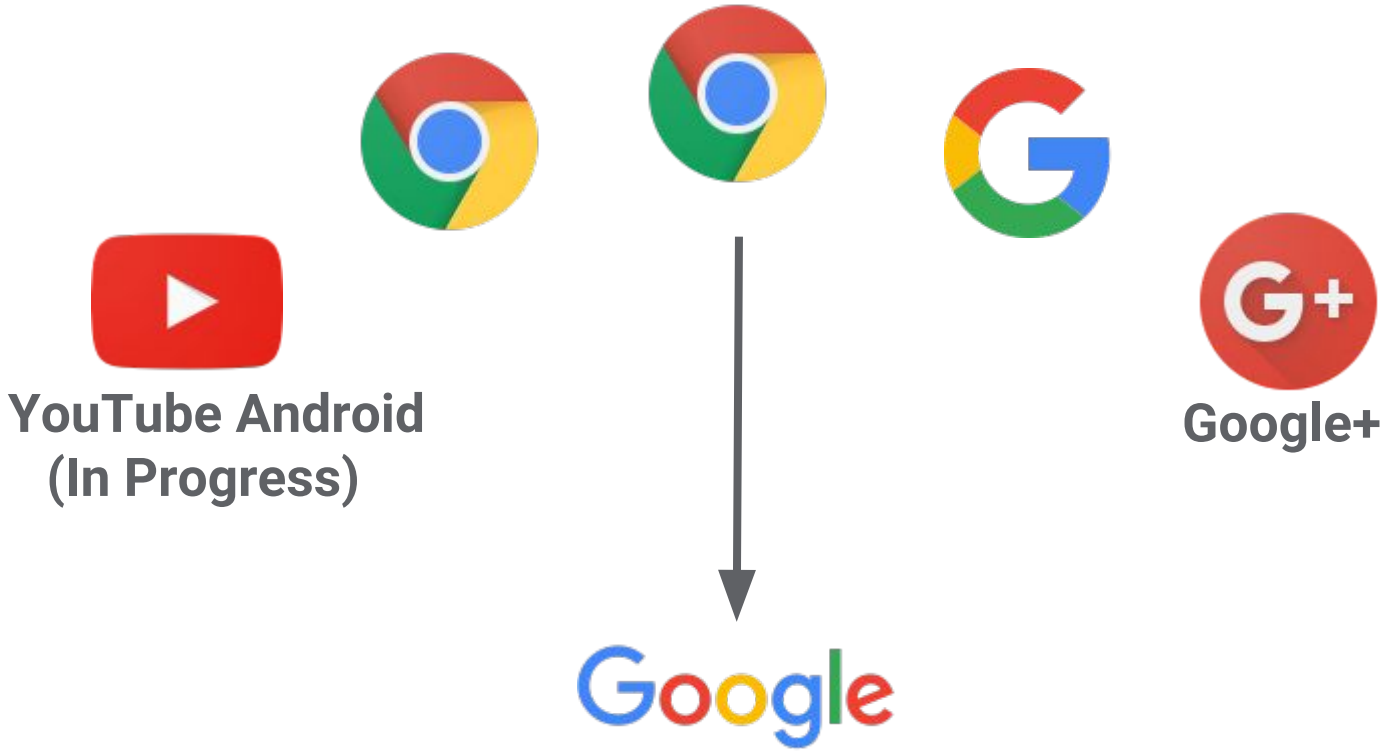


Deployment timeline: April, 2014



Deployment timeline: 2015

Deployment timeline: 2016

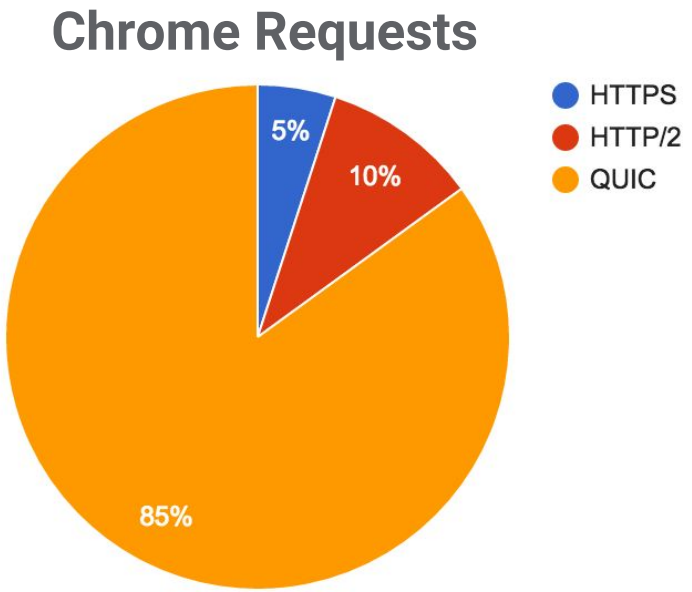


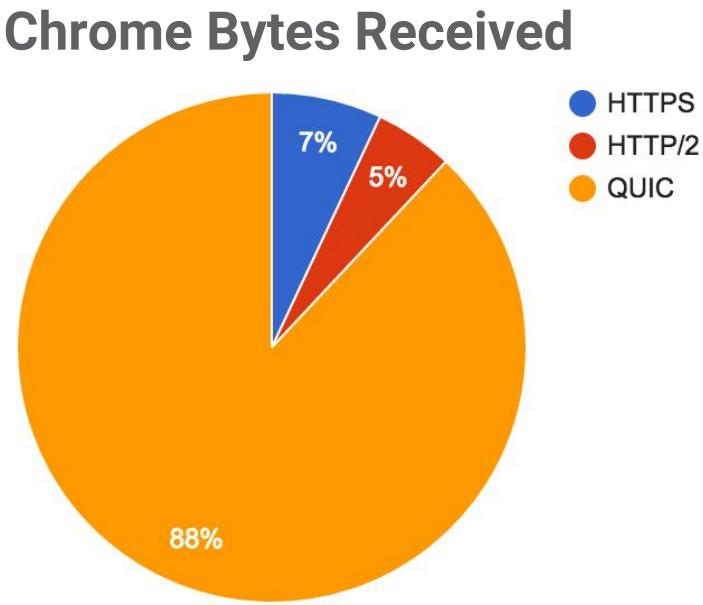
Deployment at Google

QUIC used for every major Google Site on Desktop and Android Chrome

- Disabled for domains requiring PCI compliance.

Many Google Android Apps





Fallback to HTTP/2

What if UDP is blocked?

- Chrome seamlessly falls back to HTTP/TCP

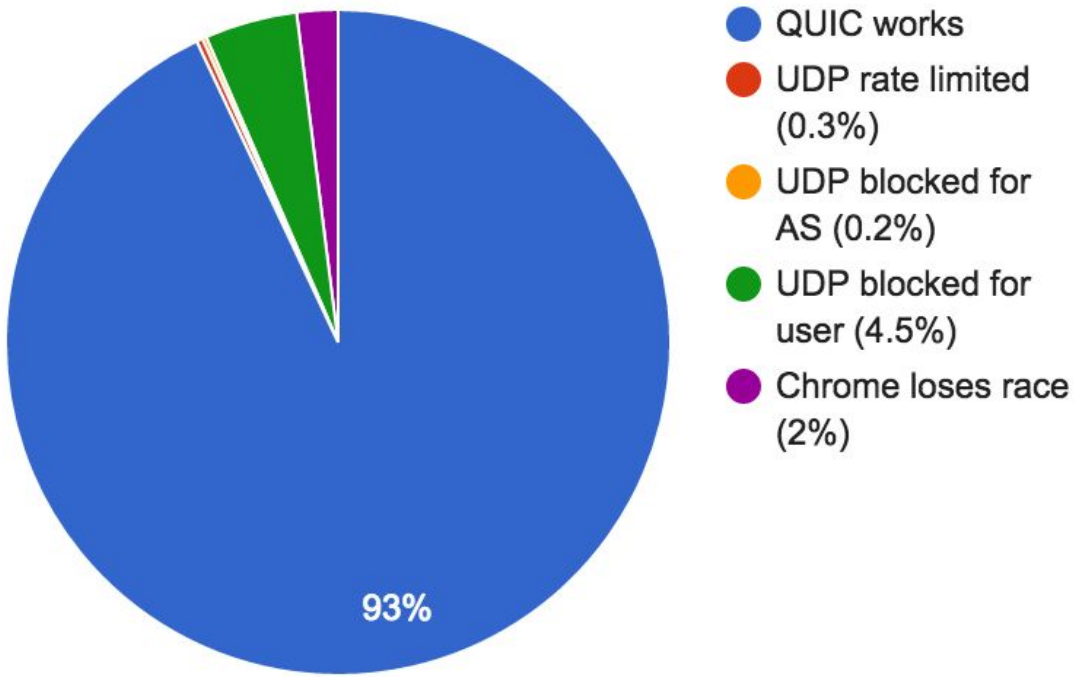
What if the path MTU is too small?

- 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

QUIC: Does it work?



Since Initial Launch, UDP rate limiting has decreased by 2/3rds

IETF Process: Four Initial Documents

draft-hamilton-quic-transport-protocol

core transport protocol description, including connection establishment, multistreaming, flow control

draft-iyengar-quic-loss-recovery

congestion control and loss recovery mechanisms for QUIC

draft-thomson-quic-tls

using TLS for QUIC's crypto handshake

draft-shade-quic-http2-mapping

mapping HTTP/2 semantics over QUIC