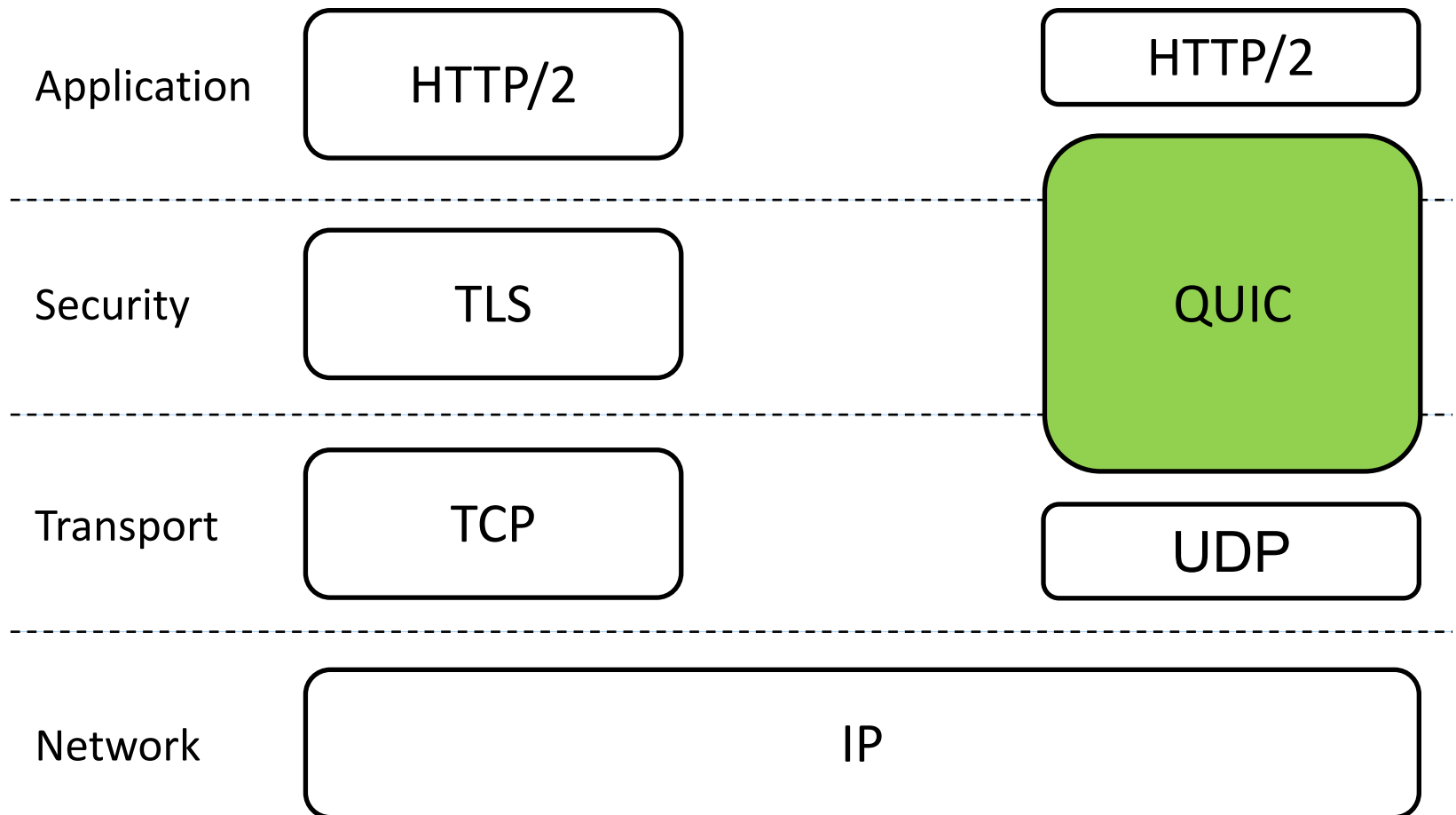




Accelerating QUIC's Connection Establishment on High-Latency Access Networks

Erik Sy

Introducing QUIC

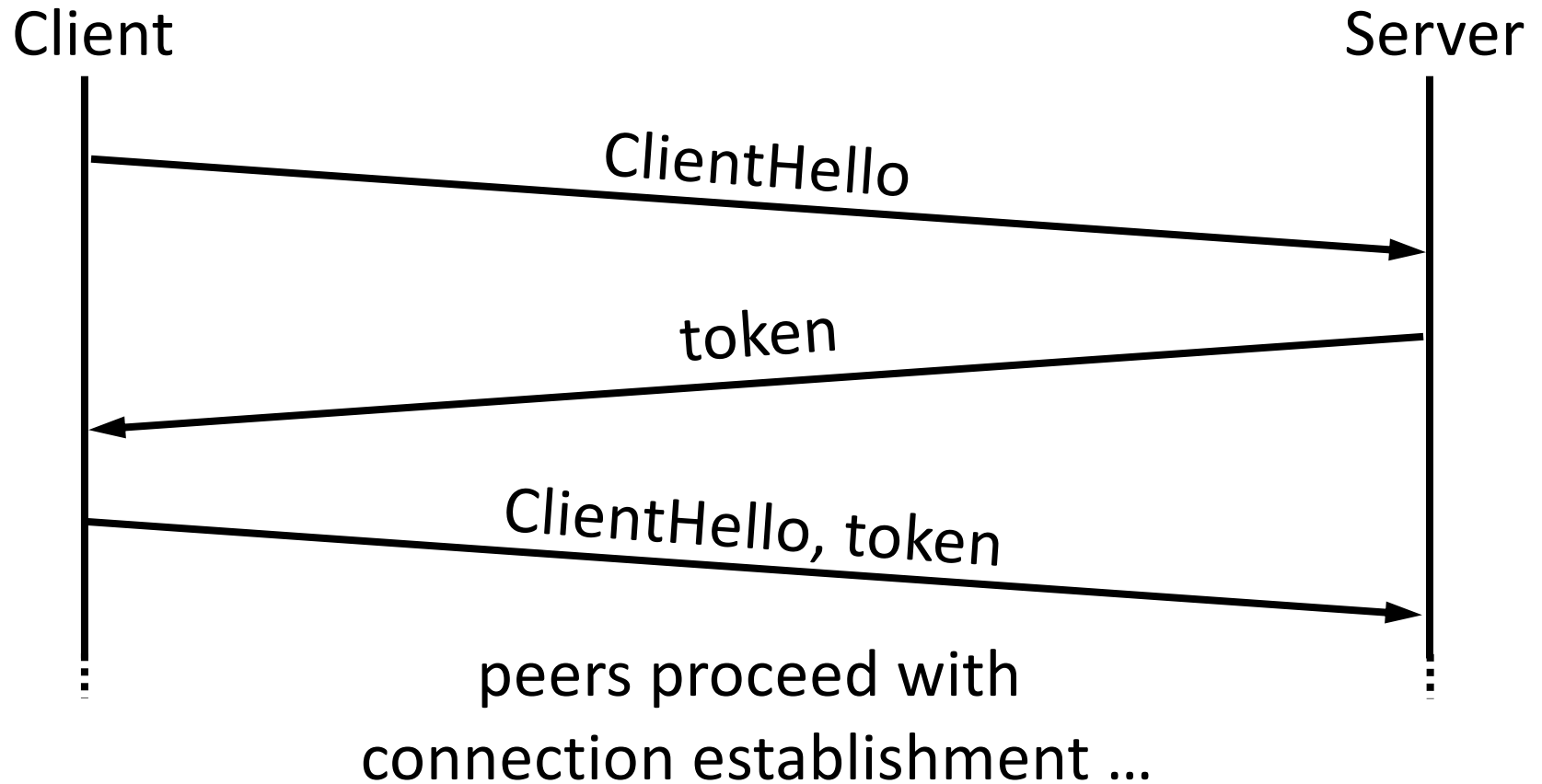


Introduction to the QUIC Transport Protocol

- QUIC is going to replace TLS over TCP in HTTP/3
- Improves problems of TLS over TCP
 - Protocol Entrenchment
 - Implementation Entrenchment
 - Handshake Delay
 - Head-of-line Blocking
 - Mobility

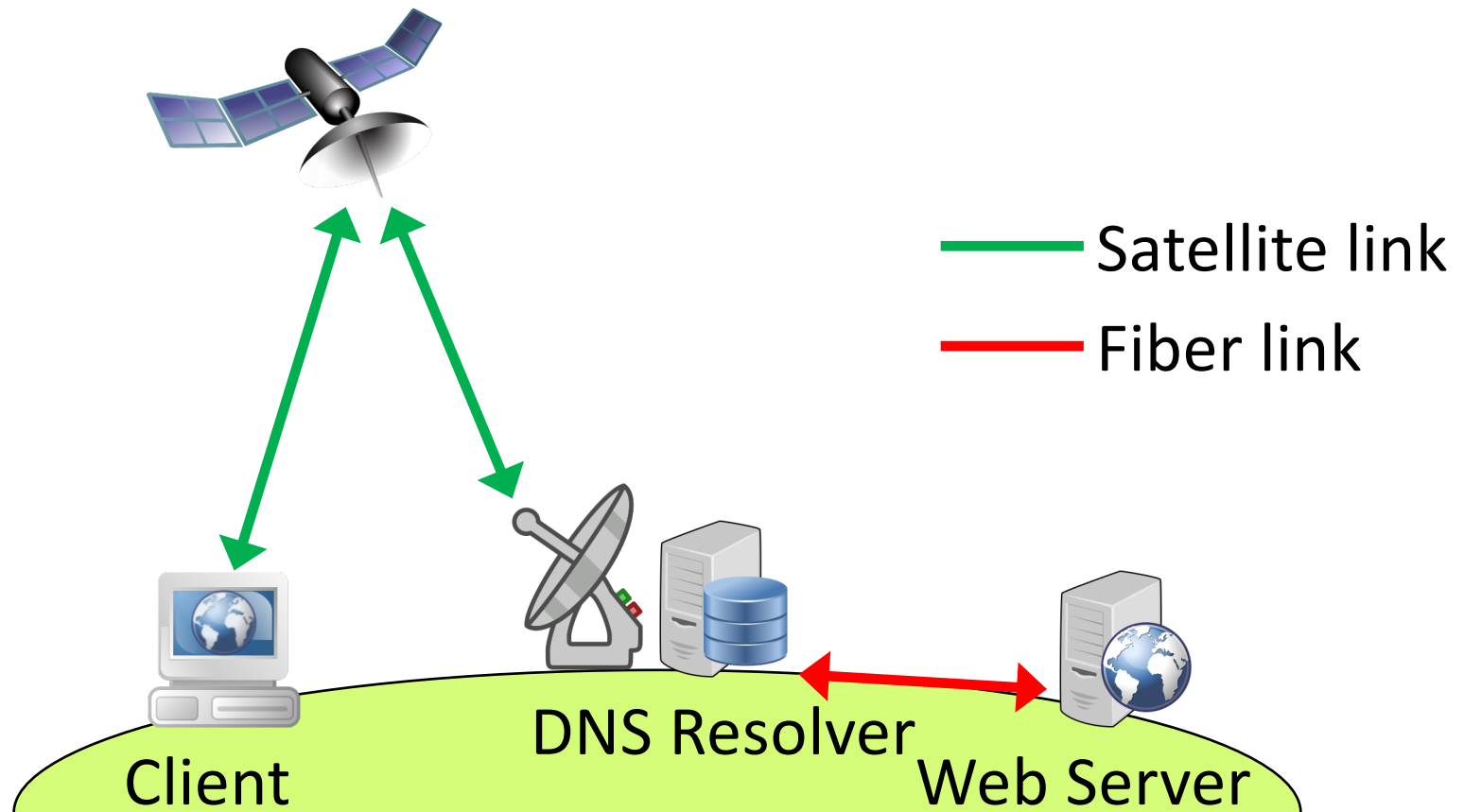
QUIC's Stateless Retry Mechanism

- Source-address tokens are used to validate the client's IP address



Problem Statement

- Clients usually experience higher network latencies to reach online services compared to their ISP-provided DNS resolver

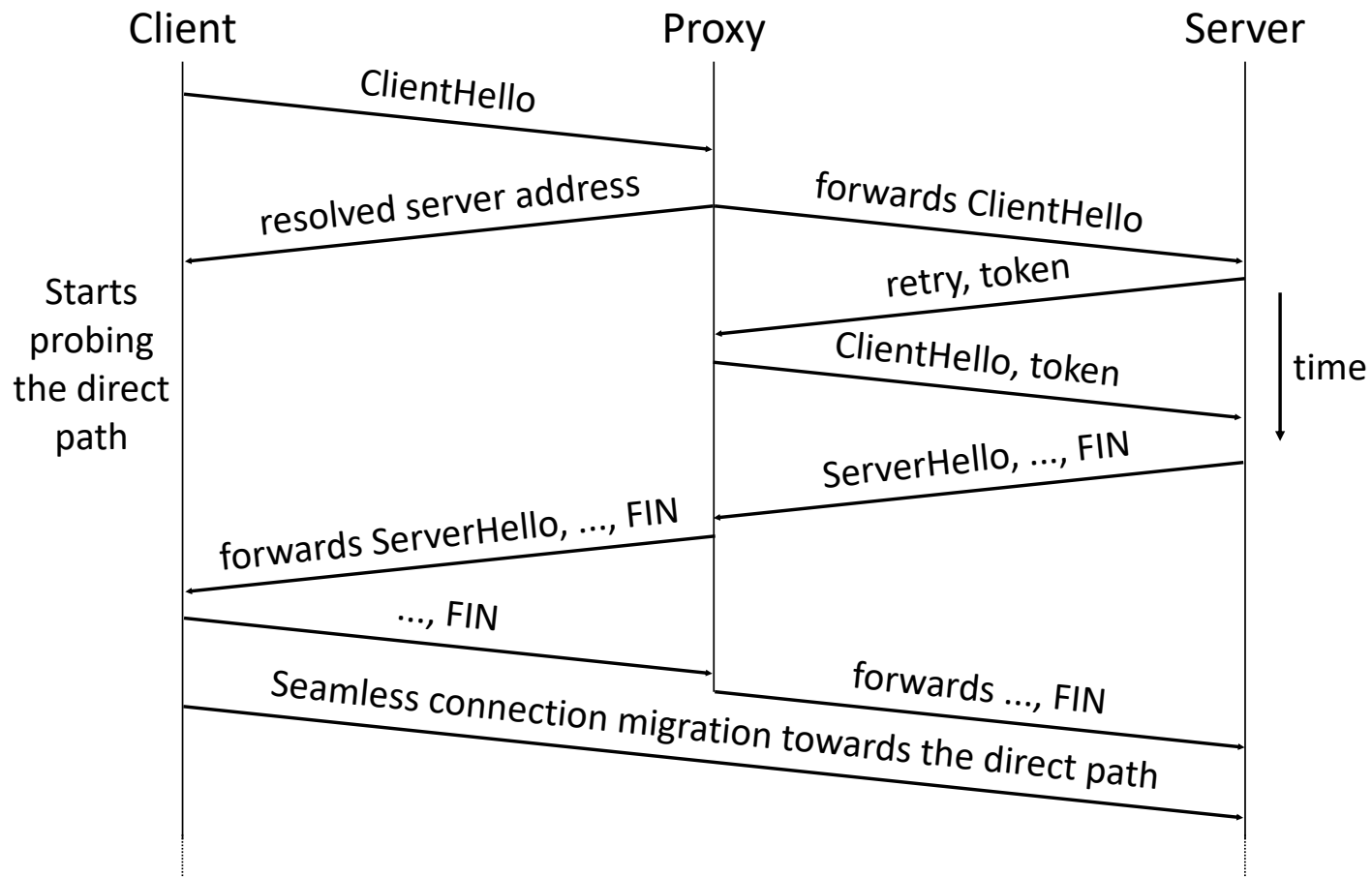


Design Goals for our Proposal

1. Deployable on today's Internet
2. Reduces the delay of QUIC handshakes requiring a prior DNS lookup
3. Supports NATed clients and does not conduct IP address spoofing
4. Guarantees end-to-end encryption between client and web server
5. Limits the consumption of the proxy's bandwidth
6. Privacy assurances matching the use of a DNS resolver

Introducing the QuicSocks Design

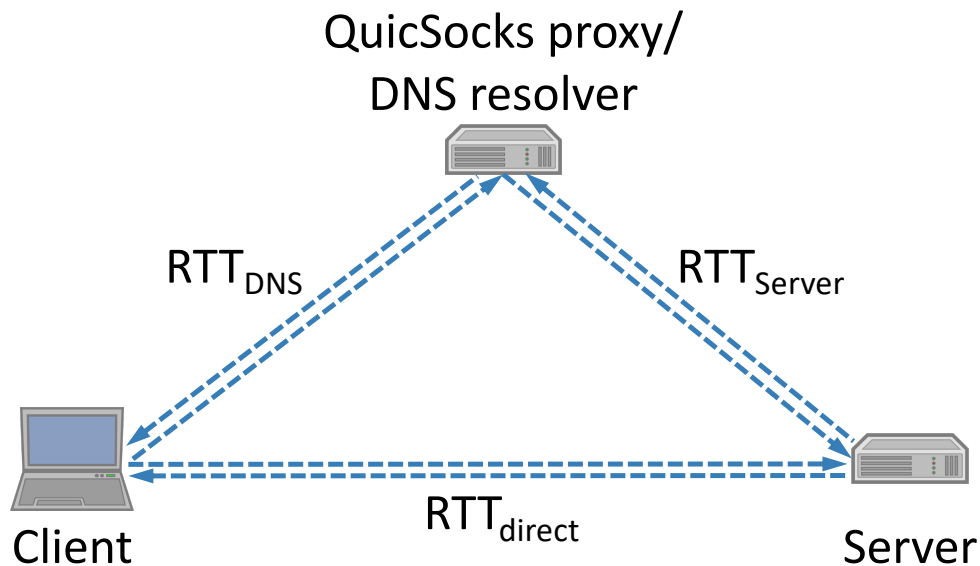
- Assumes a QuicSocks Proxy colocated with the DNS resolver



Analytical Performance Evaluation

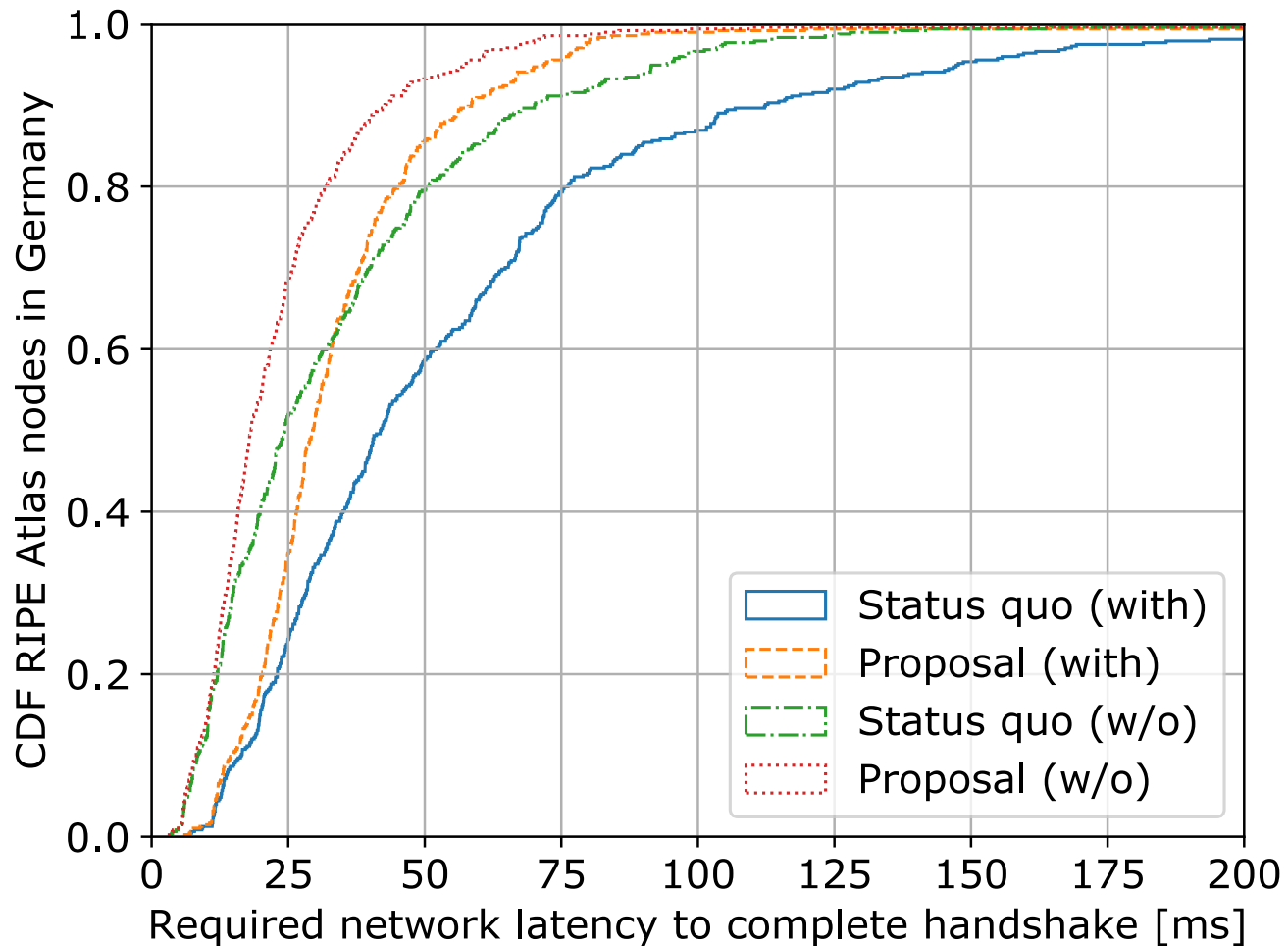
- Proposal achieves better performance if $RTT_{Server} < RTT_{direct}$

Stateless retry	Latency to establish connection (incl. DNS)	
	Status quo	Proposal
w/o	$RTT_{DNS} + RTT_{direct}$	$RTT_{DNS} + RTT_{Server}$
with	$RTT_{DNS} + 2 * RTT_{direct}$	$RTT_{DNS} + 2 * RTT_{Server}$



Empirical Performance Evaluation

- 24.3% of nodes saves at least 15ms without and 30ms with stateless retry



Conclusion

- Proposal provides great performance improvements for QUIC's connection establishment requiring prior DNS lookup
- Proposed handling of QUIC's stateless retry can be adapted by all sorts of proxies to improve the delay of QUIC's connection establishments

Thank you

Questions and Answers

E-mail: IPCCC@erik-sy.de

Slides: <https://erik-sy.de/ipccc>