SNIA NSF | NETWORKING STORAGE

QUIC- Will it Replace TCP/IP?

Live Webcast
April 2, 2020
10:00 am PT

Today's Presenters

Lars Eggert
Technical Director, Networking
NetApp

Tim Lustig
Director, Corporate Ethernet Marketing
Mellanox Technologies

SNIA-At-A-Glance

SNIA NSF | NETWORKING STORAGE

SNIA-at-a-Glance

185
industry leading organizations

2,000
active contributing members

50,000
IT end users & storage pros worldwide

Learn more: **snia.org/technical** **@SNIA**

Technologies We Cover

- Ethernet
- iSCSI
- NVMe-oF
- InfiniBand
- Fibre Channel, FCoE
- Hyperconverged (HCI)
- Storage protocols (block, file, object)
- Virtualized storage
- Software-defined storage

SNIA NSF | NETWORKING STORAGE

SNIA Legal Notice

- The material contained in this presentation copyright NetApp Inc and Lars Eggert and others as noted.
- Member companies and individual members may use this material in presentations and literature under the following conditions:
  - Any slide or slides used must be reproduced in their entirety without modification
  - The SNIA must be acknowledged as the source of any material used in the body of any document containing material from these presentations.
- This presentation is a project of the SNIA.
- Neither the author nor the presenter is an attorney and nothing in this presentation is intended to be, or should be construed as legal advice or an opinion of counsel. If you need legal advice or a legal opinion please contact your attorney.
- The information presented herein represents the author's personal opinion and current understanding of the relevant issues involved. The author, the presenter, and the SNIA do not assume any responsibility or liability for damages arising out of any reliance on or use of this information.

NO WARRANTIES, EXPRESS OR IMPLIED. USE AT YOUR OWN RISK.

Agenda

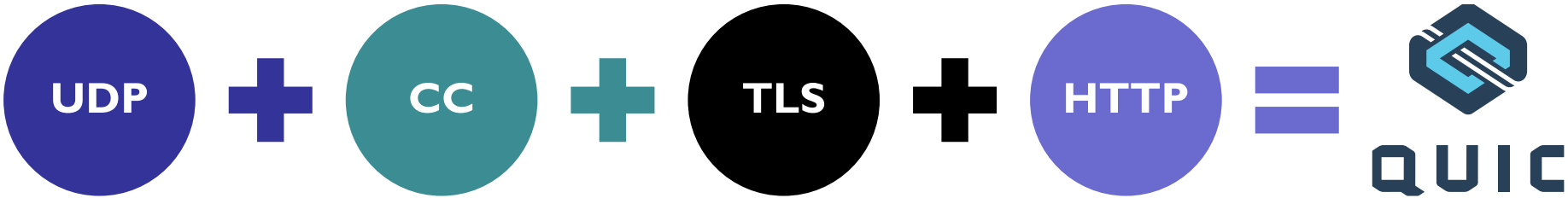
SNIA NSF | NETWORKING STORAGE

- Internet Transport
- Current Challenges
- QUIC
- Status & discussion

QUIC: a fast, secure, evolvable transport protocol for the Internet

SNIA NSF | NETWORKING STORAGE

- **Fast** — **better user experience** than TCP/TLS for HTTP/2 and other content
- **Secure** — **always-encrypted** end-to-end security, resist pervasive monitoring
- **Evolvable** — prevent network from ossifying, deploy new QUIC versions quickly
- **Transport** — **support all TCP content & more** (realtime media, etc.)
  provide better abstractions, avoid known TCP issues



tl;dr

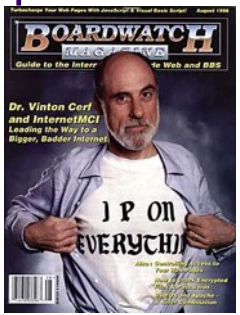
- **The web will move to QUIC first**, and then everything else will
  - This year!
- If you do anything with HTTP, TCP or just networks, **QUIC should be on your radar now**

Internet Transport

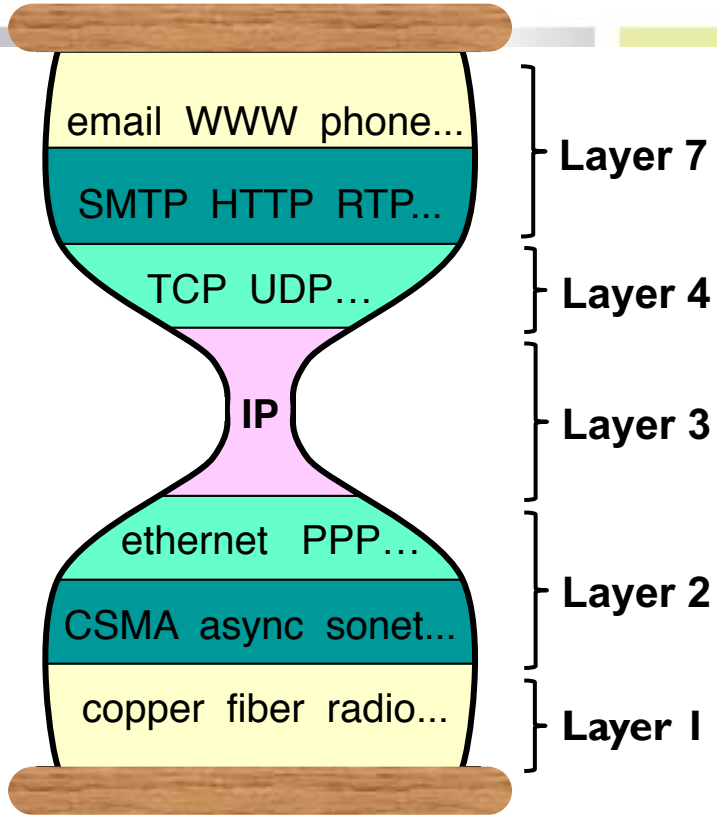
The Internet Hourglass

Classical version

- Inspired by OSI “seven-layer” model
  - Minus presentation (6) and session (5)
- “IP on everything”
  - All link tech looks the same (approx.)
- **Transport layer** provides communication abstractions to apps
  - Unicast/multicast
  - Multiplexing
  - Streams/messages
  - Reliability (full/partial)
  - Flow/congestion control
  - …



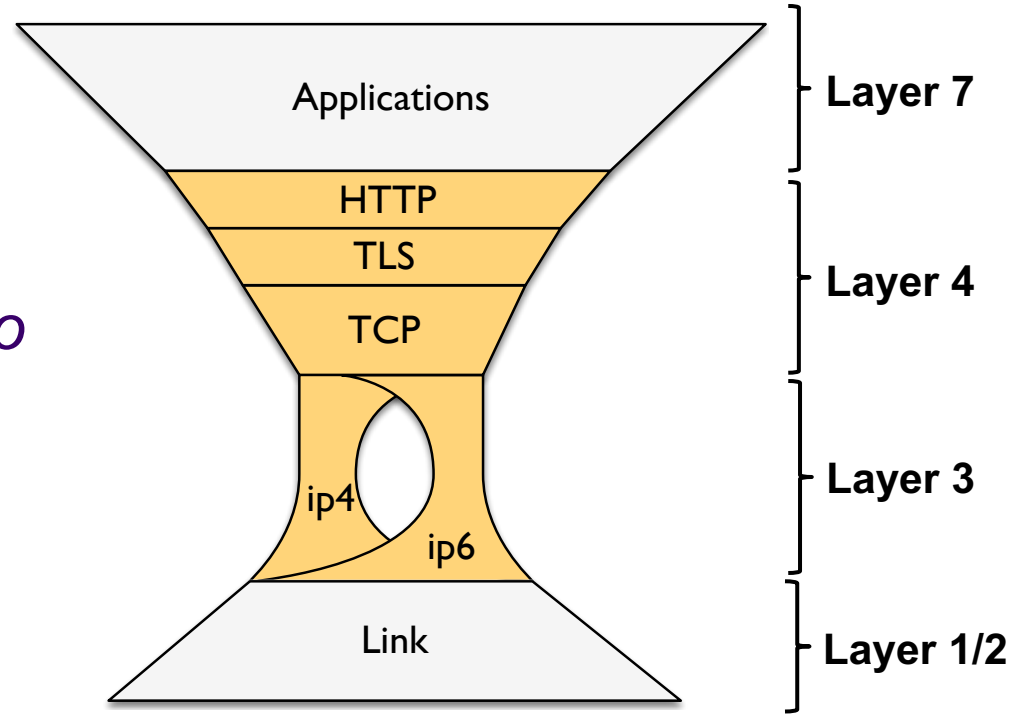
Boardwatch Magazine, Aug. 1994.



Steve Deering. Watching the Waist of the Protocol Hourglass. Keynote, IEEE ICNP 1998, Austin, TX, USA. http://www.ieee-icnp.org/1998/Keynote.ppt

The Internet Hourglass 2015 version (ca.)

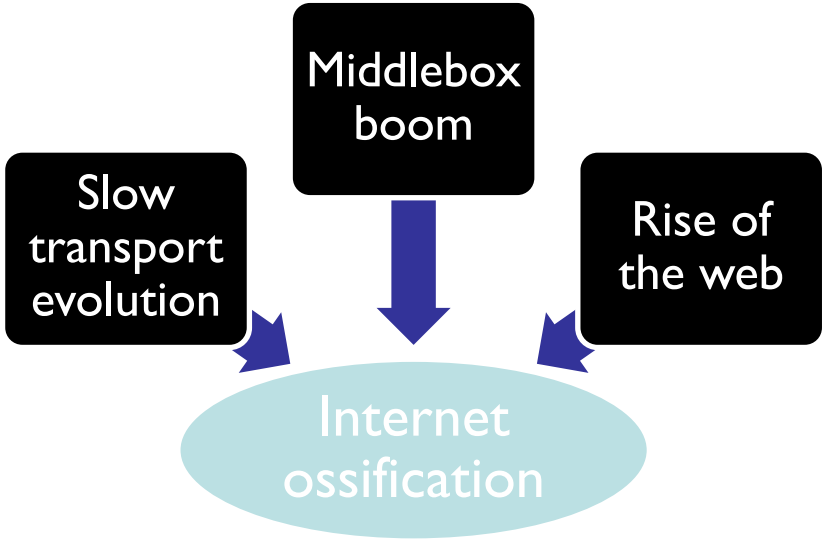
- The waist has split: **IPv4** and **IPv6**
- **TCP** is drowning out UDP
- **HTTP** and **TLS** are *de facto* part of transport
- Consequence: **web apps** on IPv4/6



B. Trammell and J. Hildebrand, "Evolving Transport in the Internet," in *IEEE Internet Computing*, vol. 18, no. 5, pp. 60-64, Sept.-Oct. 2014.

What Happened?

- **Transport slow to evolve** (esp. TCP)
  - Fundamentally difficult problem
- **Network made assumptions** about what (TCP) traffic looked like & how it behaved
- Tried to "help" and "manage"
  - TCP "accelerators" & firewalls, DPI, NAT, etc.
- **The web happened**
  - Almost all content on HTTP(S)
  - Easier/cheaper to develop for & deploy on
  - Amplified by mobile & cloud
  - Baked-in client/server assumption



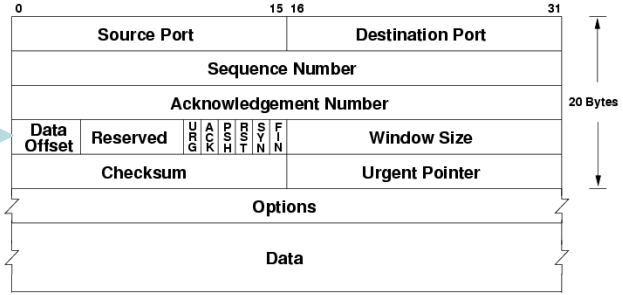
Example Ossifications

IP	•Send from/to anywhere anytime	vs. enforced directionality & timeliness
IP	•Many protocols on top of IP	vs. packets dropped unless TCP or UDP
IP	•End-to-end addressing	vs. network assumes it can rewrite addresses/ports
IP	•Use IP options to signal	vs. options not used (dropped) on WAN
*	•Bits have meaning only inside a layer	vs. network can (should!) touch bits across a packet
TCP	•Network is stateless	vs. network assumes it can track entire connection
TCP	•Data has meaning to app only	vs. network can rewrite or insert

TCP Challenges

TCP is Not Aging Well

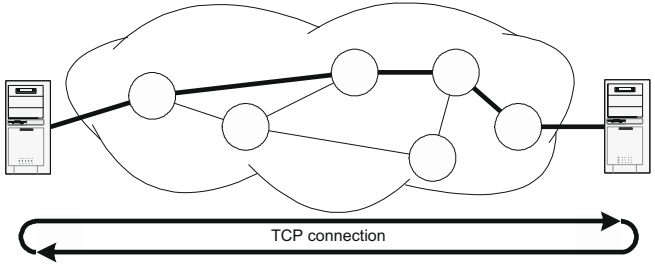
- **We're hitting hard limits** (e.g., TCP option space)
  - 40B total (15 * 4B - 20)
  - SACK-OK (2), timestamp (10), window Scale (3), MSS
  - Multipath needs 12, Fast-Open 6-18…
- **Incredibly difficult to evolve**, c.f. Multipath TCP
  - New TCP must look like old TCP, otherwise it gets dropped
  - TCP is already very complicated
- **Slow upgrade cycles** for new TCP stacks (kernel update required)
  - Better with more frequent update cycles on consumer OS
  - Still high-risk and invasive (reboot)
- **TCP headers not encrypted** or authenticated – middleboxes can still meddle
  - TCP-MD5 and TCP-AO in practice only used for (some) BGP sessions



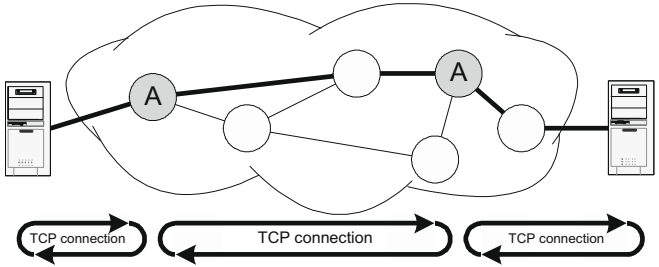
By Ere at Norwegian Wikipedia (Own work) [Public domain], via Wikimedia Commons

Middleboxes Meddle

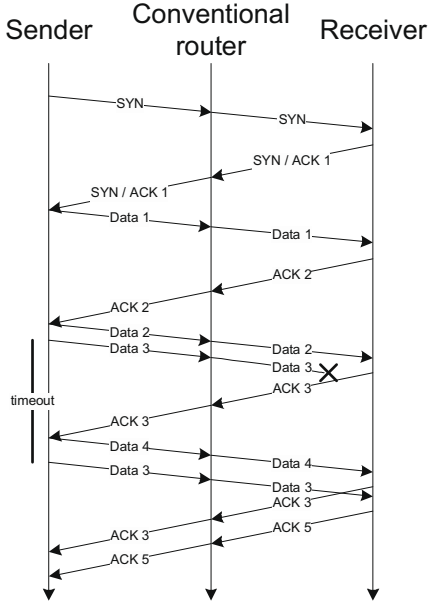
Example: TCP accelerators



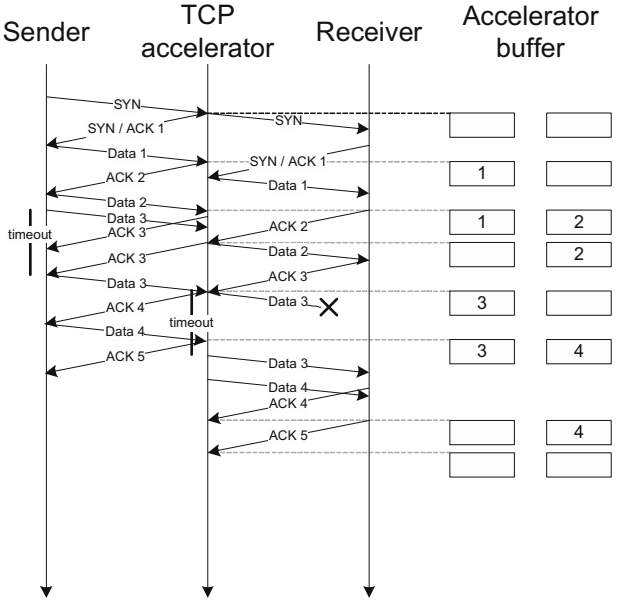
(a) Conventional TCP Connection



(b) Accelerated TCP Connection



(a) Conventional TCP Connection



(b) Accelerated TCP Connection

Sameer Ladiwala, Ramaswamy Ramaswamy, and Tilman Wolf. Transparent TCP acceleration. Computer Communications, Volume 32, Issue 4, 2009, pages 691-702.

Middleboxes Meddle

- Example: Nation states attacking end users or services

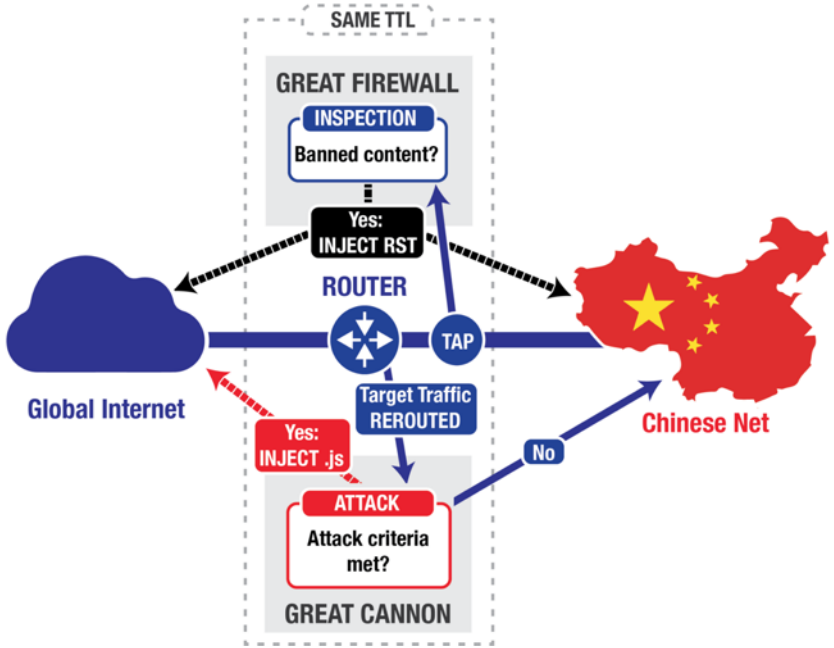
TOP SECRET//COMINT//REL TO USA, AUS, CAN, GBR, NZL

QUANTUM INSERT: racing the server

- The Game:
  - **Wait** for client to initiate new connection
  - Observe server-to-client TCP SYN/ACK
  - Shoot! (HTTP Payload)
  - **Hope** to beat server-to-client HTTP Response
- The Challenge:
  - Can only win the race on some links/targets
  - For many links/targets: too slow to win the race!

TOP SECRET//COMINT//REL TO USA, AUS, CAN, GBR, NZL

QFIRE Pilot Lead. NSA/Technology Directorate. QFIRE pilot report. 2011.



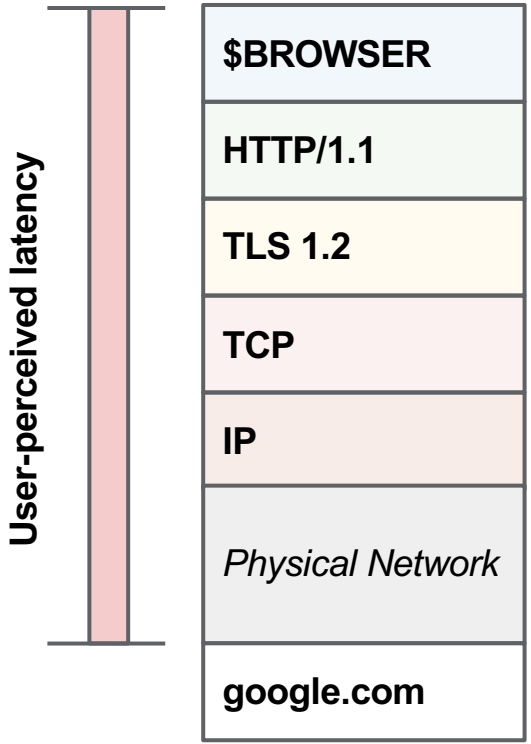
B. Marczak, N. Weaver, J. Dalek, R. Ensafi, D. Fifield, S. McKune, A. Rey, J. Scott-Railton, R. Deibert, and V. Paxson. An Analysis of China's "Great Cannon". 5th USENIX FOCI Workshop, 2015.

QUIC

Introduction

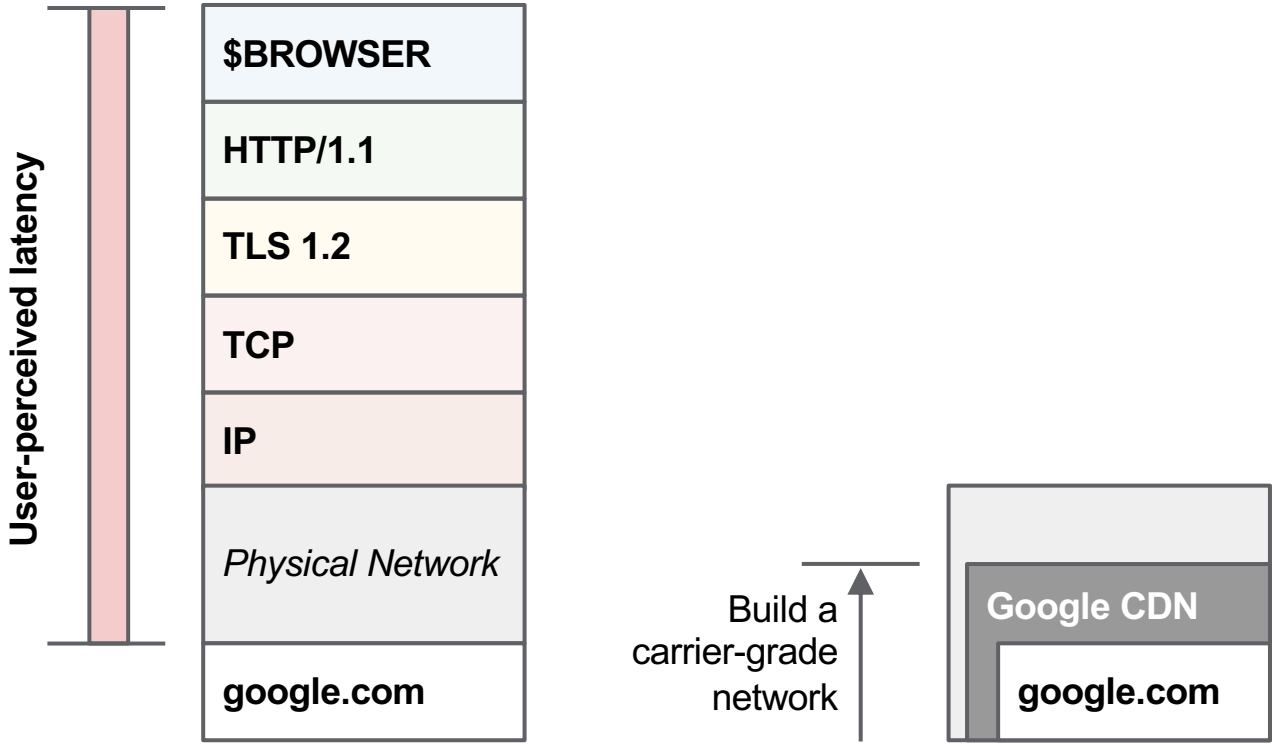
How Do You Make the Web Faster?

QUIC - Redefining Internet Transport. J. Iyengar. IETF-93 QUIC BoF presentation, 2015.



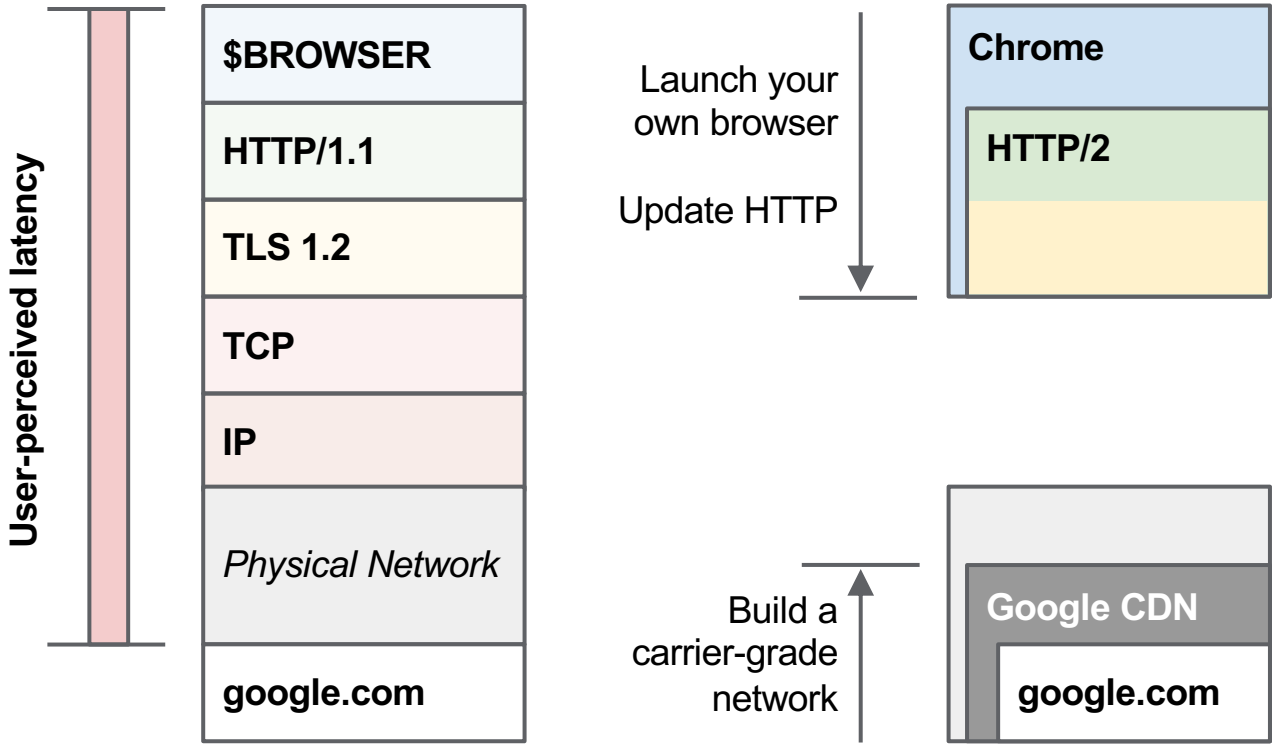
How Do You Make the Web Faster?

QUIC - Redefining Internet Transport. J. Iyengar. IETF-93 QUIC BoF presentation, 2015.



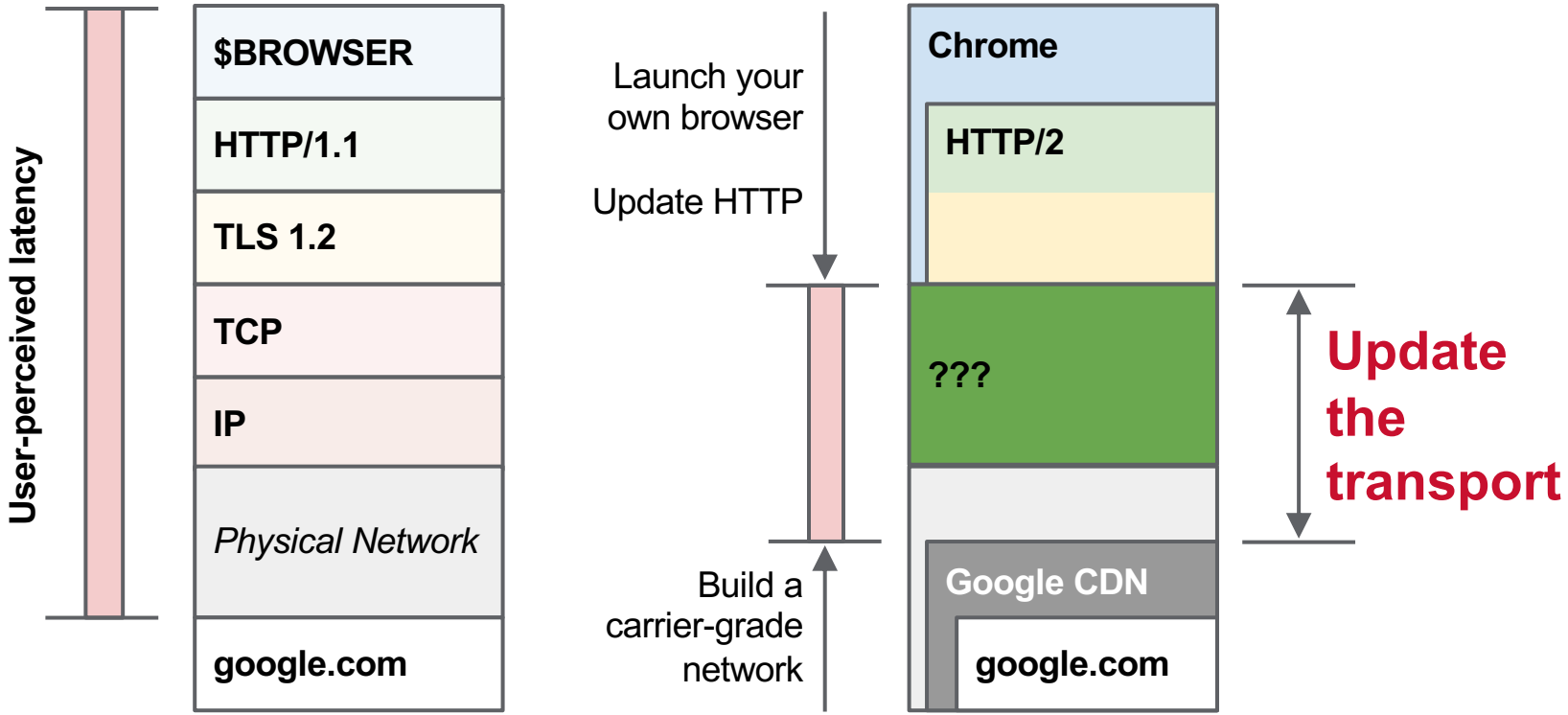
How Do You Make the Web Faster?

QUIC - Redefining Internet Transport. J. Iyengar. IETF-93 QUIC BoF presentation, 2015.



How Do You Make the Web Faster?

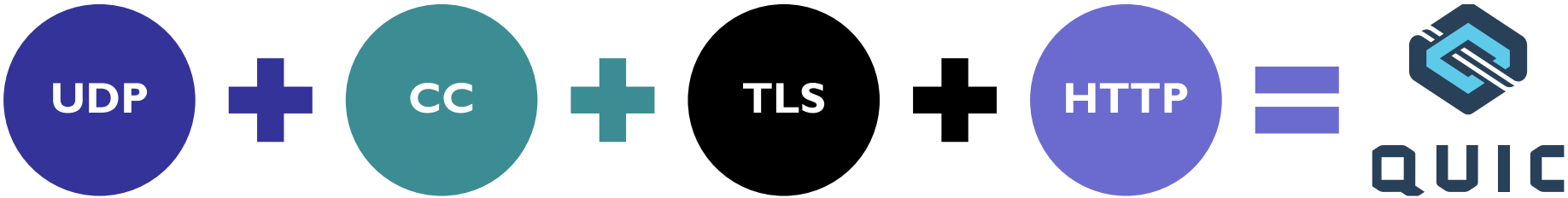
QUIC - Redefining Internet Transport. J. Iyengar. IETF-93 QUIC BoF presentation, 2015.



QUIC: a fast, secure, evolvable transport protocol for the Internet

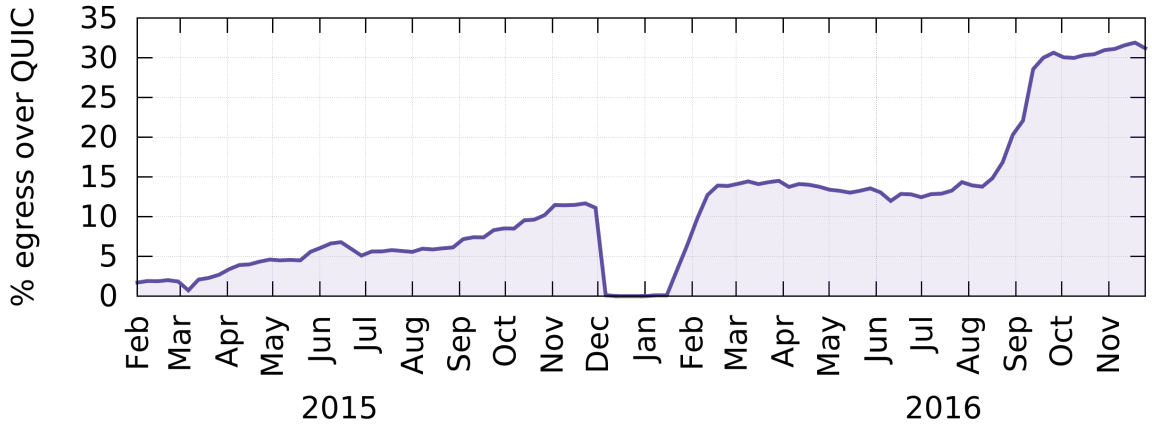
SNIA NSF | NETWORKING STORAGE

- **Fast** — **better user experience** than TCP/TLS for HTTP/2 and other content
- **Secure** — **always-encrypted** end-to-end security, resist pervasive monitoring
- **Evolvable** — prevent network from ossifying, deploy new QUIC versions quickly
- **Transport** — **support all TCP content & more** (realtime media, etc.)
  provide better abstractions, avoid known TCP issues



QUIC is Not *That New*, Actually

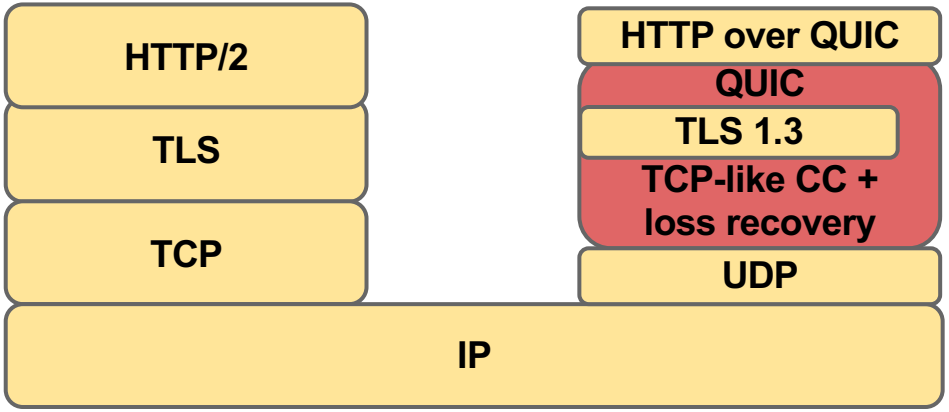
- Originates with Google, deployed between Google services and Chrome since 2014
- As of mid-2017, makes up 35% of Google egress traffic (~7% of total Internet traffic)



A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W. Chang, and Z. Shi. 2017. *The QUIC Transport Protocol: Design and Internet-Scale Deployment.*. ACM SIGCOMM, 2017.

QUIC in the Stack

- Integrated transport stack on top of UDP
- Replaces TCP and some part of HTTP; reuses TLS-1.3
- Initial target application: HTTP/2
- Prediction: many others will follow



J. Iyengar. QUIC Tutorial A New Internet Transport/ IETF-98 Tutorial, 2017.

Why UDP?

- TCP hard to evolve
- Other protocols blocked by middleboxes (SCTP, etc.)
- **UDP is all we have left**
- Not without problems!
  - Many middleboxes ossified on “UDP is for DNS”
  - Enforce short binding timeouts, etc.
  - Short-term issue with hardware NIC offloading
- Also, benefits
  - Can deploy in userspace (no kernel update needed)
  - Can offer alternative transport types (partial reliability, etc.)

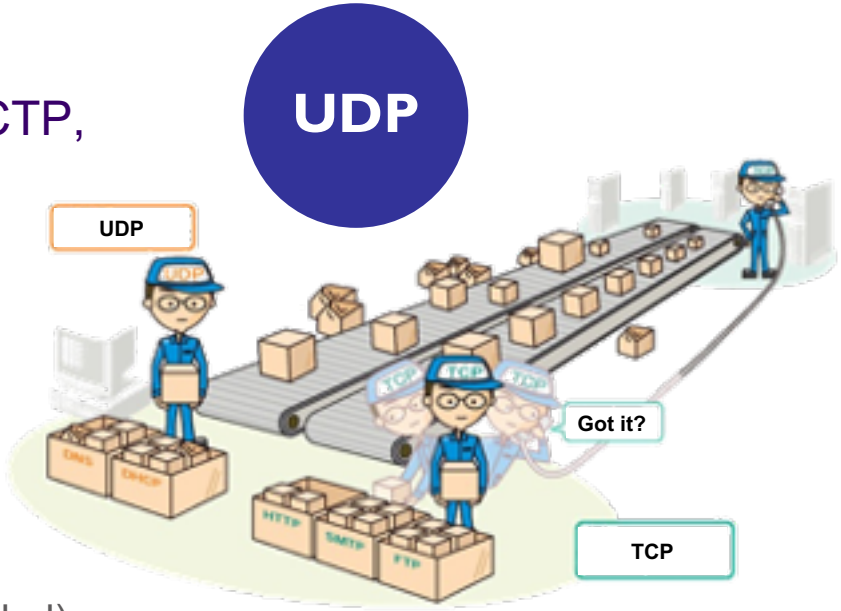


Image from http://itpro.nikkeibp.co.jp

Why Congestion Control?

SNIA NSF | NETWORKING STORAGE

- Functional CC is **absolute requirement** for operation over real networks
  - UDP has no CC
- First approach: **take what works for TCP, apply to QUIC**
- Consequence: need
  - Segment/packet numbers
  - Acknowledgments (ACKs)
  - Round-trip time (RTT) estimators
  - etc.
- Not an area of large innovation at present
  - This will change

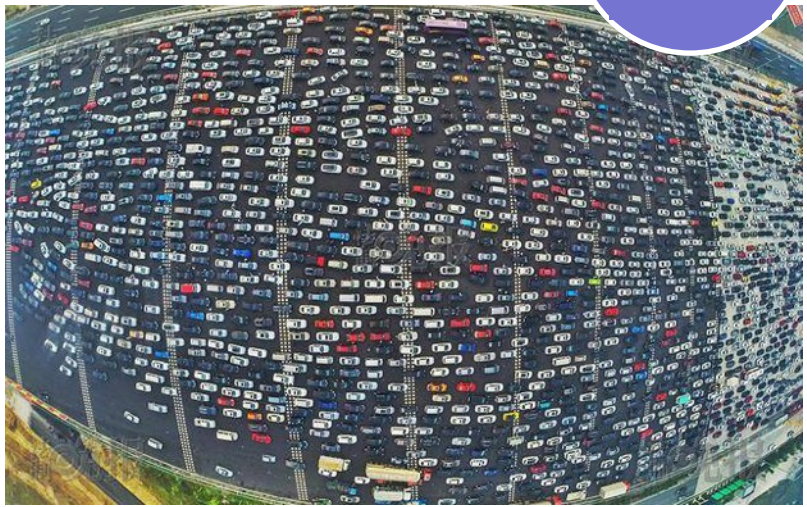
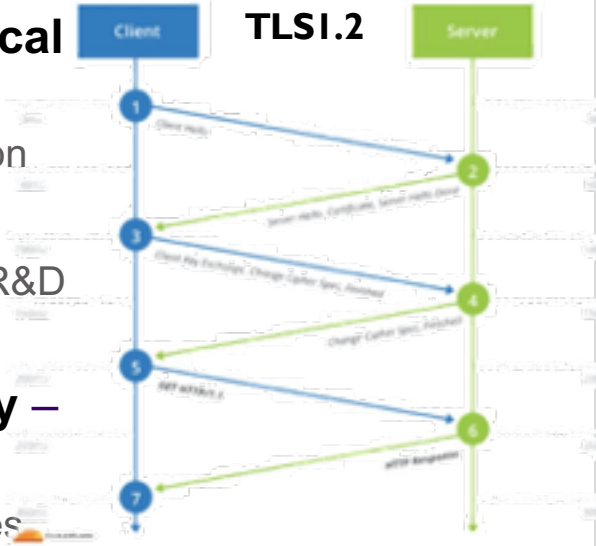


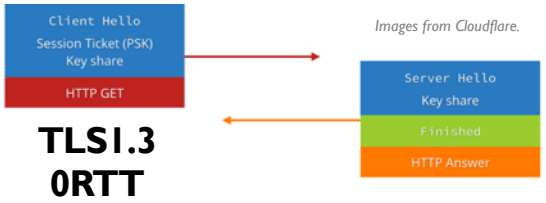
Image from People's Daily, http://people.cn/

Why Transport-layer Security (TLS)?

TLS

- **End-to-end security is critical**
  - To protect users
  - To prevent network ossification
- TLS is very widely used
  - Can leverage all community R&D
  - Can leverage the PKI
- **Don't want custom security** – too much to get wrong
  - Even TLS keeps having issues
  - But TLS 1.3 removes a lot of cruft
- And benefit from new TLS features
  - E.g., 0-RTT handshakes (inspired by gQUIC-crypto)



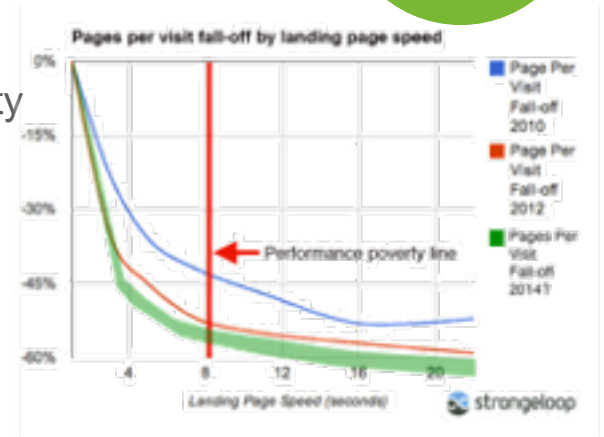


Images from Cloudflare.

Why HTTP?

HTTP

- Because that's where the **impact** is
  - Web industry incredibly interested in improved UE and security
- Rapid update cycles for browsers, servers, CDNs, etc.
  - Can deploy and update QUIC quickly
- Many other app protocols will follow

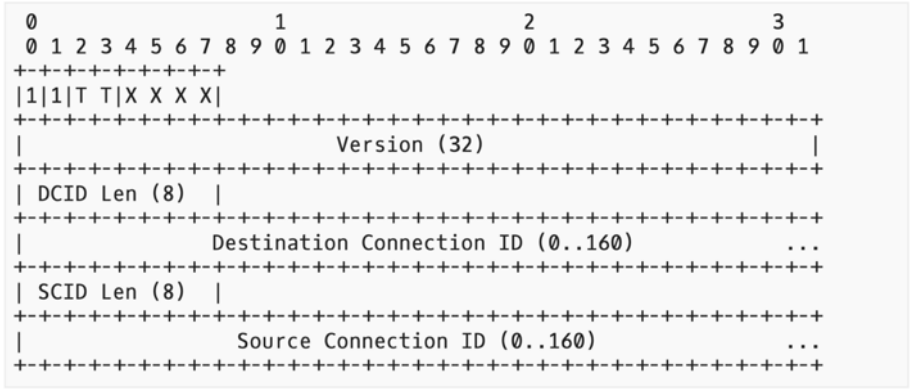


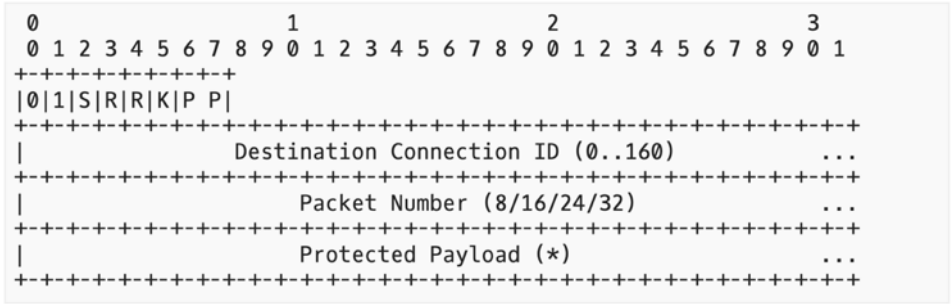
QUIC

Selected Aspects

Minimal Network-Visible Header

- With QUIC, the network sees:
  - Packet **type** (partially obfuscated)
  - QUIC **version** (only in long packet header)
  - Destination **CID**
  - Packet **number** (obfuscated)
- With TCP, also
  - ACK numbers, ECN information
  - Timestamps
  - Windows & scale factors
- Also, entire QUIC header is **authenticated**, i.e., not modifiable

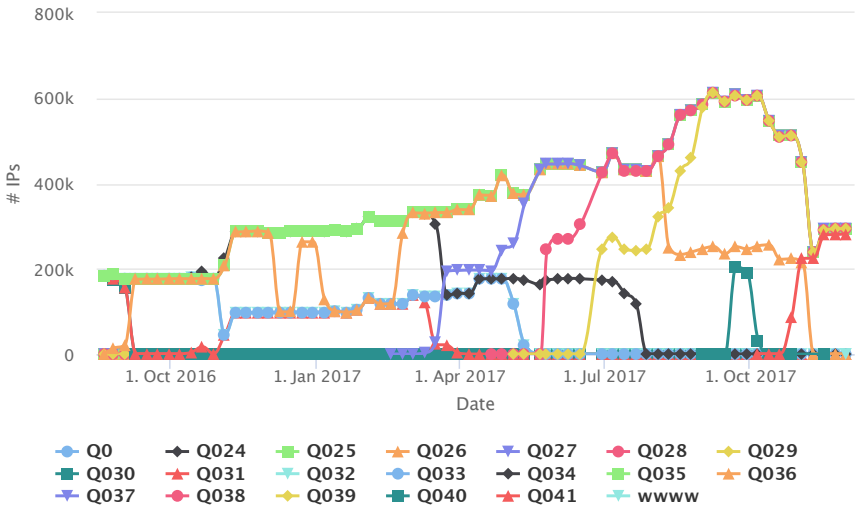




Version Negotiation

(Currently under re-design)

- 32-bit version field
  - IP: 8 bits, TCP: 0 bits
- Allows rapid deployment of new versions
  - Plus, vendor-proprietary versions
- Very few **protocol invariants**
  - Location and lengths of version and CIDs in LH
  - Location and lengths of CID in SH (if present)
  - Version negotiation server response
  - Etc. (details under discussion)
- Everything else is version-dependent
  - But must **grease** unused codepoints!



Source: RWTH QUIC Measurements: https://quic.comsys.rwth-aachen.de/

1-RTT vs. 0-RTT Handshakes

- **QUIC client can send 0-RTT data in first packets**
  - Using new TLS 1.3 feature
- Except for very first contact between client and server
  - Requires 1-RTT handshake (same latency as TCP w/o TLS)
- **Huge latency win in many cases** (faster than TCP)
  - HTTPS: 7 messages
  - QUIC 1-RTT or TCP: 5 messages
  - QUIC 0-RTT: 2 messages
- Also helps with
  - Tolerating NAT re-bindings
  - Connection migration to different physical interface
- But only for **idempotent** data

Everything Else is Frames

- Inside the crypto payload, **QUIC carries a sequence of frames**
  - Encrypted = can change between versions
- Frames can come in **any order**
- Frames carry **control data** and **payload data**
- Payload data is carried in **STREAM** frames
  - Most other frames carry control data
- Packet acknowledgment blocks in **ACK** frames

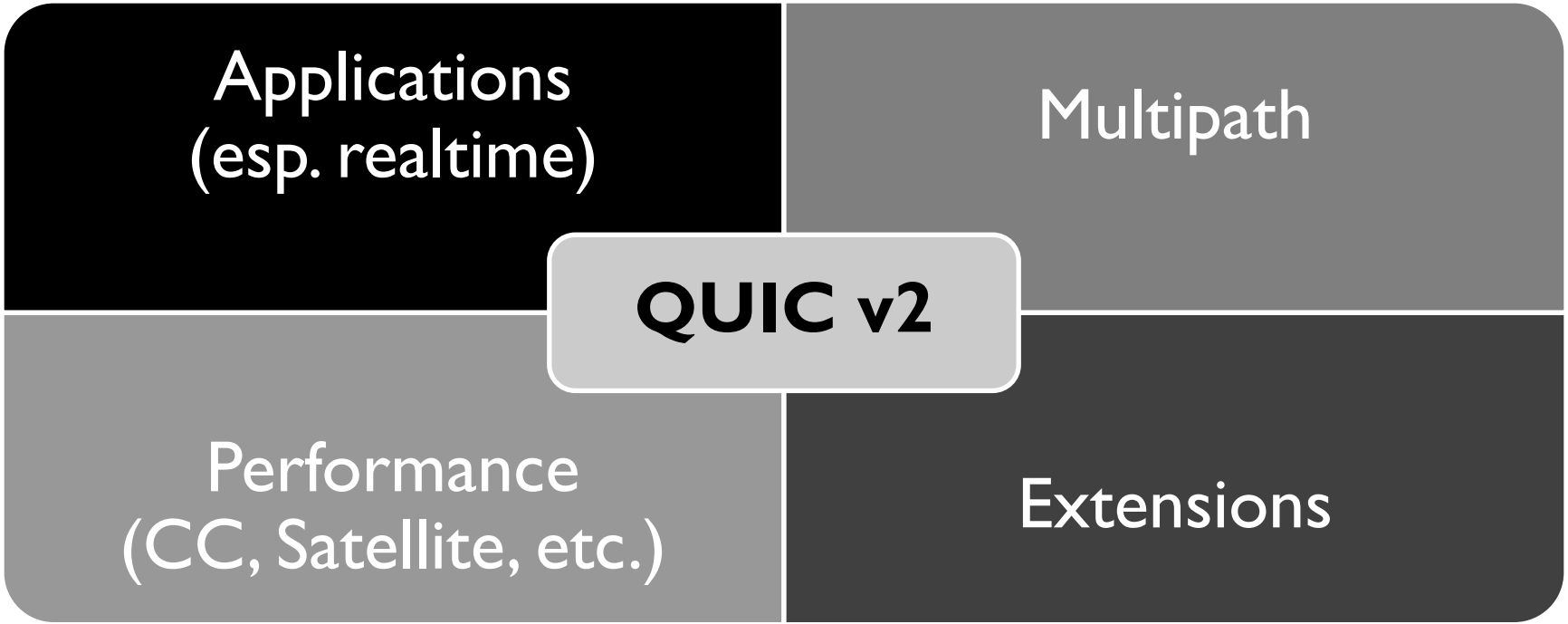
- PADDING
- PING
- **ACK**
- RESET_STREAM
- STOP_SENDING
- CRYPTO
- NEW_TOKEN
- **STREAM**
- MAX_DATA
- MAX_STREAM_DATA
- MAX_STREAMS
- DATA_BLOCKED
- STREAM_DATA_BLOCKED
- STREAMS_BLOCKED
- NEW_CONNECTION_ID
- RETIRE_CONNECTION_ID
- PATH_CHALLENGE
- PATH_RESPONSE
- CONNECTION_CLOSE
- HANDSHAKE_DONE

Stream Multiplexing

- A QUIC **connection** multiplexes potentially many **streams**
  - Congestion control happens at the connection level
  - Connections are also flow controlled
- **Streams**
  - Carry units of application data
  - Can be uni- or bidirectional
  - Can be opened by client or server
  - Are flow controlled
  - Currently, always reliably transmitted (partial reliability coming soon)
- Number of open streams is negotiated over time (as are stream windows)
- Stream prioritization is up to application

Current Status & Discussions

Beyond QUIC v1



Encryption vs. Network Management

SNIA NSF | NETWORKING STORAGE

- Claims that network management systems rely on TCP header inspection
  - To obtain loss, RTT, etc. information
- Concern that encrypting this information will be troublesome for operators
- Proposals for limited information exposure
  - e.g., the “spin bit”, the “loss bits”
- Uncertainties
  - Can networks trust this information?
  - Incentives for opting in? Penalties??

Encryption vs. Allowing Passive Measurements

- **Independent passive measurability** of the Internet one key factor to its success
  - Many protocols deficiencies were identified and fixed based on independent measurements
- **Are we giving up something** fundamental here?
- Or are we at a point where active measurements have taken over anyway?

QUIC and the IETF

- **QUIC is being standardized in the IETF**
  - QUIC is already very different from Google QUIC
- Est. delivery date: Sep 2020
- 20+ known implementation efforts:

CLOUDFLARE · TUM · facebook · Microsoft · NGINX · Google · NetApp · mozilla · traffic server™ · LITESPEED · fastly · f5 · Apple

- **https://quicwg.github.io/**
- **https://quicdev.slack.com**

Interop Status

SNIA NSF | NETWORKING STORAGE

server → / client ↓	h2o/quicly	quant	ngtcp2	mvfst	picoQUIC	msquic	f5	quiche	lsquic	ngx_quic	Quinn	AkamaiQUIC	aioquic	~gQUIC	Kwik&Flupke	Haskell QUIC
h2o/quicly																
quant	VHDCRZSQ U 3	VHDCRZSQ MBAUPELT	VHDCRZSQ MBU 3	VHDCRZQ MBT 3	VHDCRZSQ MBUPT 3	VHDCRZQ MBUPT 3	VHDCRZSQ UET 3	VHDCRZSQ 3	VHDCRZSQ MBUPET 3	VHDCRZQ 3	VHDCRZSQ MBUPE		VHDCRZSQ MBUPT 3	VHDCRQ 3		VHDCRZSQ MB 3
ngtcp2	VHDR U 3	V	VHDCRZS MBAU 3dp	VHDCRZ MBA 3d	VHDCRZS MBAU 3	VHDCR MBAU 3	VHDCRZS U 3	VHDCRZS 3	VHDCRZS MBAU 3d	VHDCRZ 3	V		VHDCRZ MBAU 3dp	VHDCR 3d		VHDCRZS MBA 3d
mvfst		VHDCRSZQ MB	VHDCRSZQ 3	VHDCRZQ MBLT 3d	VHDCRSZQ MB 3	VHDCSQ MBT 3	VHDCSRZ T 3d		VHDCRZSQ MBT 3d							
picoQUIC	VHDCRZSQ UT 3	VHDCRZSQ MBAUP	VHDCRZSQ MBAUT 3	VHDCRZQ MBLT 3	VHDCRZSQ MBAUPLT 3	VHDCRQ MBUPT 3	VHDCRZSQ UT 3	VHDCRZSQ 3	VHDCRZSQ MBUPT 3	VHDCRQ 3	VHDCRZSQ MBUP		VHDCRZSQ MBUP 3	VHDCRQ B 3		VHDCRZSQ MBL 3
msquic	VHCRQ U	VHDCRZSQ UBP	VHCRSQ MBU	VHDCRZQ MB	VHDCRZSQ MBUP	VHDCRSQ MBUPLT	VHCRSQ U	VHDCRZQ	VHCRSQ MBU	VHCRQ		-	VHDCRZSQ MBUP	VHCQ	-	VHDCRZSQ M
f5		VHDS	VHDS T 3		VHDCS 3	VH	VHDCS T 3d	VHDCS 3		VHDC 3			VHDCS T 3			VCS 3
f5_test		V	VHDCRZSQ 3d	V	VHDCS	VHDC	VHDCS		VHDCRZS		VS					VHDCS
lsquic	VHDCRSQ 3	VSQ	VHDCRSQ 3d	VHDCRQ 3dp	VHDCRSQ P 3	VHDCRQ PT 3d	VHDCRSQ ET 3d	VHDCRSQ 3	VHDCRSQ MPET 3dp	VHDCRQ 3			VHDCRSQ PT 3dp	VHDCRQ 3d		VHDCRSQ 3d
ngx_quic																
Quinn	VHDCRZS U	VHDCRZS U		VHDCRZS B	VHDCRZS BU 3	VHDCRZS BU 3	VHDCRSZ U	VHDCRZS B 3	VHDCRZS BU 3		VHDCRZSQ BU 3					VHDCRZS
AkamaiQUIC																
aioquic	VHDCRZSQ U		VHDCRZSQ MBU 3dp		VHDCRZSQ MBUPLT 3	VHDCRZQ MBUPL			VHDCRZSQ MBUT 3dp		VHDCRZSQ BUP		VHDCRZSQ MBUPLT 3dp			
~gQUIC	VHDCR 3	V	VHDCR B 3d	-	VHDCRS 3	V	VHDCRS 3d	VHDCRS 3	VHDCRS 3	VHDCR 3	V	-	VHDCRS B 3d	VHDCR B 3d	-	VHDCRS 3
Kwik&Flupke	HDCRS	VHDCRS MB		HDCR 3	VHDCRS B		VHDCRS M 3	VHDCRS 3	VHDCRS MB 3	VHD 3	VHDCRS 3		VHDCRS B			
Haskell QUIC	VHDRZSQ B	VHDRZSQ MBA	VHDRZSQ MBA	VHDRZQ MBA	VHDRZSQ MBA	VHDRZSQ MBA	VHDRZSQ	VHDRZSQ	VHDRSQ MBA	VHDRZQ	VHDRZSQ MBA		VHDRZSQ MBA	VHDRQ B		VHDRZSQ MBA

https://docs.google.com/spreadsheets/d/1D0tW89vOoaScs3IY9RGC0UesWGAwE6xyLk0I4JtvTVg/edit#gid=438405370

How to Participate?

- QUIC WG is open to all
  - Use the mailing list
  - Discuss issues/PRs on GitHub
  - Participate in meetings
- https://quicwg.org/ will get you started
- You can talk to us first, too
- “Note Well” – disclose IPR

- IETF is open to all
- 3x meetings/year, next:
  - Vancouver, March
  - Madrid, July
  - Bangkok, November
- Grants for academics:
  - ACM/IRTF ANRW workshop (travel grants, only students)
  - IRTF Chair discretionary fund (need strong reason)

GitHub

- https://quicwg.org/ links to a list of implementations
- Many are open source and live on GitHub
- Contact maintainers and start issues/PRs

After this Webcast

- Please rate this webcast and provide us with feedback
- This webcast and a PDF of the slides will be posted to the SNIA Networking Storage Forum (NSF) website and available on-demand at www.snia.org/forums/nsf/knowledge/webcasts
- A full Q&A from this webcast, including answers to questions we couldn't get to today, will be posted to the SNIA-NSF blog: sniansfblog.org
- Follow us on Twitter @SNIANSF

Thank you

Questions later?

Email lars@netapp.com