QUIC

Geoff Huston AM

Chief Scientist, APNIC

TCP is..

The workhorse of the Internet!

A transport protocol that constructs a reliable full duplex adaptive streaming service on top of an unreliable IP datagram service

- Uses a coordinated state between the two end systems without any network intervention or mediation
- Uses a sliding window to allow lost data to be resent
- Uses ACK-clocking to regulate the sending behaviour to match network path capacity estimate

TCP is NOT…

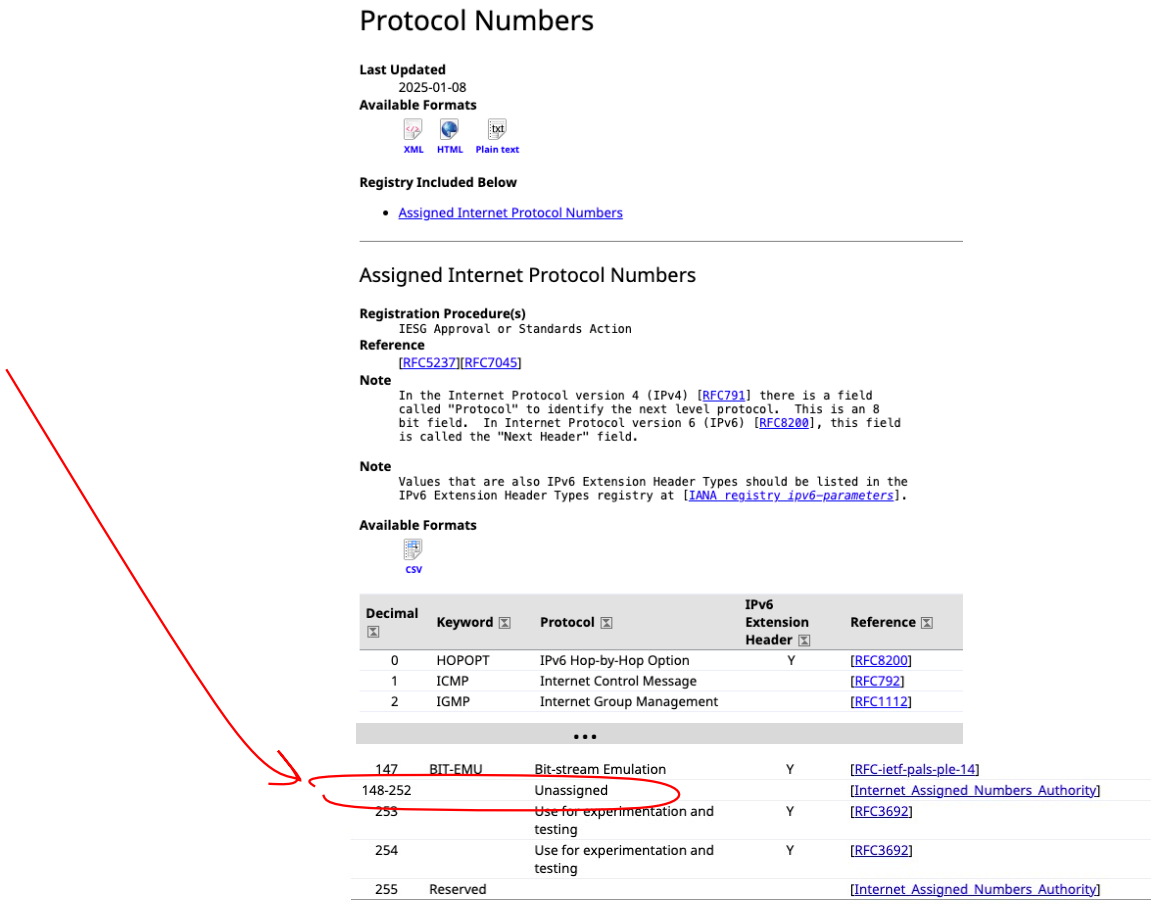
- Fully independent of the underlying platform Operating System's transport services
- Fully multi-stream (it has head-of-line blocking)
- Fully multi-path (yes, MP-TCP exists, but there are some outstanding issues here!)
- Address-agile
- Free from on-the-wire network intervention (TCP control parameters are sent in the clear)
- Has e2e encryption as a second step / afterthought (TLS)
- Everything for everyone – it relies on the application to perform data framing and in-band control

Can we fix this in TCP?

- We've been trying to fix this for about 40 years
  - Without much success!
- TCP does have a capabilities exchange in the opening handshake, but it has only been used for Max Segment Size, Window Scaling and Selective Acknowledgement
- A large part of the issue is the proliferation of TCP-aware middleware embedded within networks, which prevents significant modification of TCP behaviours
- So how about an entirely new protocol?

A New Protocol?

- We could define a new IP protocol
- We still have 105 available protocol number slots in the IANA Protocol Number registry!



Protocol Numbers

Last Updated
2025-01-08

Available Formats
XML HTML Plain text

Registry Included Below

- Assigned Internet Protocol Numbers

Assigned Internet Protocol Numbers

Registration Procedure(s)
IESG Approval or Standards Action

Reference
[RFC5237][RFC7045]

Note
In the Internet Protocol version 4 (IPv4) [RFC791] there is a field called "Protocol" to identify the next level protocol. This is an 8 bit field. In Internet Protocol version 6 (IPv6) [RFC8200], this field is called the "Next Header" field.

Note
Values that are also IPv6 Extension Header Types should be listed in the IPv6 Extension Header Types registry at [IANA registry *ipv6-parameters*].

Available Formats
CSV

Decimal	Keyword	Protocol	IPv6 Extension Header	Reference
0	HOPOPT	IPv6 Hop-by-Hop Option	Y	[RFC8200]
1	ICMP	Internet Control Message		[RFC792]
2	IGMP	Internet Group Management		[RFC1112]
...				
147	BIT-EMU	Bit-stream Emulation	Y	[RFC-ietf-pals-ple-14]
148-252		Unassigned		[Internet Assigned Numbers Authority]
253		Use for experimentation and testing	Y	[RFC3692]
254		Use for experimentation and testing	Y	[RFC3692]
255	Reserved			[Internet Assigned Numbers Authority]

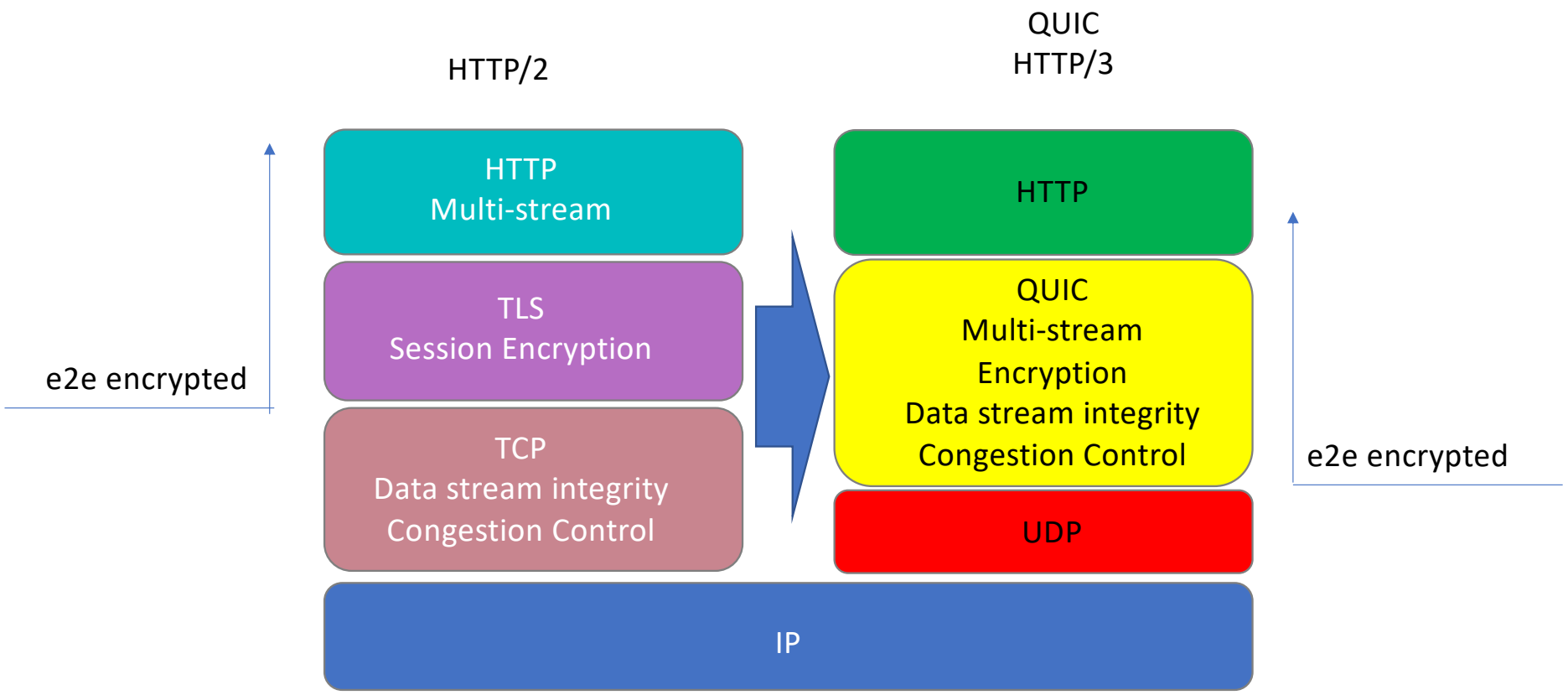
Unlikely!

- It's the same problem .. every host and every piece of active middleware needs to be aware of this new protocol
- Widespread adoption of a new transport protocol in the public Internet is just not a realistic option
  - Just look at the protracted adoption saga for IPv6 to understand the issues around the dynamics of technology adoption in a highly decentralized environment

Borrowing from the Past

- The Internet was conceived as an Inter-net, an overlay network that provided an end-to-end model of connectivity layered above a disparate collection of networks
- Why not place a new transport protocol ABOVE an end-to-end UDP service as an overlay transport?
  - And why not scramble the fields with end-to-end encryption to deter network middleware from intruding (and ossifying) the protocol?
- Which gives us QUIC!

QUIC is a mashup of TCP and TLS



QUIC is…

Constructed upon a basic UDP datagram service

- All other transport services (data integrity, session control, congestion control, encryption) are shifted upwards in the protocol stack towards the application. A host platform may provide a QUIC API as part of the host library, but the application can also provide its own QUIC service independent of the host

QUIC is…

So much more than just "*encrypted TCP over UDP*"

- Support for multi-stream multiplexing that avoids head-of-line blocking and exploits a shared congestion and encryption state
- Faster - Combines transport and encryption setup exchange in a single 3-way exchange at session start, and supports fast reopen
- Customisable - QUIC implementations can use individual flow controllers
- QUIC places its transport control fields inside the encryption envelope using TLS 1.3, so QUIC features minimal exposure to the network
- Supports Remote Procedure Call service models as well as bit-streaming and datagram services

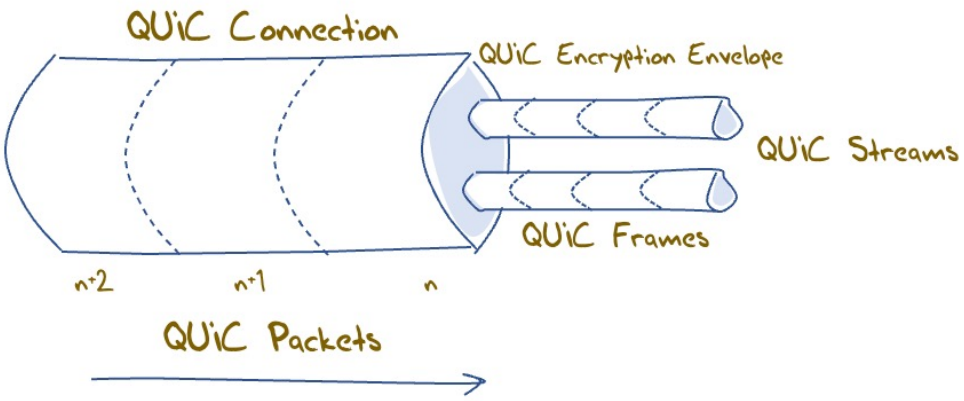
QUIC is address agile

- NATs are potentially hostile to QUIC because of the outer UDP wrapper
  - A NAT may rebind a QUIC session (shift the externally visible address/port of a host during a session), as NATs are not generally aware of UDP streaming states
- QUIC uses a persistent “connection ID”
  - If a host receives a QUIC frame with the same connection ID and a new source IP address / port it will send a challenge by way of a random value that should be echoed back. This is all performed within the e2e encryption envelope. That way a QUIC e2e session can map into new address/port associations on the fly

QUIC also…

- Is IP **fragmentation intolerant** – QUIC uses PMTUD, or defaults to 1,200 octet UDP payloads
- **Never retransmits a QUIC packet** – retransmitted data is sent in the next QUIC packet number – this avoids ambiguity about packet retransmission
- Extends TCP SACK to **256 packet number ranges** (up from 3 in TCP SACK)
- Separately encrypts each QUIC packet – no inter-packet dependencies on decryption
- May load multiple QUIC packets in a single UDP frame

QUIC flow structuring



A QUIC connection is broken into "streams" which are reliable data flows – each stream performs stream-based loss recovery, congestion control, and relative stream scheduling for bandwidth allocation

QUIC also supports unreliable encrypted datagram delivery

QUIC is:

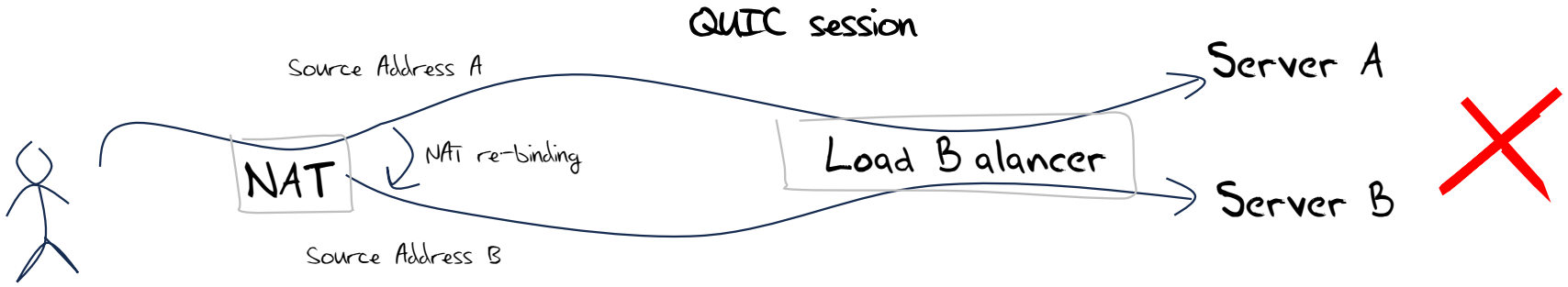
- A logical evolutionary step for transport services, providing more flexibility, faster connection setup, and a larger set of transport services

- It’s what we should expect from a capable modern transport protocol!

But…

QUIC and Network Load Balancing

- Front-end load balances performing load balancing on UDP flows using the UDP connection 5-tuple
- If the remote end performs NAT rebinding the load balancer will be thrown by this shift, and it has no direct visibility into the e2e session to uncover the connection ID



QUIC and Network Load Balancing

- Front-end load balances performing load balancing on UDP flows using the UDP connection 5-tuple
- If the remote end performs NAT rebinding the load balancer will be thrown by this shift, and it has no direct visibility into the e2e session to uncover the connection ID

- **If we really want large scale QUIC with front-end load balancing and if we still need to tolerate NATs then we will need to think about how the end point can share the connection ID state with its front-end load balancer, or how to terminate the QUIC session in the front-end and use a second session to a selected server**

QUIC and NIC Offloading

- Many (most?) NICs these days offer ”TCP Offloading”
  - Platform sends a large data buffer to the NIC, and the processor in the NIC performs TCP segmentation
  - The NIC reassembles smaller received data units to a larger unit before raising an interrupt to the host processor
  - Relieves the CPU from TCP processing overheads, improving server capacity
- Offloading QUIC to NICs is work-in-progress
  - But its looking good – QUIC is well suited to device offload
  - There is a need for mods to kernel and network drives, as well as QUIC libraries
  - It’s not here just yet, but it is looking promising!

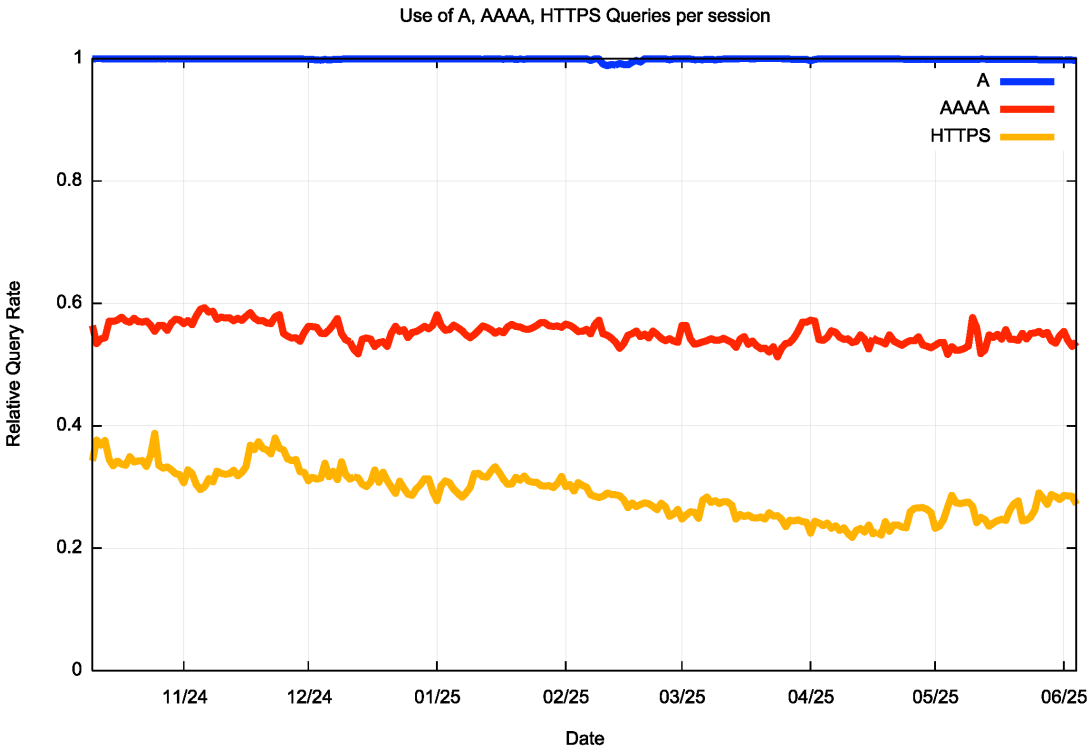
Measuring QUIC

Triggering QUIC in HTTP

1 - Use the DNS to trigger QUIC:

- Set up an HTTPS record for each server name, with value: **alpn="h3"**

DNS HTTPS Query Rate



The query volume for DNS names now includes a query for the HTTPS record in addition to the A and AAAA queries

This has increased the total query volume by some 33%

Triggering QUIC in HTTP

1 - Use the DNS to trigger QUIC:

- Set up an HTTPS record for each server name, with value: **`alpn="h3"`**

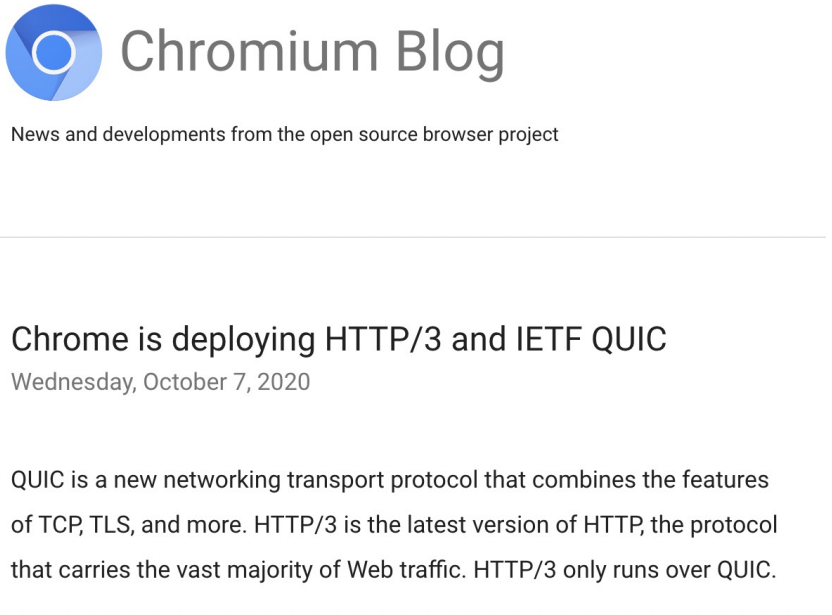
2 - Use content-level controls to trigger QUIC:

- Add **`Alt-Svc: h3=":443"`** to the HTML headers

(This second method requires a subsequent query in a distinct HTTP session to allow the client to use the Alt-Svc capability.)

Setting Expectations

- Chrome has a dominant share of browser instances - roughly, some 65%*
- And Chrome has been supporting a switch to QUIC via the Alt-Svc directive since 2020



Chromium Blog

News and developments from the open source browser project

Chrome is deploying HTTP/3 and IETF QUIC

Wednesday, October 7, 2020

QUIC is a new networking transport protocol that combines the features of TCP, TLS, and more. HTTP/3 is the latest version of HTTP, the protocol that carries the vast majority of Web traffic. HTTP/3 only runs over QUIC.

** Oberlo.com*

Setting Expectations

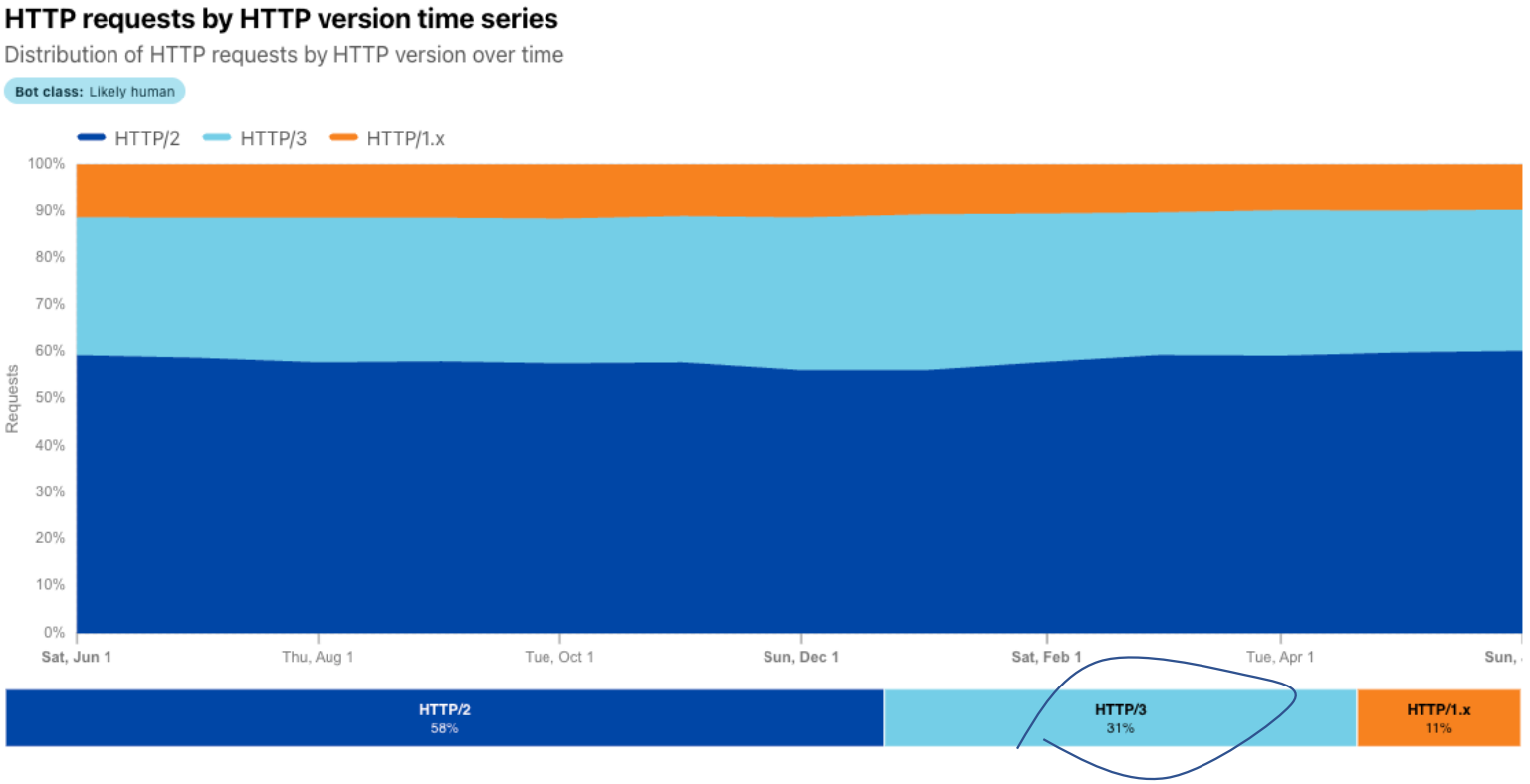
- Chrome has a dominant share of browser instances - roughly, some 65%*
  - And Chrome has been supporting a switch to QUIC via the Alt-Svc directive since 2020
- Apple Safari is now supporting QUIC, using the DNS **`apln`** directive
- So a QUIC-aware server platform should be seeing **up to** 85% of its sessions using QUIC
  - This figure is probably not achievable as the content level control requires some precise conditions for the “second” visit:
    - long enough between visits for the session keepalive timer to expire
    - Short enough such that the local cache of server capabilities has not expired

** https://gs.statcounter.com/browser-market-share*

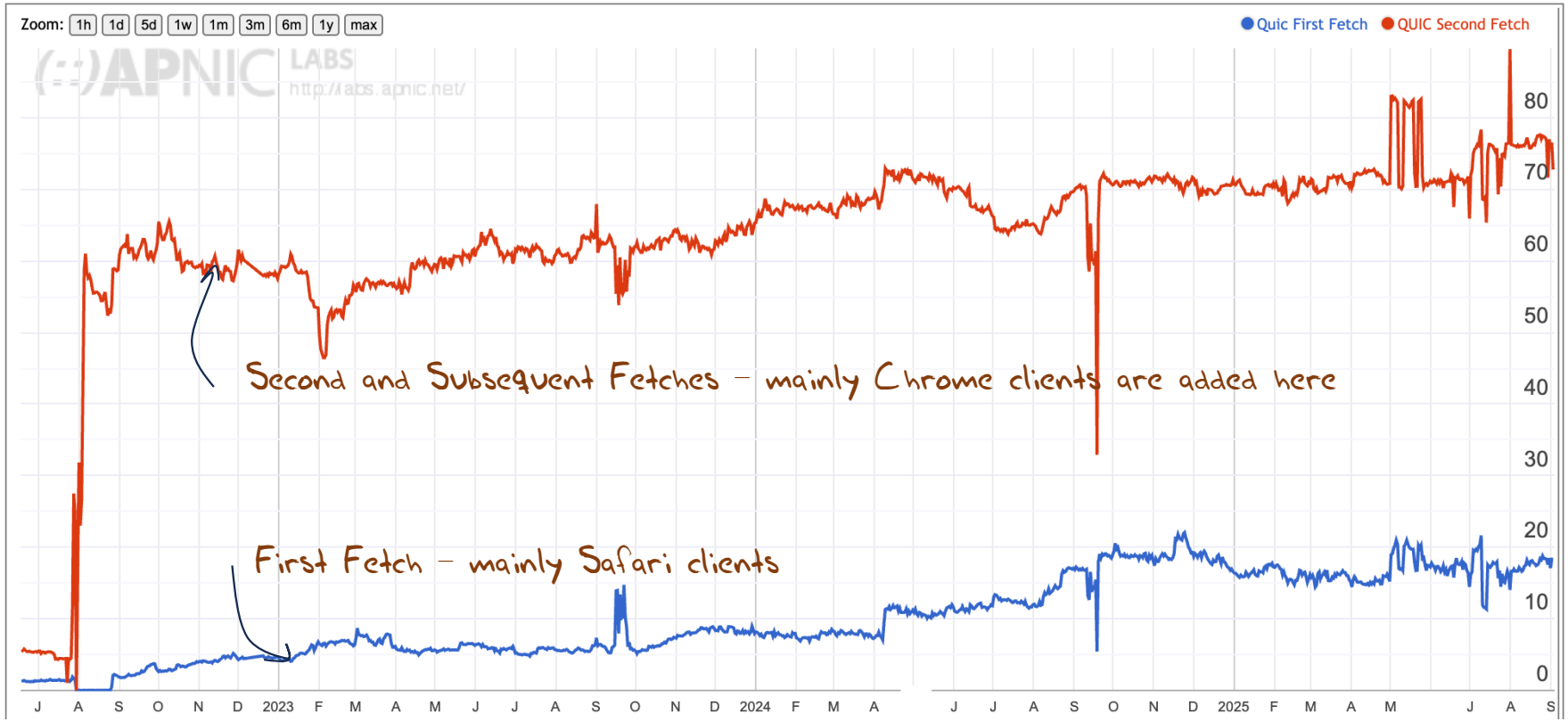
Cloudflare's Numbers - 31%

12 Month Time Series

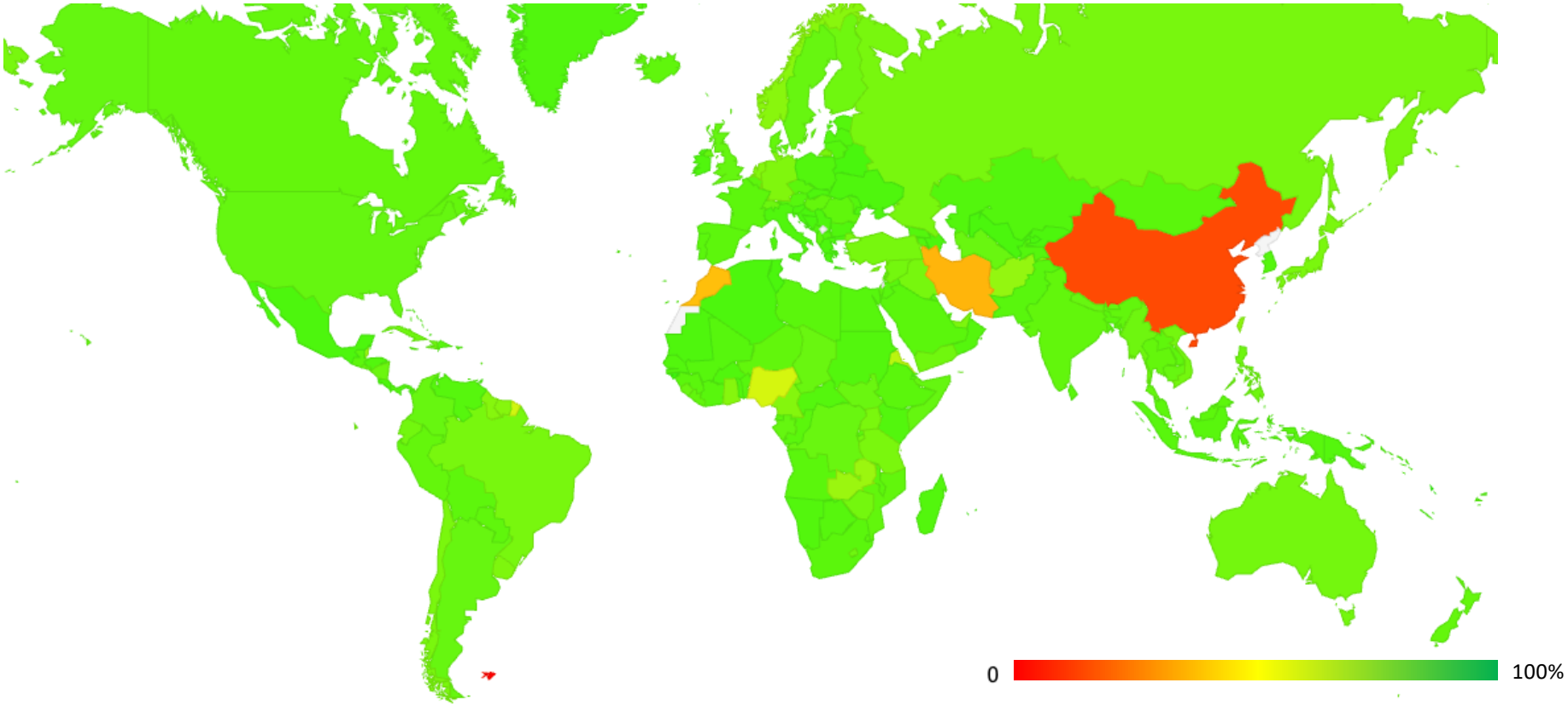
Cloudflare Radar



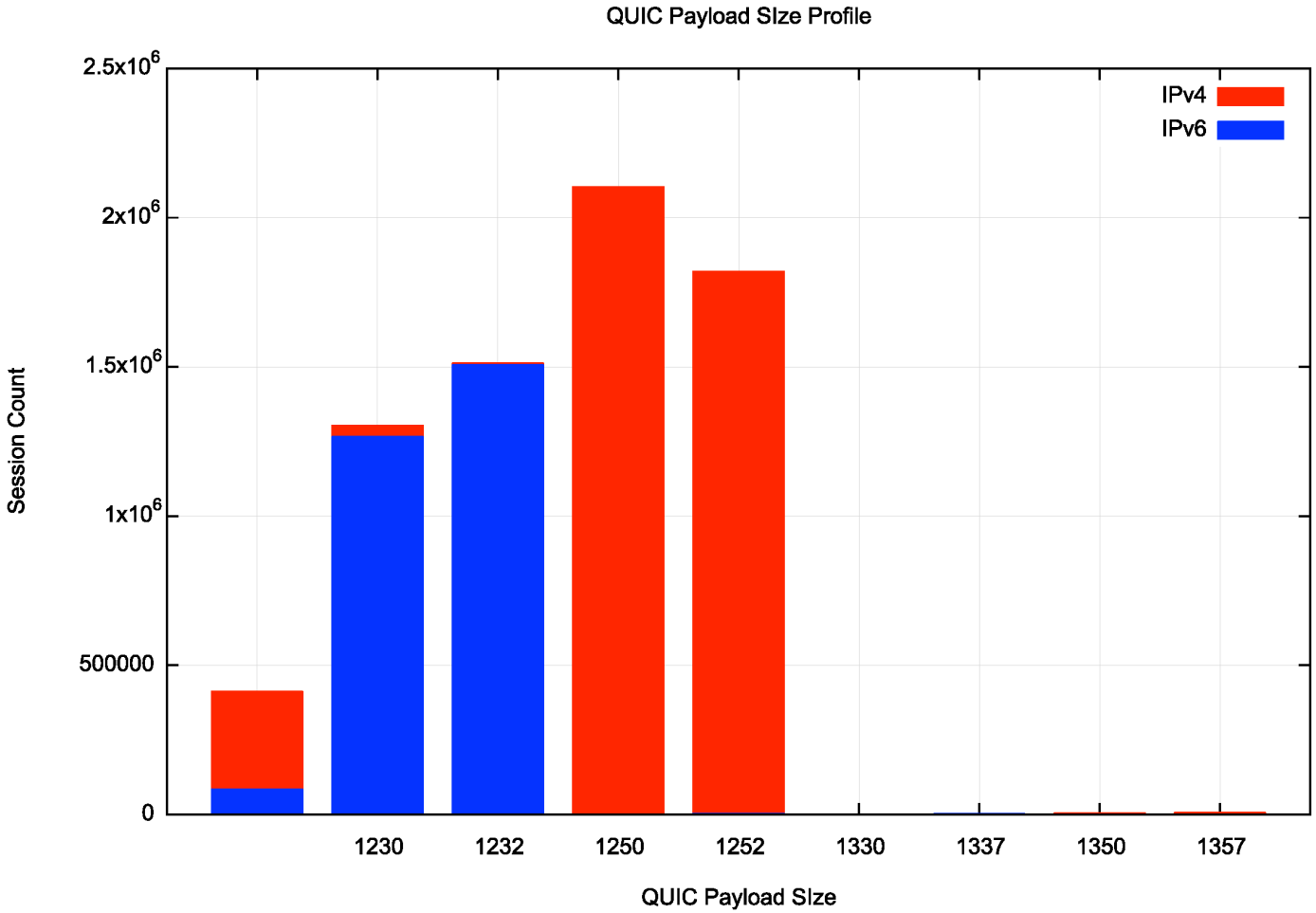
APNIC's Numbers - 75%



APNIC Measured QUIC Use - June 2025



QUIC Payload Sizes



This data has been collected from a single day of measurement (4/6/25)

This disparity between V4 and V6 appears to reflect a popular implementation's design choice to aim at an IP MTU of 1,280 bytes

Browsers and QUIC Use

		HTTPS Query	QUIC First Fetch	QUIC Sub. Fetch	NO Quic
Chrome	9,258,909	174,193	96,824	8,139,214	1,022,871
Safari	5,085,639	4,727,457	2,761,376	20,246	2,304,017
Edge	67,009	47,441	16,776	37,957	12,576
Firefox	12,989	3,689	4,783	5,789	2,417
Opera	4,054	2,218	1,715	1,544	795
Others	29,540	9,828	3,427	15,975	9,838
	14,458,140	4,964,826	2,884,901	8,220,725	3,352,514

Browsers and QUIC Use

		HTTPS Query	QUIC First Fetch	QUIC Sub. Fetch	NO Quic
Chrome	64.0%	3.5%	3.4%	99.0%	30.5%
Safari	35.2%	95.2%	95.7%	0.2%	68.7%
Edge	0.5%	1.0%	0.6%	0.5%	0.4%
Firefox	0.1%	0.1%	0.2%	0.1%	0.1%
Opera	0.0%	0.0%	0.1%	0.0%	0.0%
Others	0.2%	0.2%	0.1%	0.2%	0.3%
Total	100%	100%	100%	100%	100%

QUIC Use

- If QUIC access is supported by the current releases by both the major browsers then we should see a high QUIC use rate when the ability to use QUIC is signaled by both methods (alt-svc and DNS HTTPS)
- What do we see?
- In most locales the **alt-svc** method of triggering QUIC is supported by browsers and network infrastructure
- What about the DNS HTTPS method of triggering QUIC?
  - Who uses a DNS HTTPS query?
  - Are HTTPS responses being filtered by DNS infrastructure in some cases?

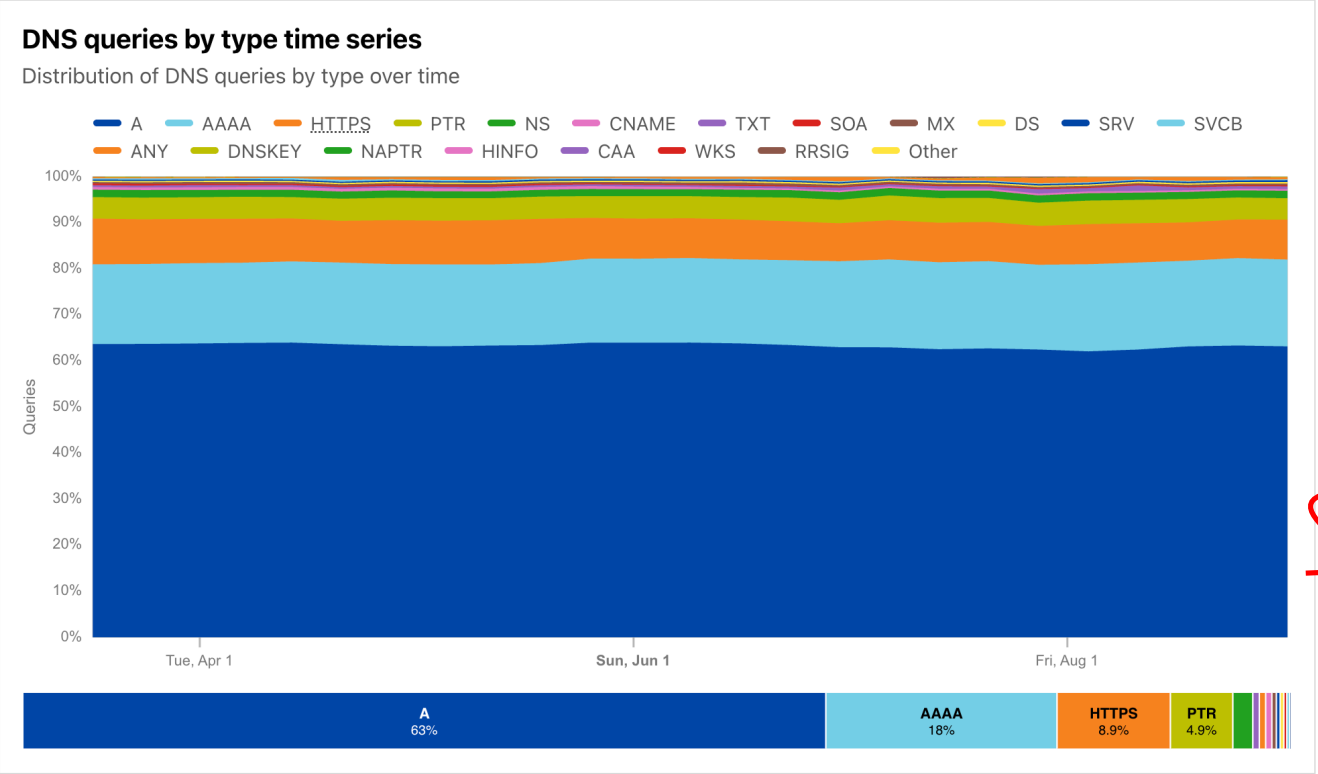
The DNS HTTPS record

- The HTTPS record can also contain **ipv4hint** and **ipv6hint** attributes
- Any A and AAAA records for a name will be used by a client in preference to these hint attributes
- But if there is no A and no AAAA record in the zone, then a HTTPS-aware client will be forced to use these address hint attributes
- Let's try that, and allow the client to use either HTTP/2 OR HTTP/3:

```
test_name   IN   HTTPS    1   .   alpn="h2,h3" ipv4hint=192.0.2.1 ipv6hint=2001:db8::1
```

DNS HTTPS Use Rate

How many users query HTTPS records?



DNS Query Data from Cloudflare Radar

A – 63%

AAAA – 18%

HTTPS – 8.9%

DNS HTTPS Use Rate

How many users can use DNS HTTPS responses?

Data collected over a 24-hour period (7/7/2025)

	All		Chrome		Safari		Others	
Samples	13,177,108		9,487,295		3,602,160		87,653	-
DNS HTTPS Query	3,708,895	28.1%	157,695	1.7%	3,506,664	97.3%	44,536	50.8%
Web Fetch (h2/h3)	3,480,873	26.4%	5,957	0.1%	3,469,867	96.3%	5,049	5.8%
Web Fetch (QUIC)	2,710,668	20.6%	4,793	0.1%	2,701,516	75.0%	4,359	5.0%

Few Chrome users (1.7%) perform an HTTPS query, and even fewer (0.1%) followup with a fetch of the web object.

Most Safari users (97.3%) perform an HTTPS query, and most (96.3%) followup with a fetch of the web object. Fewer users (75%) prefer to use QUIC to perform web object retrieval when given the choice.

DNS HTTPS Use Rate

How many users can use DNS HTTPS responses?

Data collected over a 24-hour period (7/7/2025)

	All		Chrome		Safari		Others	
Samples	13,177,108		9,487,295		3,602,160		87,653	-
DNS HTTPS Query	3,708,895	28.1%	157,695	1.7%	3,506,664	97.3%	44,536	50.8%
Web Fetch (h2/h3)	3,480,873	26.4%	5,957	0.1%	3,469,867	96.3%	5,049	5.8%
Web Fetch (QUIC)	2,710,668	20.6%	4,793	0.1%	2,701,516	75.0%	4,359	5.0%

Why is Safari not using QUIC in 25÷ of cases?

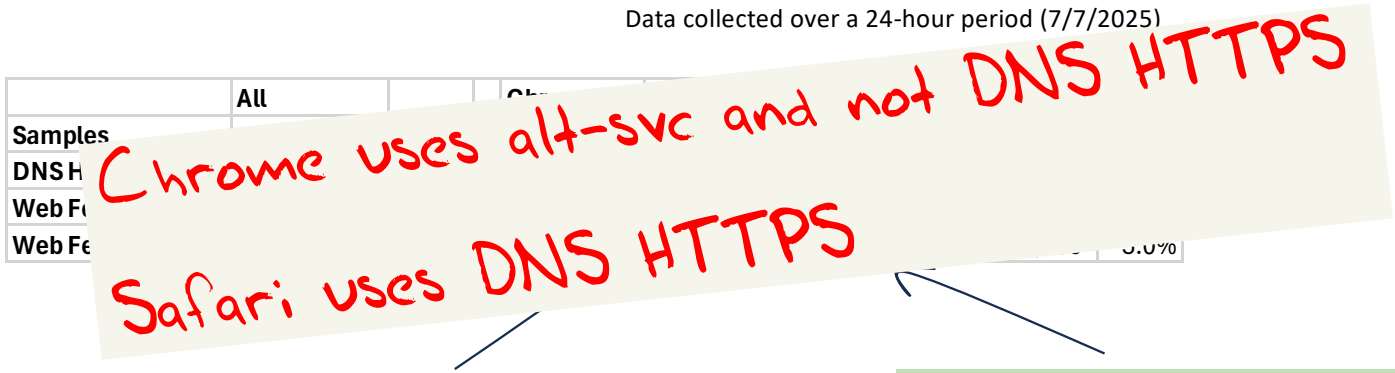
Few Chrome users (1.7%) perform an HTTPS query, and even fewer (0.1%) followup with a fetch of the web object.

Most Safari users (97.3%) perform an HTTPS query, and most (96.3%) followup with a fetch of the web object. Fewer users (75%) prefer to use QUIC to perform web object retrieval when given the choice.

DNS HTTPS Use Rate

How many users can use DNS HTTPS responses?

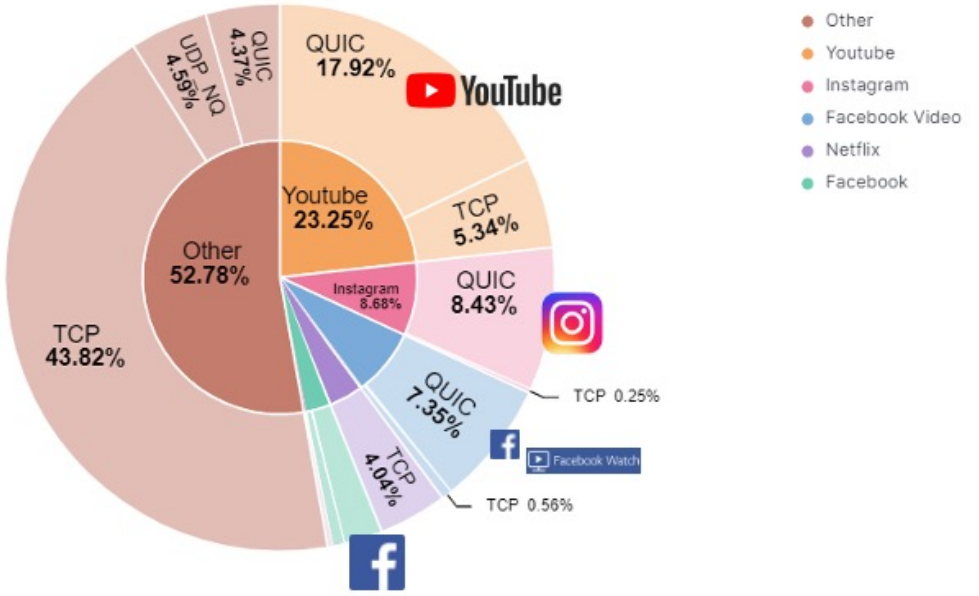
Data collected over a 24-hour period (7/7/2025)



Few Chrome users (1.7%) perform an HTTPS query, and even fewer (0.1%) followup with a fetch of the web object.

Most Safari users (97.3%) perform an HTTPS query, and most (96.3%) followup with a fetch of the web object. Fewer users (75%) prefer to use QUIC to perform web object retrieval when given the choice.

Network Traffic Volume



*source EU Operator 2022

Presentation to RIPE 86: The New Encrypted Protocol Stack and How to Deal with it – Bart van de Velde, Cisco

Why is QUIC important?

Because QUIC is fast

Because QUIC encrypts everything

- No visible transport control settings
- No visible Server Name Indication in the crypto-setup
- No visible traffic profile other than inter-packet timing
- And if you use a MASQUE-based VPN then there no residual visibility!

What does this mean for TCP?

It’s not looking all that good for TCP’s prospects

- QUIC not only does faster start up, but it supports multi-channel in a frictionless manner
- QUIC resists network operator efforts to perform traffic shaping through direct manipulation of TCP control parameters
- QUIC allows the application service provider to control the congestion behaviour of its sessions

What does this mean for TCP?

Normally you would expect any transition from TCP to QUIC to take forever

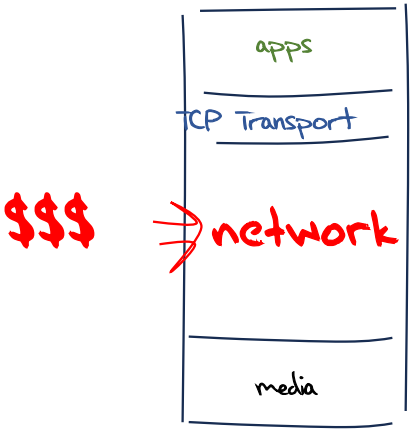
BUT:

- QUIC gives benefit to adopters through more responsive web services
- QUIC does a better job of hiding content, which is a benefit to the service operator
- QUIC has fewer external dependencies
- QUIC can be deployed on a piecemeal basis

So it all may be over for TCP in a very small number of years!

What does this mean for the Internet?

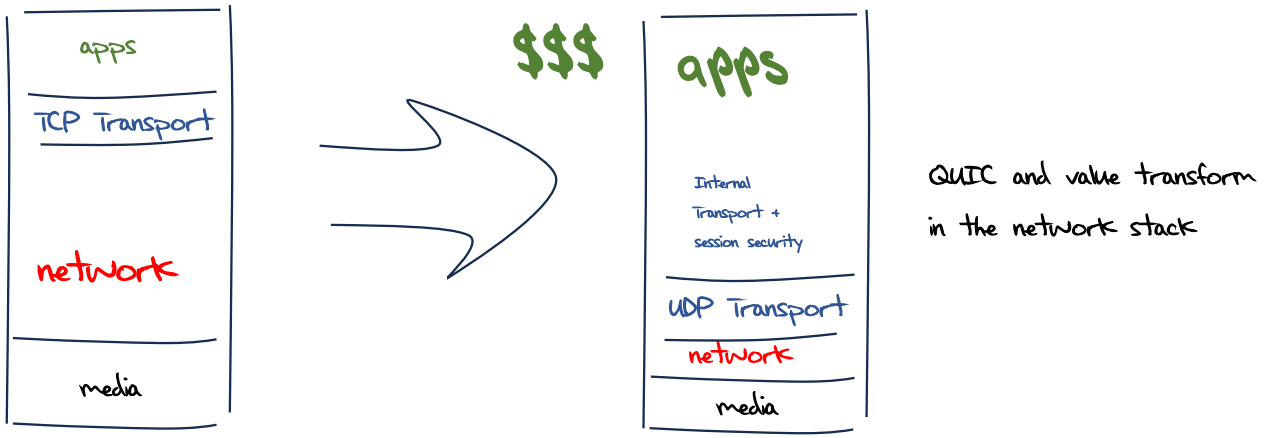
- IP was a ***network*** *protocol* that provided services to attached devices
- The network service model used by IP was minimal
  - Packets may be dropped, fragmented, duplicated, corrupted and/or reordered on their path through the network
  - It's left to the edge systems to recover from this network behaviour.
- Efforts to expand the network's role have foundered
  - QoS has just got nowhere!
  - Various forms of source-directed forwarding are resisted by network operators who want control over traffic engineering



The new Networking Space

And this is why QUIC is so interesting – it is pushing both network carriage and host platform into commodity roles in networking and allowing applications to effectively customize the way in which they want to deliver services and dominating the entire networked environment

QUIC is the application's view of what Transport should be!



What does this mean for the Internet?

- The relationship between applications, hosts and networks has soured into mutual distrust and suspicion
- The application now defends its integrity by wrapping up as much of the service transaction with encryption and indirection
- QUIC (and MASQUE) is an intrinsic part of this process of wrapping up traffic in encryption and redirection
- For the network operator there is little left to see
- And I suspect that there is no coming back from here!

What can a Network Operator Do?

- When **all** customer traffic is completely obscured and encrypted?
  - Traffic Shaping?
  - Regulatory Requirements for traffic interception?
  - Load Balancing / ECMP

The new Internet Space

“What you can’t dominate, you commoditise*”

- Vertically integrated service providers have faded away into history - the deregulated competitive service industry continues to specialize rather than generalize at every level
- Carriage is no longer an inescapable monopoly - massively replicated content can be used as a substitute for many carriage service elements
- Control over the platform is no longer control over the user. Operating systems have been pushed back into a basic task scheduling role, while functions are being absorbed into the application space

* A related quote is Peter Thiel’s “Competition is for losers!”

Thanks!