Networking in the Penumbra

Geoff Huston AM
APNiC

The Trusted Network

Networks enjoyed a privileged position of being able to observe:

- Who is communicating with whom
- What they are saying to each other

The Trusted Network

Users have an expectation of privacy in their communications

- This expectation was often reinforced through regulatory measures intended to constrain public network operators from disclosing knowledge gained through network operation

The Erosion of Trust

Trust has been eroded by intrusive middleware that collects aggregate (and sometimes specific) data on user behaviours

- The general adoption of advertising revenue as a means of funding for service platforms act as a major incentive to assemble detailed profiles of individual users: age, gender, location, educational level, marital status, income, interest, purchase history,…
- The better the profile the higher the value of the user to the advertiser

The Erosion of Trust

This network position of trust was further eroded by leakage of the activities of US state-based actors performing various forms of mass surveillance on network users

- The Snowden Papers was a watershed moment for the Internet
- But it was by no means the first time, nor was it the last
- Large scale state-sponsored surveillance continues

So how did we react?

RFC 7258

Pervasive Monitoring is an attack on privacy

> “The IETF community's technical assessment is that PM is an attack on the privacy of Internet users and organisations. The IETF community has expressed strong agreement that PM is an attack that needs to be mitigated where possible, via the design of protocols that make PM significantly more expensive or infeasible.”

RFC 7258 – May 2014

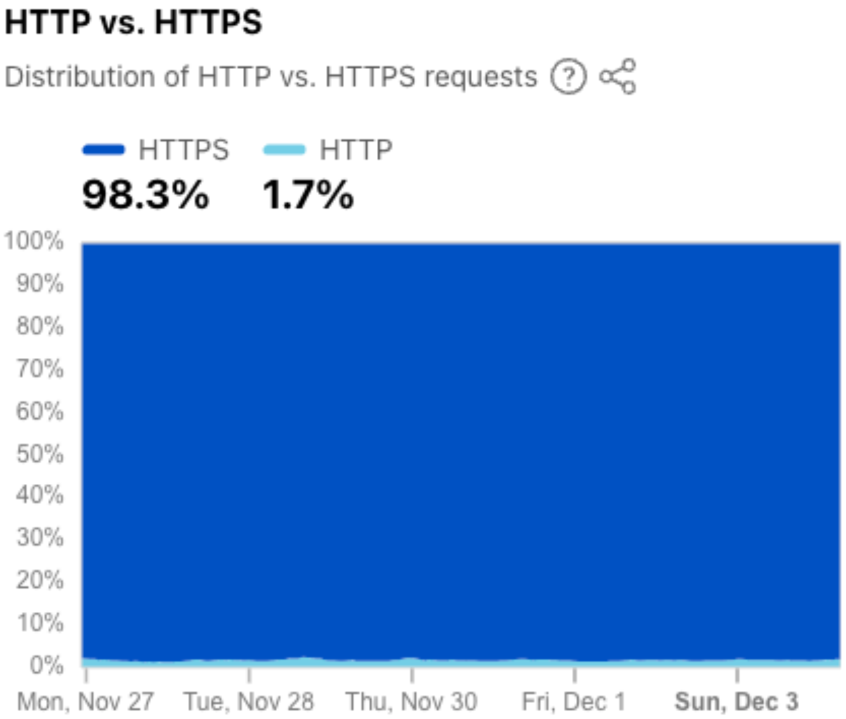
What did this mean?

Changes to the Applications

Hiding the Web

- Shift to use TLS for all web transactions – HTTPS
  - TLS authenticates the identity of the server to the client
    - Is this service name authentic? Can the service operator demonstrate to the client that is has knowledge of the private part of the key pair that is associated with this DNS service name?
  - Service transactions are encrypted
    - TLS generates a session key used to encrypt all subsequent on-the-wire data

HTTPS Today



HTTP vs. HTTPS

Distribution of HTTP vs. HTTPS requests

HTTPS 98.3%

HTTP 1.7%

https://radar.cloudflare.com/adoption-and-usage

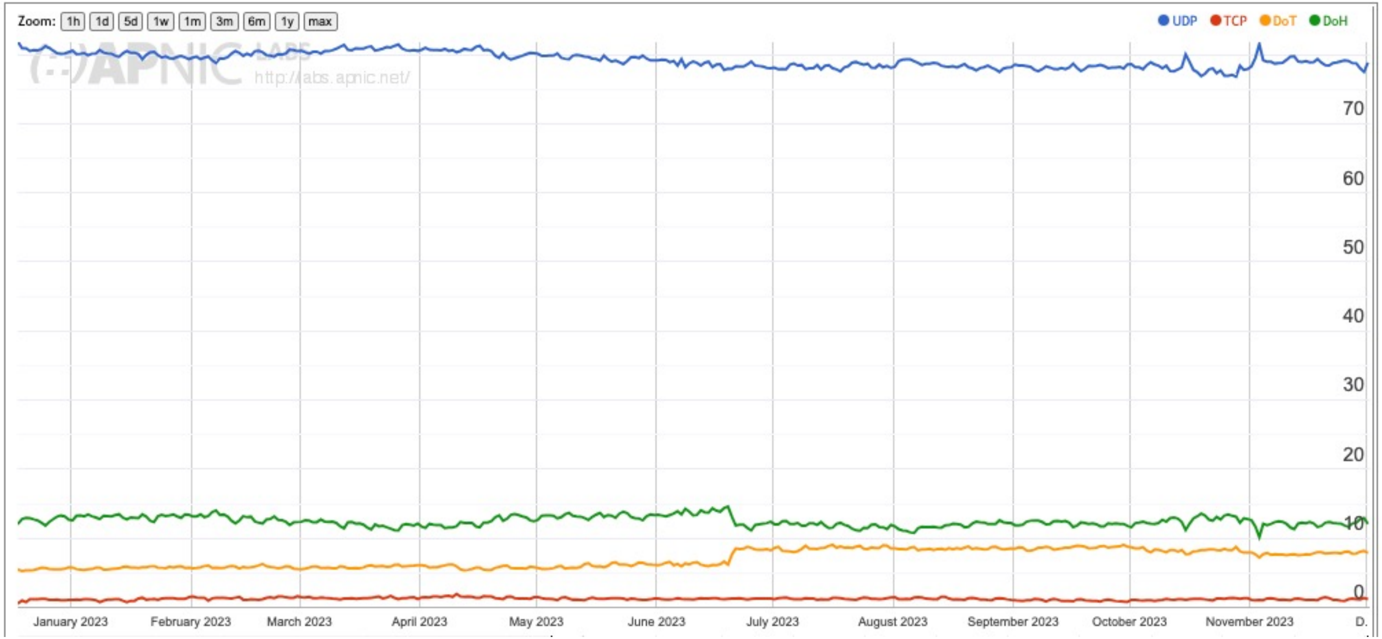
Changes to the Applications

Hiding the DNS

- Hide the query and response in DNS resolution transactions from the network
- The initial work has concentrated on hiding the DNS query names from the network by encrypting the DNS data exchanged
  - DNS over TLS
  - DNS over HTTPS/2 and DNS over HTTPS/3

DoH, DOT Today

Cloudflare Open Recursive Resolver DNS Query Profile for World (XA)



DNS over HTTPS
DNS over TLS

https://stats.labs.apnic.net/edns

Can we go further?

- Can we hide the two ends from each other such that at no point in the network (and even at the server) are the two ends of the transaction visible at once?
- Can we also selectively obscure the content of the transaction such that the endpoints and the content of the transaction are not simultaneously discoverable

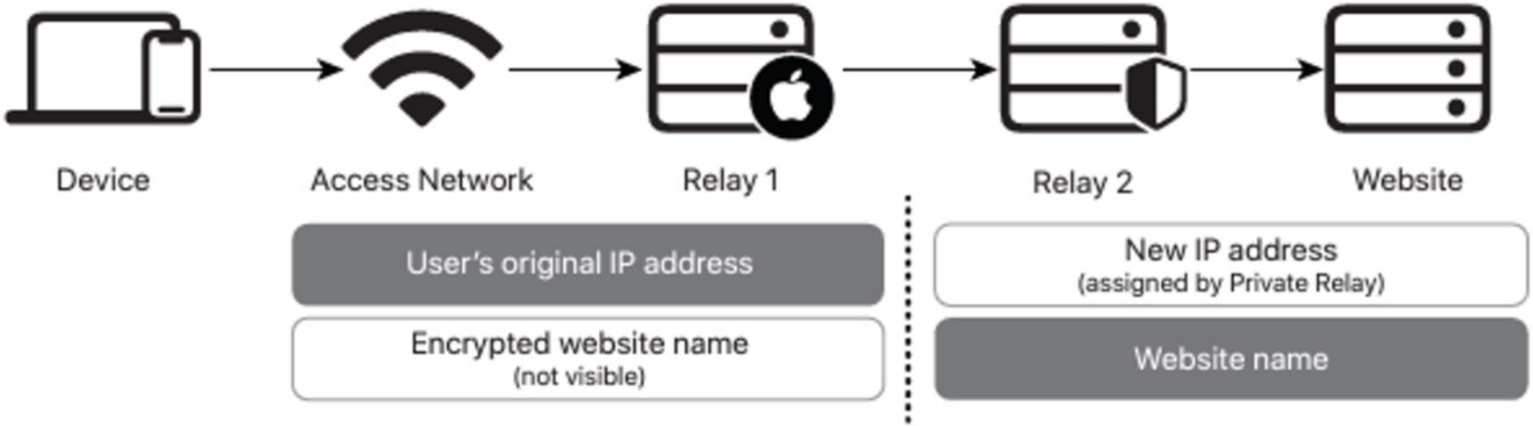
MASQUE and Relays

- With the use of 2-layer encryption and active relays then its possible to hide the endpoints from the network
- There is no single network observation point that can put together the combination of the service identity and the identification of the two endpoints of the service transaction
- Only the client endpoint knows its own identity and service, but does not know the identity used by the relay to present the service transaction to the server
- The server may use the application-level identity of the client, but does not know the client’s network-level identity (IP address)
- This technique can be used in DNS resolution and HTTPS transactions

Apple Private Relay

When Private Relay is enabled, your requests are sent through two separate, secure internet relays.

- Your IP address is visible to your network provider and to the first relay, which is operated by Apple. Your DNS records are encrypted, so neither party can see the address of the website you're trying to visit.
- The second relay, which is operated by a third-party content provider, generates a temporary IP address, decrypts the name of the website you requested, and connects you to the site.



Sealing up the Peepholes

- Attention has turned to the Server Name Indication (SNI) field in the TLS handshake
  - This is the one part of TLS that is still shown in the clear
  - Efforts to encrypt this field in a robust manner are being studied
  - The most effective way to securely communicate the public key that is used to encrypt the SNI (and the entire ClientHello message) appears to be a TLSA record in the DNS (DANE) using DoT or DoH, using a DNSSEC-signed record
  - A shortcut hack is to use a trusted intermediary (https://blog.cloudflare.com/announcing-encrypted-client-hello/)

Sealing up the Peepholes

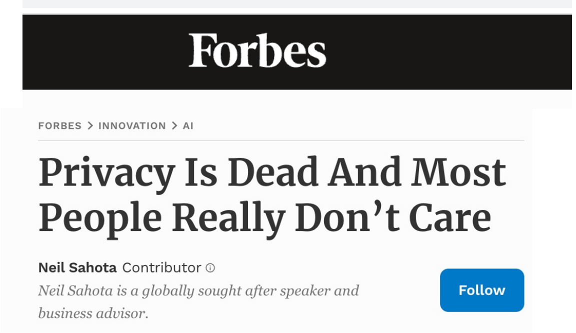
- And there‘s the the Online Certificate Status Protocol, which exposes the IP address of the client and the name of the service that they are visiting
  - Which likely explains why Chrome browsers do not perform “live” certificate revocation checks, and rely instead on short validity periods for certificates

Why are we doing this?

Who wants privacy?

Do users really care?

- Users cheerfully gave up email privacy in exchange for free email services
- Users happily tell Google Search way too much about themselves in exchange for instant answers
- In general, users will happily trade off privacy for access to services



Forbes

FORBES > INNOVATION > AI

Privacy Is Dead And Most People Really Don't Care

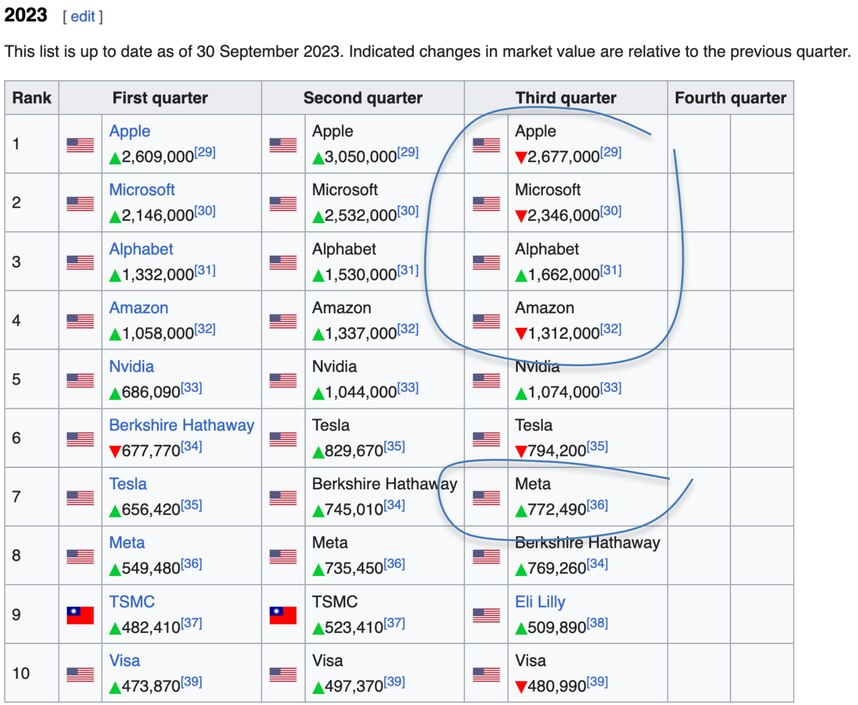
Neil Sahota Contributor ⓘ

Neil Sahota is a globally sought after speaker and business advisor.

If not users, then whom?

If not users, then whom?

The folk with the most to lose!

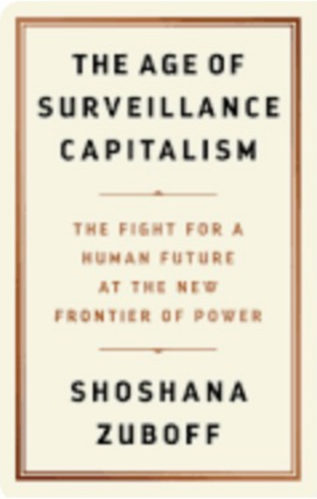


2023 [edit]

This list is up to date as of 30 September 2023. Indicated changes in market value are relative to the previous quarter.

Rank	First quarter		Second quarter		Third quarter		Fourth quarter	
1	🇺🇸	Apple ▲2,609,000[29]	🇺🇸	Apple ▲3,050,000[29]	🇺🇸	Apple ▼2,677,000[29]		
2	🇺🇸	Microsoft ▲2,146,000[30]	🇺🇸	Microsoft ▲2,532,000[30]	🇺🇸	Microsoft ▼2,346,000[30]		
3	🇺🇸	Alphabet ▲1,332,000[31]	🇺🇸	Alphabet ▲1,530,000[31]	🇺🇸	Alphabet ▲1,662,000[31]		
4	🇺🇸	Amazon ▲1,058,000[32]	🇺🇸	Amazon ▲1,337,000[32]	🇺🇸	Amazon ▼1,312,000[32]		
5	🇺🇸	Nvidia ▲686,090[33]	🇺🇸	Nvidia ▲1,044,000[33]	🇺🇸	Nvidia ▲1,074,000[33]		
6	🇺🇸	Berkshire Hathaway ▼677,770[34]	🇺🇸	Tesla ▲829,670[35]	🇺🇸	Tesla ▼794,200[35]		
7	🇺🇸	Tesla ▲656,420[35]	🇺🇸	Berkshire Hathaway ▲745,010[34]	🇺🇸	Meta ▲772,490[36]		
8	🇺🇸	Meta ▲549,480[36]	🇺🇸	Meta ▲735,450[36]	🇺🇸	Berkshire Hathaway ▲769,260[34]		
9	🇹🇼	TSMC ▲482,410[37]	🇹🇼	TSMC ▲523,410[37]	🇺🇸	Eli Lilly ▲509,890[38]		
10	🇺🇸	Visa ▲473,870[39]	🇺🇸	Visa ▲497,370[39]	🇺🇸	Visa ▼480,990[39]		

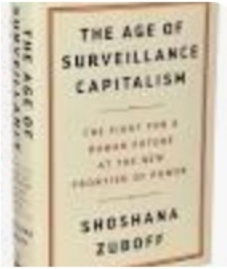
Surveillance capitalism ⋮



THE AGE OF SURVEILLANCE CAPITALISM

THE FIGHT FOR A HUMAN FUTURE AT THE NEW FRONTIER OF POWER

SHOSHANA ZUBOFF



THE AGE OF SURVEILLANCE CAPITALISM

THE FIGHT FOR A HUMAN FUTURE AT THE NEW FRONTIER OF POWER

SHOSHANA ZUBOFF

Who cares about privacy?

- None of the entities who spend large sums to assemble detailed profiles of users want to leak that data to their competitors
- So, privacy is about protecting the core asset of gathering individual profiles of users from:
  - Other services
  - The common host platform
  - Common Infrastructure services
  - The network

Let me rephrase that:

We want to allow **the application** to operate in a mode that obscures its behaviour from:

- Other services
- The common host platform
- Common Infrastructure services
- The network

How do you do that?

By lifting out as much as you can from the lower levels of the protocol stack that are managed by common services and performing it within the application

So, how do you do that?

Transport Privacy

Which means we are looking at how to lift TCP out of the common parts of the host platform and and shift it across to the application

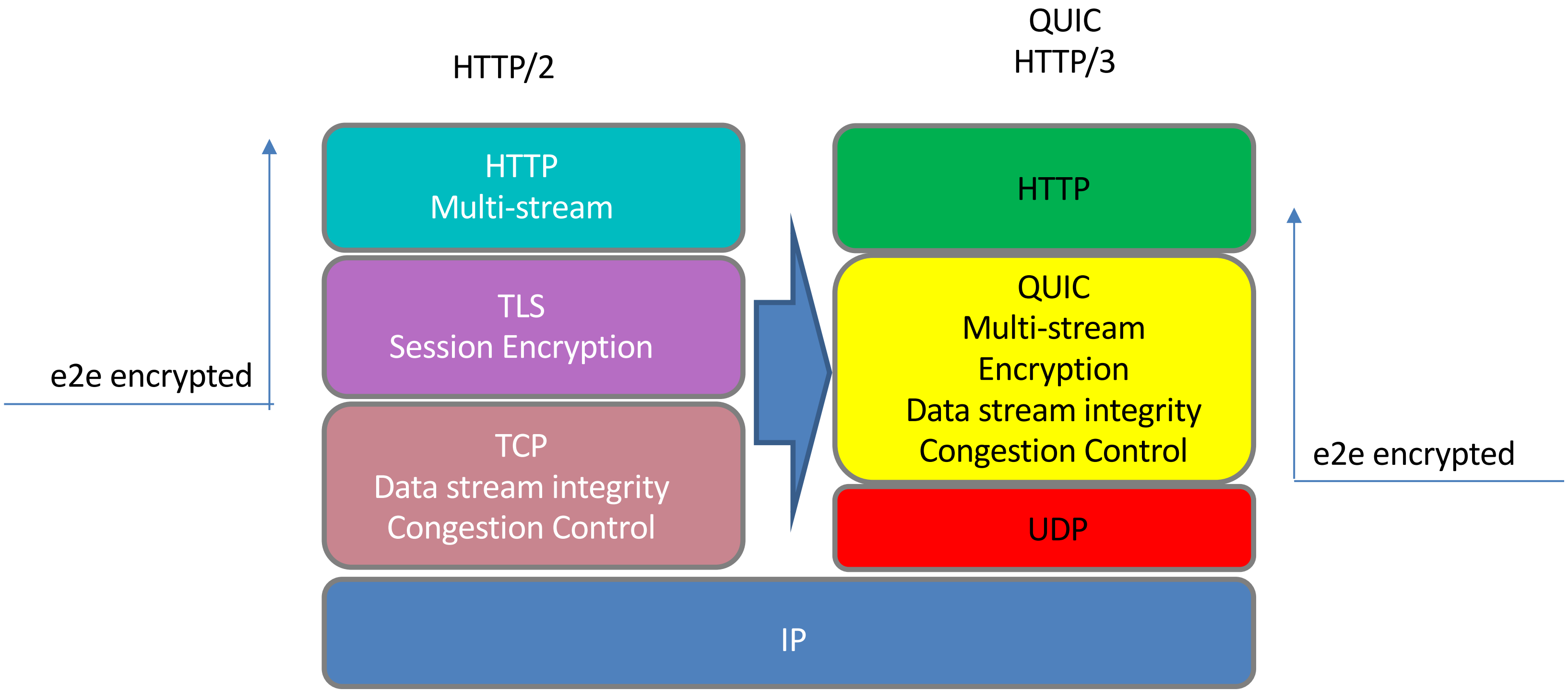
We need to change TCP!

Transport Surgery

How do you change TCP?

- TCP is a kernel function that is defined at the platform level
- Applications have no intrinsic ability to alter the TCP characteristics for the application on a customized basis
- You could try to define a new transport protocol (such as SCTP)
- But the deployed infrastructure (NATs) tends to discard all packets that are not protocol 6 (TCP) or protocol 17 (UDP)
- If you want to bypass kernel handling of TCP and get through existing network filters and middleware then you're forced into using UDP
- So you change TCP by using QUIC!

QUIC is the new TCP



HTTP/2

QUIC
HTTP/3

HTTP
Multi-stream

TLS
Session Encryption

TCP
Data stream integrity
Congestion Control

HTTP

QUIC
Multi-stream
Encryption
Data stream integrity
Congestion Control

UDP

IP

e2e encrypted

e2e encrypted

TCP is..

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 isn't…

- Fully independent of the underlying platform'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
- Everything for everyone – it relies on the application to perform data framing and in-band control

QUIC is…

Constructed upon a transport level framing protocol that offers applications access to the basic IP datagram services offered by IP through the use of UDP

> 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 “*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 per flow
- QUIC places its transport control fields inside the encryption envelope, so QUIC features minimal exposure to the network
- Supports record and Remote Procedure Call service models as well as bit-streaming and datagram services

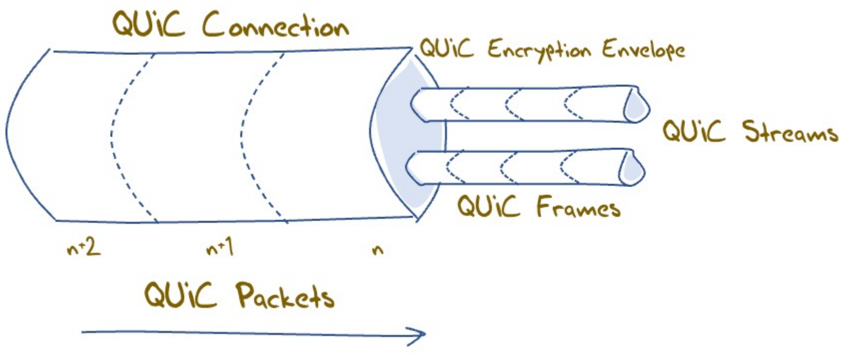
QUIC is address agile

- 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

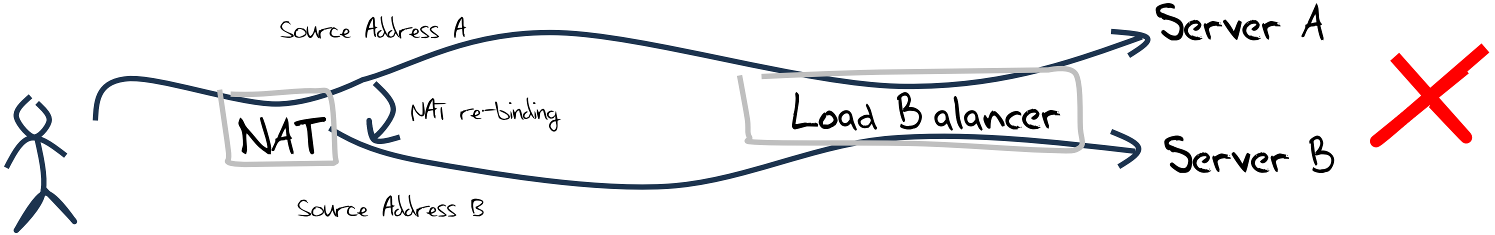
QUIC also supports unreliable encrypted datagram delivery

QUIC and Remote Procedure Calls

- By associating each RPC request/reply with a new stream, QUIC can support asynchronous RPC transactions using reliable messaging
  - This can handle lost, mis-ordered and duplicated RPC messages without common blocking or throttling

QUIC and Load Balancing

- This assumes that a front-end load balancer is capable of 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
- Using UDP to carry sustained high-volume streams may not match the internal optimisations used in server content delivery networks



QUIC and Load Balancing

- This assumes that a front-end load balancer is capable of 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
- Using UDP to carry sustained high-volume streams may not match the internal optimisations used in server content delivery networks

- **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 DOS

- Very little lies outside the encryption envelope in QUIC
- Which means all incoming packets addressed to the QUIC port need to be decrypted
- But the QUIC session uses symmetric crypto so the packet decode overhead is far smaller than an asymmetric crypto load for the same packet rate
- It’s not the best answer, but it’s not disastrous either!

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!

Triggering QUIC in HTTP

Use the DNS to trigger QUIC:

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

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.)

Triggering QUIC in HTTP

Use the DNS to trigger QUIC:

- Set up an HTTPS record for each server name, with value: **`alpn=“h3”`** First Fetch

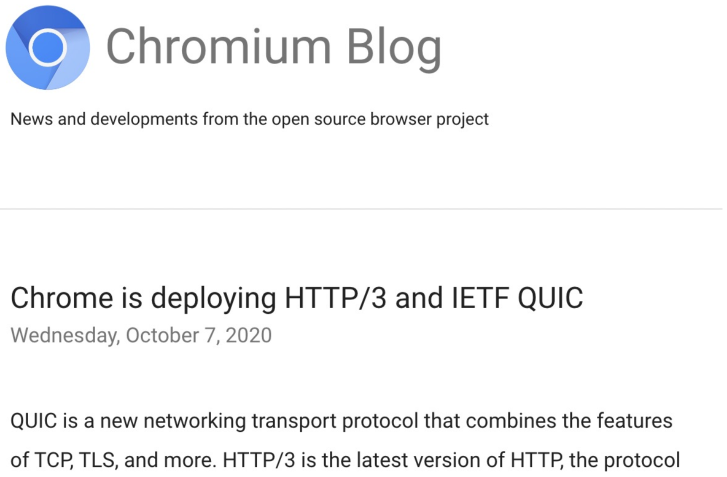
Use content-level controls to trigger QUIC:

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

Second Fetch

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

* Oberlo.com

Setting Expectations

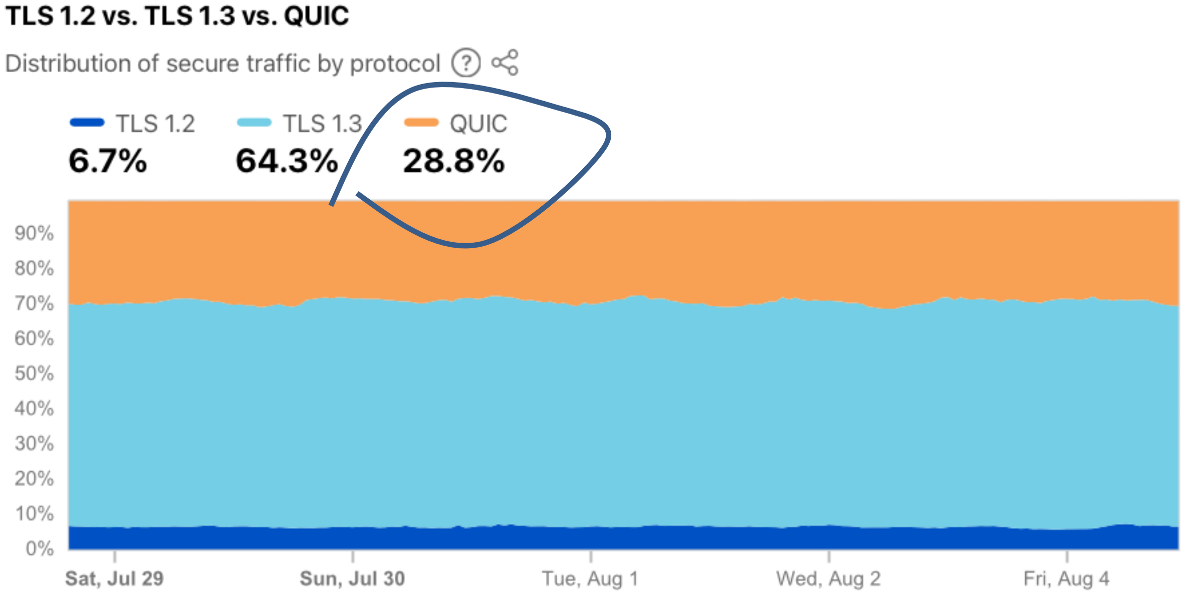
- Chrome has a dominant share of browser instances - roughly, some 65%*
- And Chrome has been supporting a switch to QUIC via the Alt-Svc directive since 2020
- And Apple Safari is now supporting QUIC, using the DNS SVCB **`apln`** directive
- So a QUIC-aware server platform should be seeing some 85% of its sessions using QUIC – right?

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

Cloudflare's Numbers

Cloudflare reports a far lower level of QUIC use

Cloudflare Radar



TLS 1.2 vs. TLS 1.3 vs. QUIC

Distribution of secure traffic by protocol

TLS 1.2	TLS 1.3	QUIC
6.7%	64.3%	28.8%

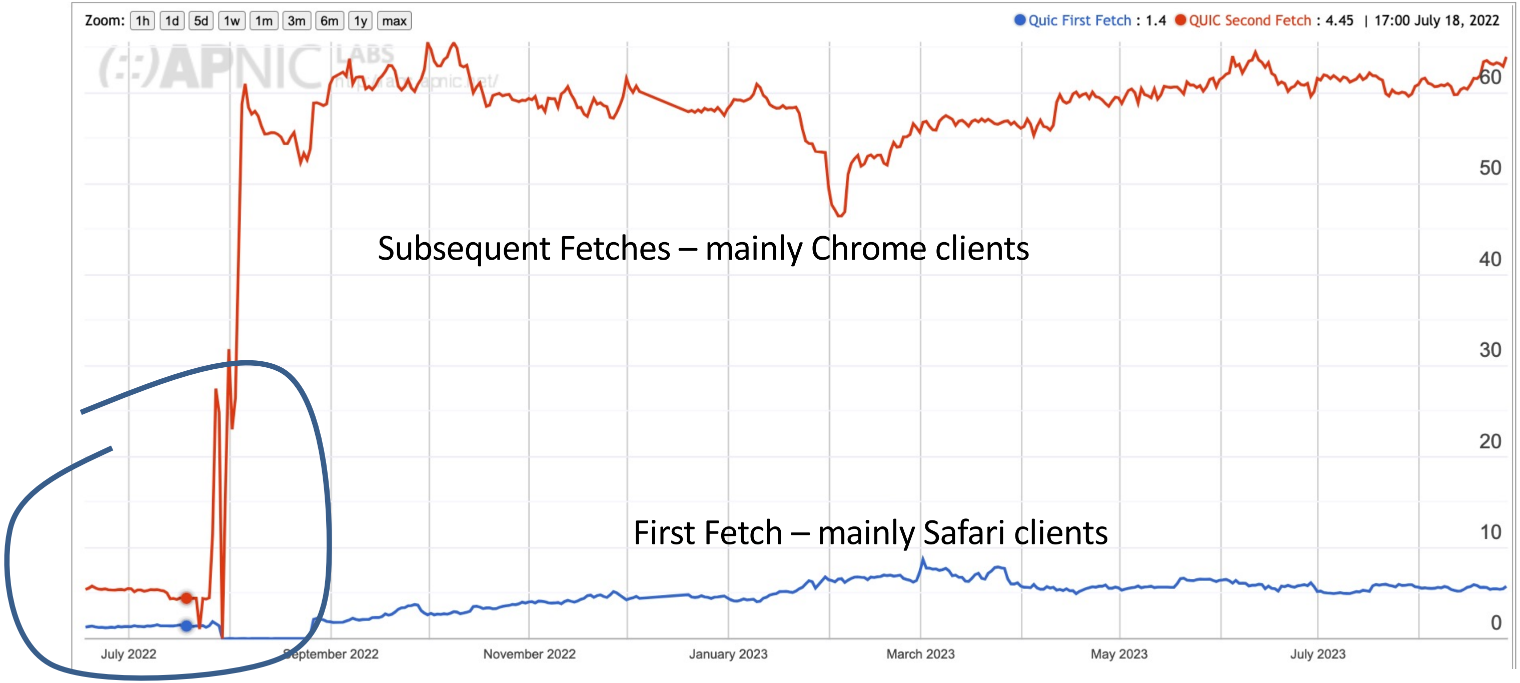
APNIC's QUIC measurement

- We have configured a server to support QUIC sessions
- We support both DNS and content triggers
- The content trigger requires us to measure across multiple fetches within each measurement
  - Which means that we need to carefully set the HTTP/2 session keepalive timer to make this work as intended

Server Session Keepalive Timers

- After much searching under many rocks we were advised that a **server keepalive** timer value of 1 second is too small, as the server drops the QUIC connection too aggressively and the browser client then drops back to using HTTP/2
- The default value of 65 seconds for the server keepalive interval seems to be too long
- So we used a **server keepalive** value of 20 seconds…

QUIC Use

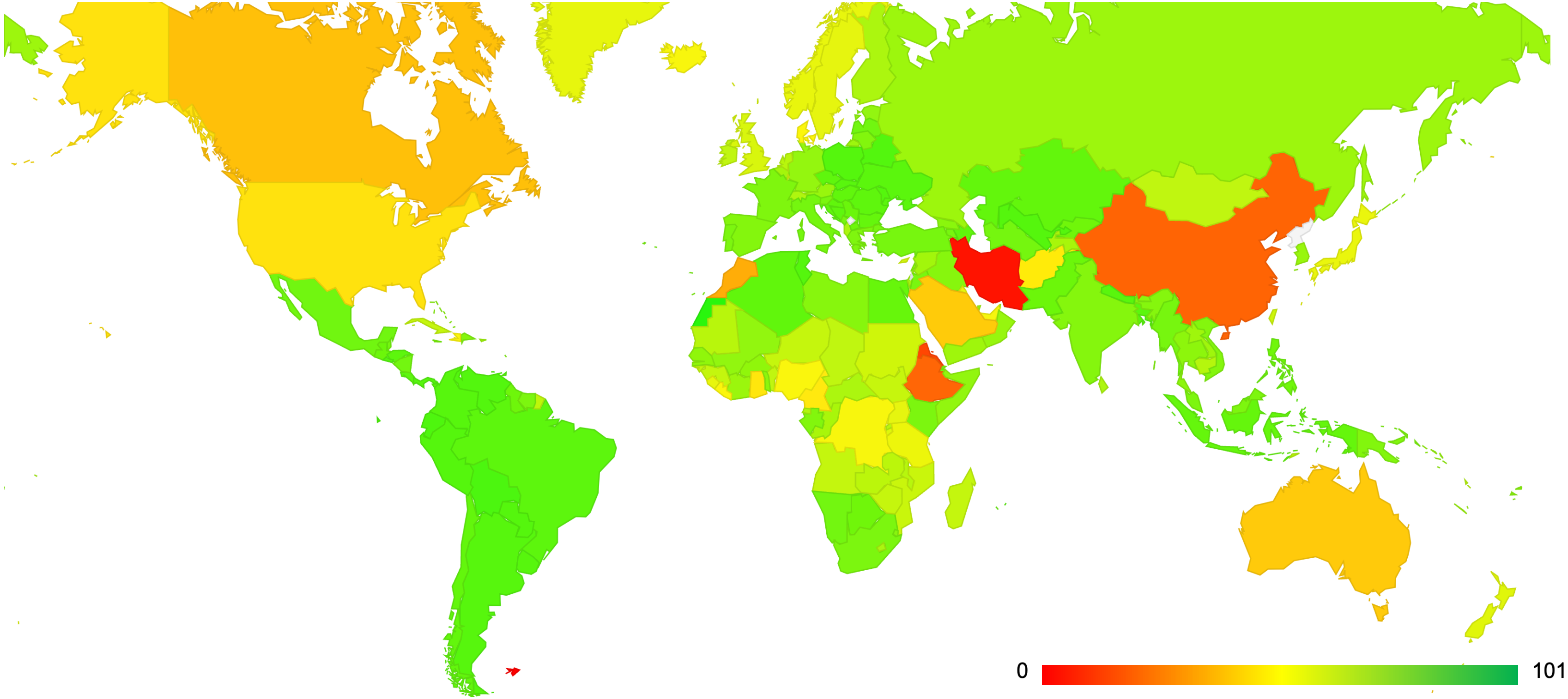


Subsequent Fetches – mainly Chrome clients

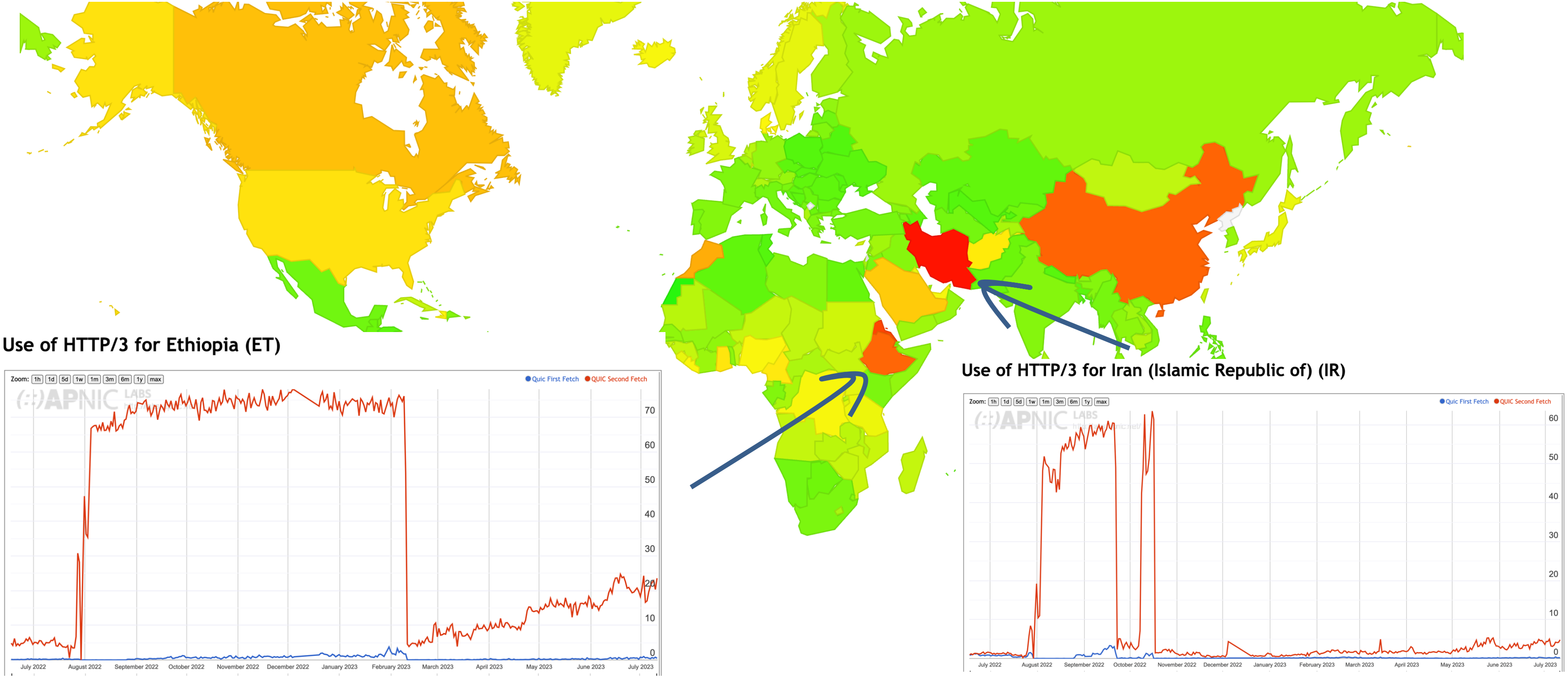
First Fetch – mainly Safari clients

Playing with keepalive parameters!

QUIC Use - July 2023



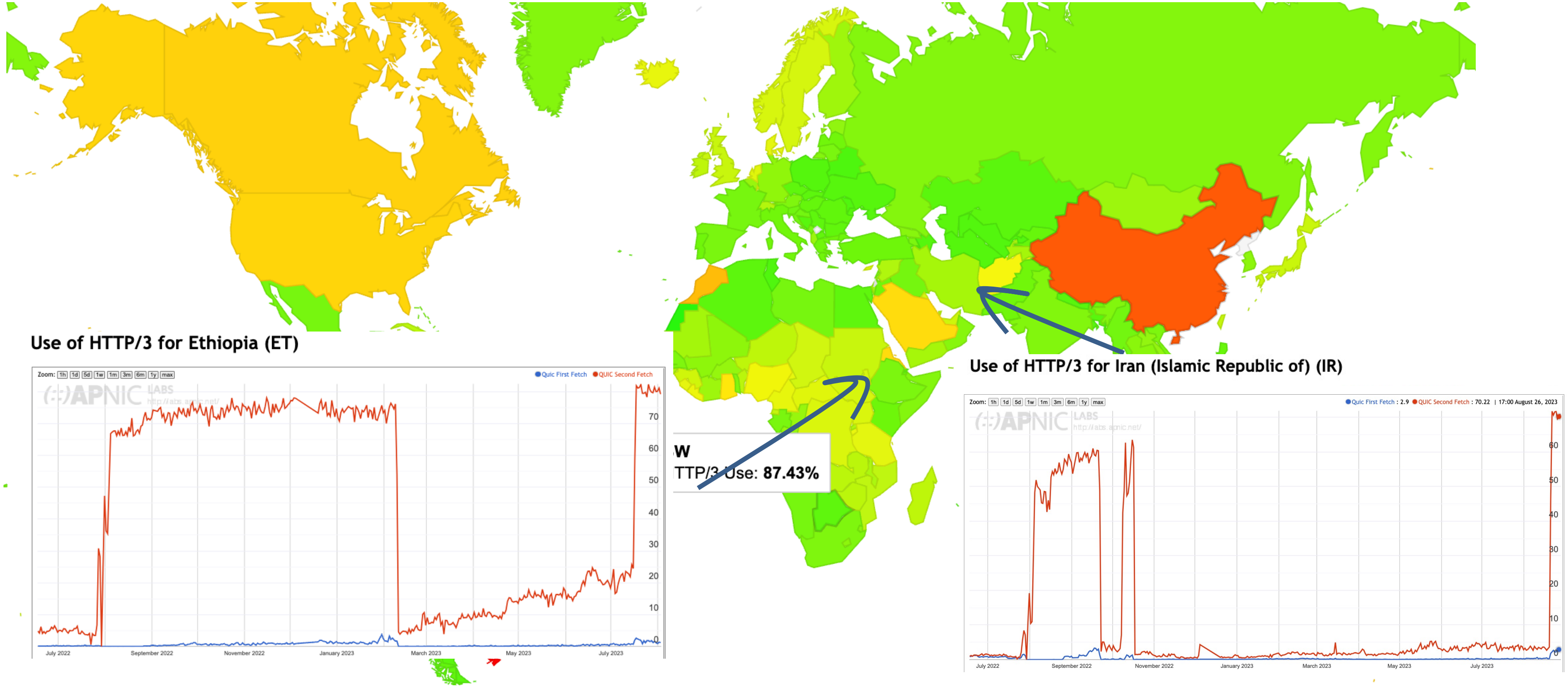
National Filtering of QUIC?



Use of HTTP/3 for Ethiopia (ET)

Use of HTTP/3 for Iran (Islamic Republic of) (IR)

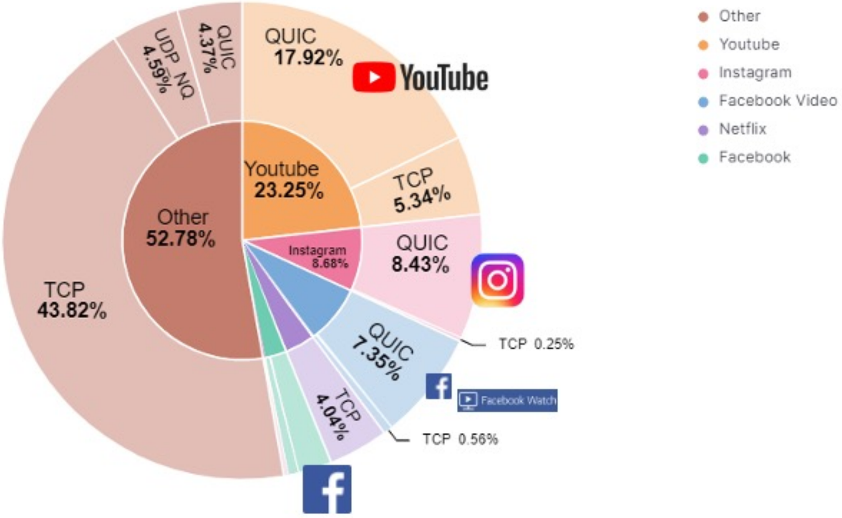
National Filtering of QUIC?



Use of HTTP/3 for Ethiopia (ET)

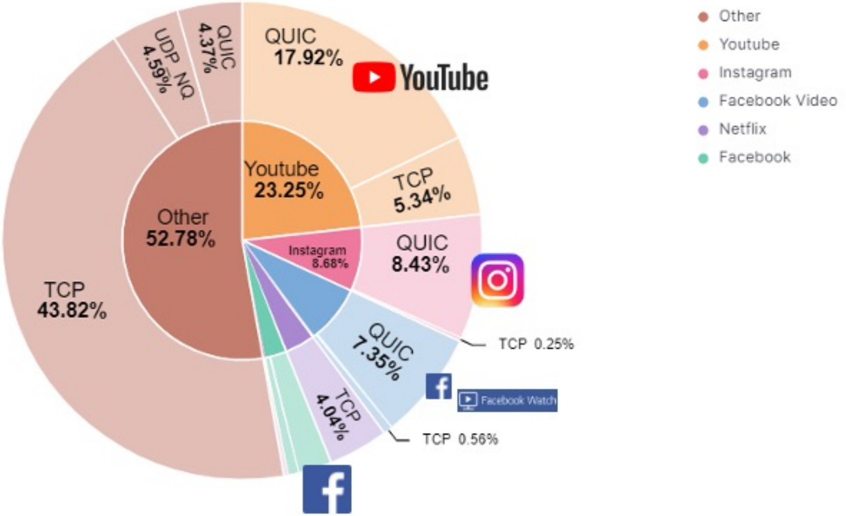
Use of HTTP/3 for Iran (Islamic Republic of) (IR)

Other Measures: Network Traffic Volume



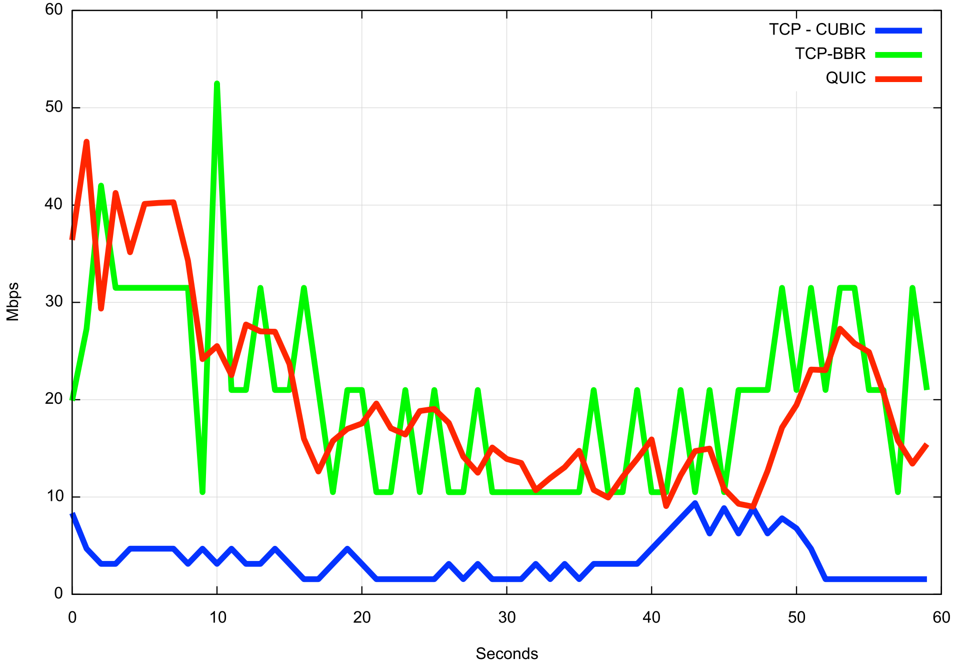
*source EU Operator 2022

Network Traffic Volume



*source EU Operator 2022

Measuring QUIC Performance



In this test (between the same endpoints) over a Starlink circuit, TCP CUBIC underperforms badly, while TCP BBR and QUIC both perform reasonably well

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!

Because QUIC is an application capability

- QUIC can interact with the platform through the UDP API, so all of QUIC can be implemented within the application. This gives the application more control over its service outcomes and reduces external dependencies

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

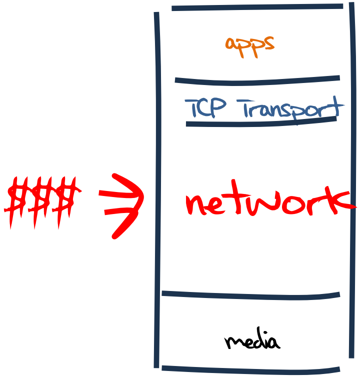
BUT:

- 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?

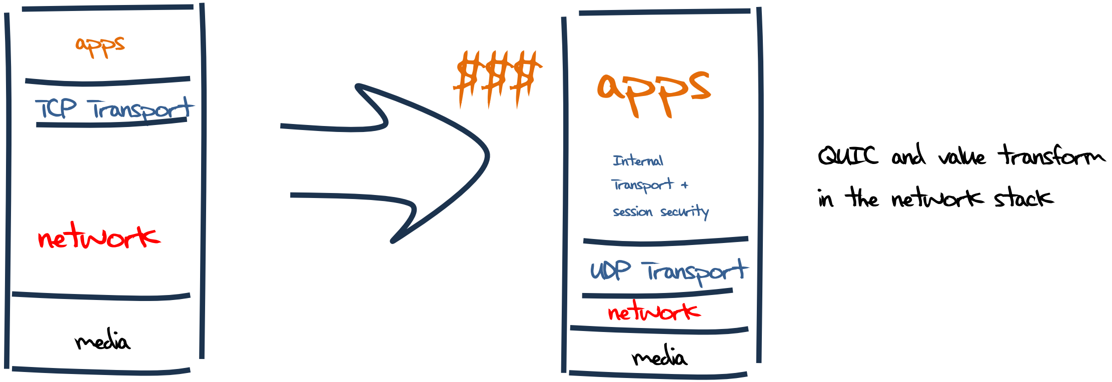
- IP was a ***network*** *protocol* that provided services to attached devices
- The network service model used by IP was minimal
  - Packets may be dropped, fragmented, duplicated, corrupted and/or reordered on their path through the network
  - It's left to the edge systems to recover from this network 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
- Networks took up a role of defending the network resource against aggressive application behaviour
- Some networks enabled user surveillance



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!"

The new Internet Space

- Each service has an ability to define its own operational behaviours that are intrinsic to that service
  - Which is the anti-definition of “interoperability”
- We have managed to minimize and commoditize the common parts of the Internet and push the valued functionality and service delivery up into each application
- Which means:
  - Standards for Interoperability is dead!
  - Open is historic!
  - The hyperscalers have won everything!

Thanks!